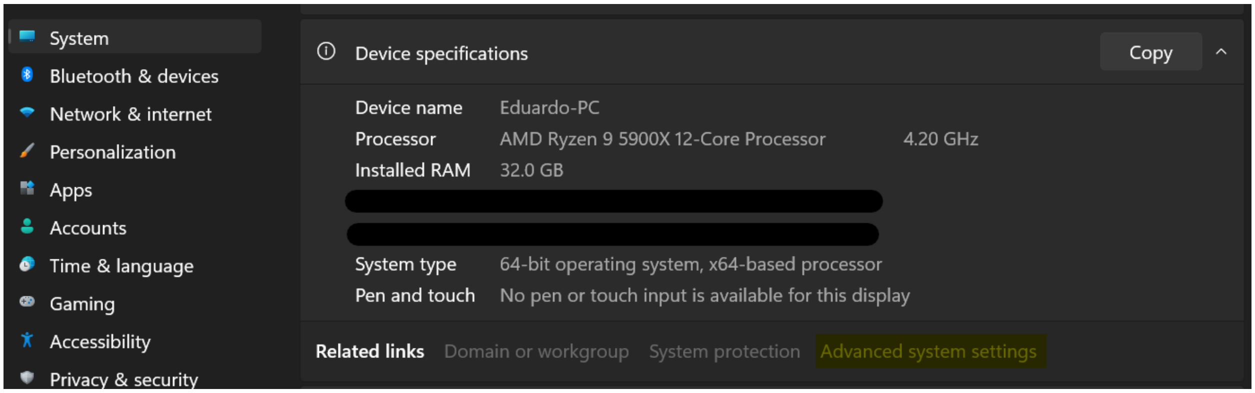
Report 7

For this week's lab, we worked on the second part of Windows Forensics. Part 1 involved Windows Crash Dump. The next Park was about Collecting Process Information. The last task was about RAM Acquisition.

The first task of the lab was to retrieve the Windows Crash Dump. When there is a system failure in windows, The OS stores the memory. This can be recovered by analyzing information on the system state, memory locations, applications, program status, etc. To create memory dumps, you can navigate to Startup and Recovery in SYSTEM > ABOUT >ADVANCED SYSTEM SETTINGS.



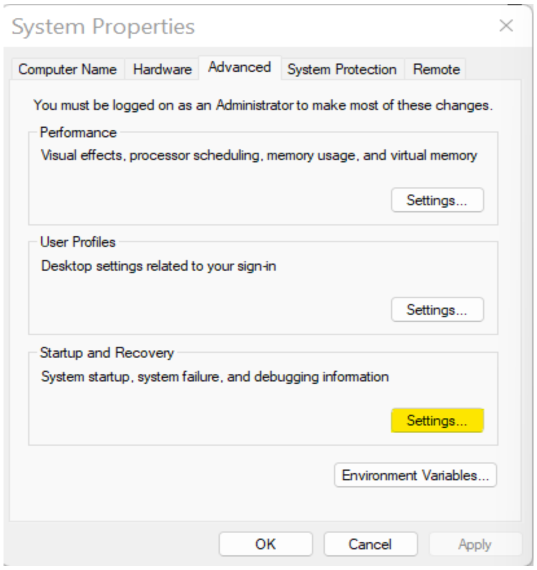
Device specifications

Device name Eduardo-PC
Processor AMD Ryzen 9 5900X 12-Core Processor 4.20 GHz
Installed RAM 32.0 GB
[redacted]
System type 64-bit operating system, x64-based processor
Pen and touch No pen or touch input is available for this display

Related links Domain or workgroup System protection Advanced system settings

Then to ADVANCED,

STARTUP AND RECOVER > SETTINGS INFORMATION]



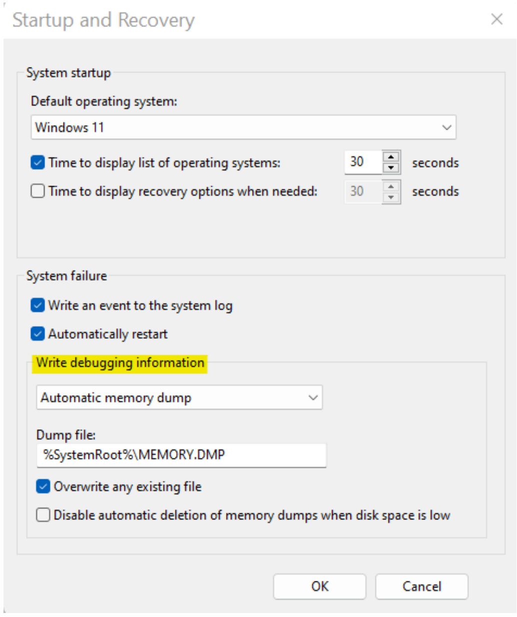
You must be logged on as an Administrator to make most of these changes.

Performance
Visual effects, processor scheduling, memory usage, and virtual memory

User Profiles
Desktop settings related to your sign-in

Startup and Recovery
System startup, system failure, and debugging information

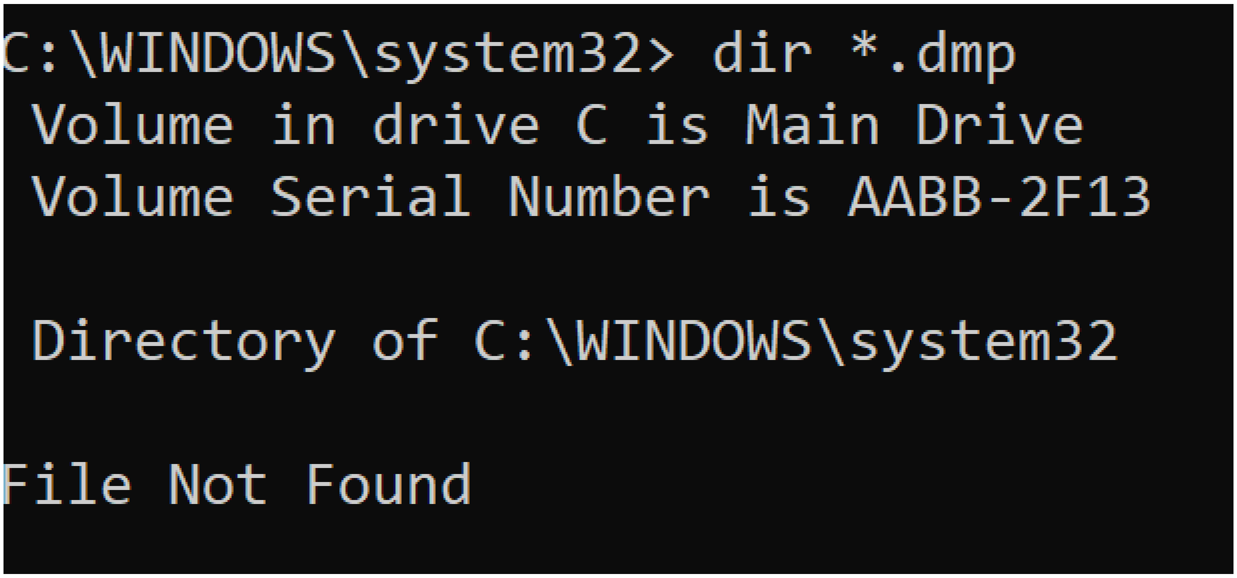
The Last Step WRITE DEBUGGING



System startup
Default operating system: Windows 11
☑ Time to display list of operating systems: 30 seconds
☐ Time to display recovery options when needed: 30 seconds

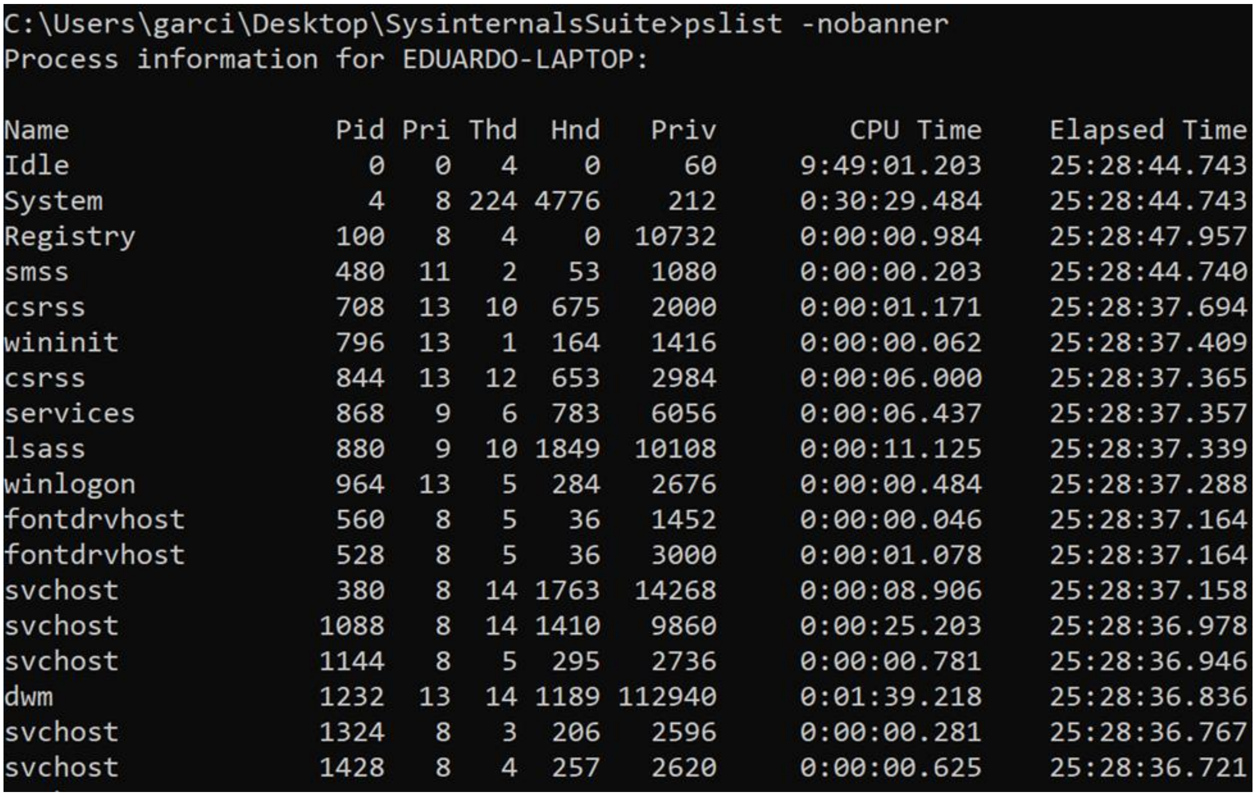
System failure
☑ Write an event to the system log
☑ Automatically restart
Write debugging information
Automatic memory dump
Dump file: %SystemRoot%\MEMORY.DMP
☑ Overwrite any existing file
☐ Disable automatic deletion of memory dumps when disk space is low

You can locate the crash dump using the command **dir *.dmp.** In my case, there was no memory dump on my PC.



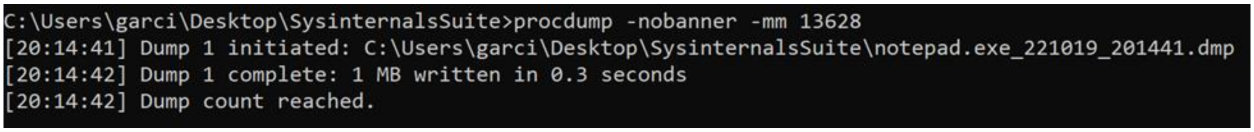
```
C:\WINDOWS\system32> dir *.dmp
 Volume in drive C is Main Drive
 Volume Serial Number is AABB-2F13

 Directory of C:\WINDOWS\system32

File Not Found
```

The next part of the lab is about the Collection Process Information. There are situations where you want to analyze certain processes rather than going through the whole memory. You can use the command **pslist -nobanner** to view all the processes.



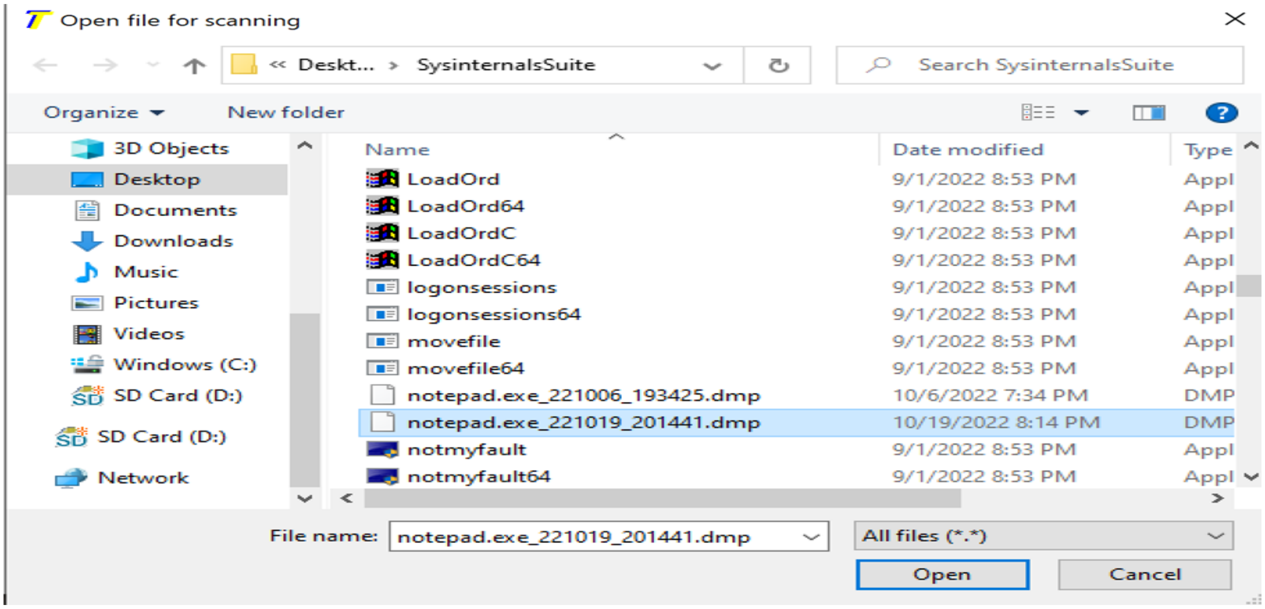
```
C:\Users\garci\Desktop\SysinternalsSuite>pslist -nobanner
Process information for EDUARDO-LAPTOP:

Name                Pid Pri Thd  Hnd   Priv        CPU Time    Elapsed Time
Idle                  0   0   4    0     60     9:49:01.203    25:28:44.743
System                4   8 224 4776    212     0:30:29.484    25:28:44.743
Registry            100   8   4    0  10732     0:00:00.984    25:28:47.957
smss                480  11   2   53   1080     0:00:00.203    25:28:44.740
csrss               708  13  10  675   2000     0:00:01.171    25:28:37.694
wininit             796  13   1  164   1416     0:00:00.062    25:28:37.409
csrss               844  13  12  653   2984     0:00:06.000    25:28:37.365
services            868   9   6  783   6056     0:00:06.437    25:28:37.357
lsass               880   9  10 1849  10108     0:00:11.125    25:28:37.339
winlogon            964  13   5  284   2676     0:00:00.484    25:28:37.288
fontdrvhost         560   8   5   36   1452     0:00:00.046    25:28:37.164
fontdrvhost         528   8   5   36   3000     0:00:01.078    25:28:37.164
svchost             380   8  14 1763  14268     0:00:08.906    25:28:37.158
svchost            1088   8  14 1410   9860     0:00:25.203    25:28:36.978
svchost            1144   8   5  295   2736     0:00:00.781    25:28:36.946
dwm                1232  13  14 1189 112940     0:01:39.218    25:28:36.836
svchost            1324   8   3  206   2596     0:00:00.281    25:28:36.767
svchost            1428   8   4  257   2620     0:00:00.625    25:28:36.721
```

Then you can use **procdump -nobanner -mm** followed by a pin number to dump a particular process. In this case, I dumped a notepad.exe process using 13628 since that was the Pid Value.



```
C:\Users\garci\Desktop\SysinternalsSuite>procdump -nobanner -mm 13628
[20:14:41] Dump 1 initiated: C:\Users\garci\Desktop\SysinternalsSuite\notepad.exe_221019_201441.dmp
[20:14:42] Dump 1 complete: 1 MB written in 0.3 seconds
[20:14:42] Dump count reached.
```

To display the contents of the dump, I used McAfee's Software called the BinText tool. I opened the program, and I located the file. It was located in the C:\Users\garci\Desktop\SysInternalsSuite folder. Here is below the dumped notepad.exe

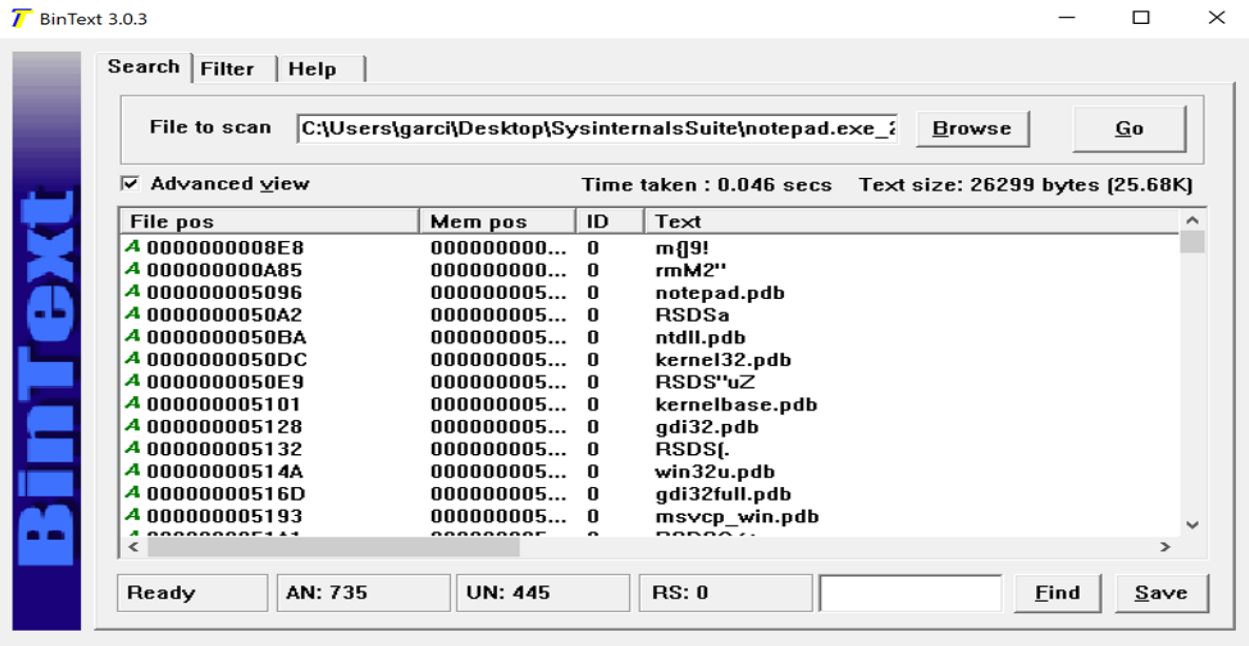


Name	Date modified	Type
LoadOrd	9/1/2022 8:53 PM	Appl
LoadOrd64	9/1/2022 8:53 PM	Appl
LoadOrdC	9/1/2022 8:53 PM	Appl
LoadOrdC64	9/1/2022 8:53 PM	Appl
logonsessions	9/1/2022 8:53 PM	Appl
logonsessions64	9/1/2022 8:53 PM	Appl
movefile	9/1/2022 8:53 PM	Appl
movefile64	9/1/2022 8:53 PM	Appl
notepad.exe_221006_193425.dmp	10/6/2022 7:34 PM	DMP
notepad.exe_221019_201441.dmp	10/19/2022 8:14 PM	DMP
notmyfault	9/1/2022 8:53 PM	Appl
notmyfault64	9/1/2022 8:53 PM	Appl

File name: notepad.exe_221019_201441.dmp

All files (*.*)

Here below are the contents of the dumped file.



File to scan: C:\Users\garci\Desktop\SysinternalsSuite\notepad.exe_2

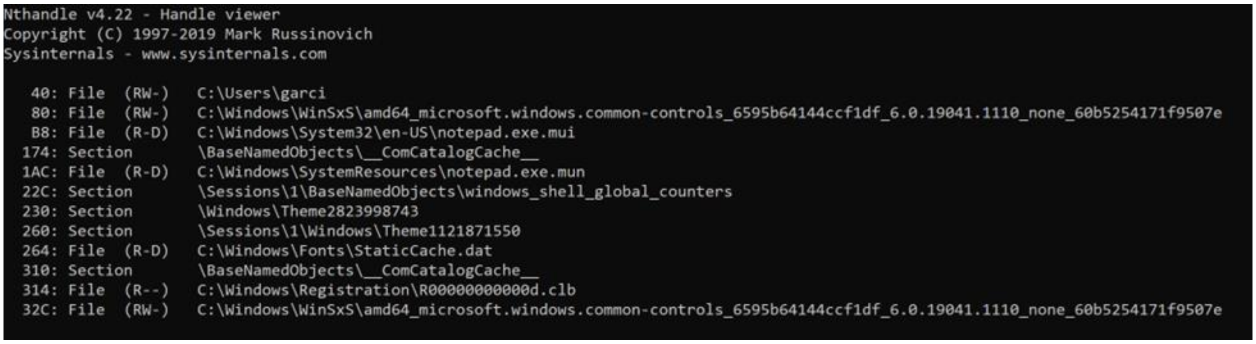
☑ Advanced view

Time taken : 0.046 secs Text size: 26299 bytes (25.68K)

File pos	Mem pos	ID	Text
A 0000000008E8	000000000...	0	m{]9!
A 000000000A85	000000000...	0	rmM2"
A 000000005096	000000005...	0	notepad.pdb
A 0000000050A2	000000005...	0	RSDSa
A 0000000050BA	000000005...	0	ntdll.pdb
A 0000000050DC	000000005...	0	kernel32.pdb
A 0000000050E9	000000005...	0	RSDS"uZ
A 000000005101	000000005...	0	kernelbase.pdb
A 000000005128	000000005...	0	gdi32.pdb
A 000000005132	000000005...	0	RSDS(.
A 00000000514A	000000005...	0	win32u.pdb
A 00000000516D	000000005...	0	gdi32full.pdb
A 000000005193	000000005...	0	msvcp_win.pdb

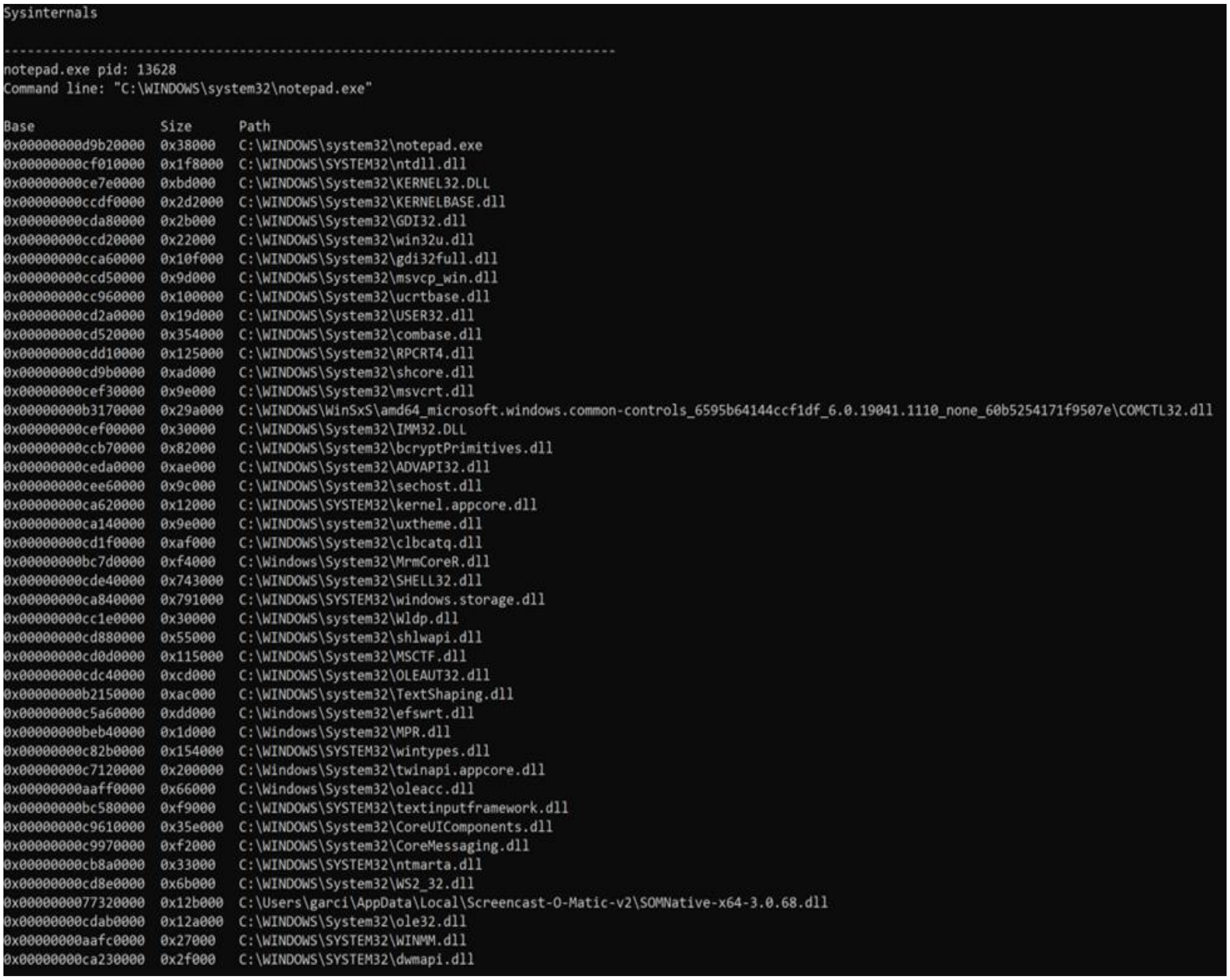
Ready | AN: 735 | UN: 445 | RS: 0

A process contains a unique identifier called a PID. A set of handles are also created. These can be used by internal functions to access resources. The **handle** command will show a long list of all processes with their handlers. So, to only look for a particular process, you use the **handle -p** followed by the PID number. In my case, I used the same value for the notepad which was 13628



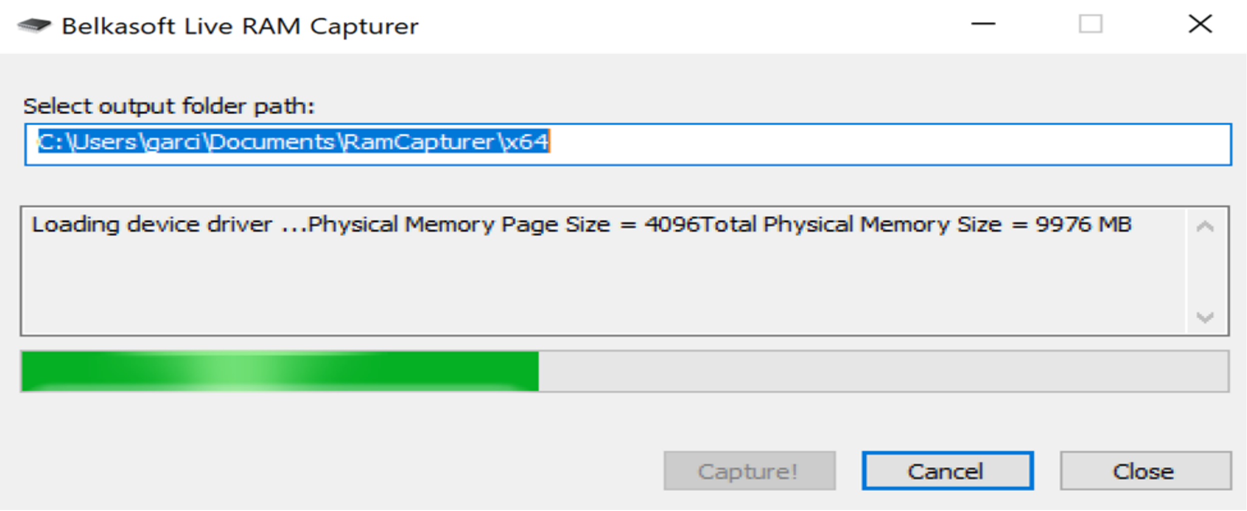
```
Nthandle v4.22 - Handle viewer
Copyright (C) 1997-2019 Mark Russinovich
Sysinternals - www.sysinternals.com

   40: File  (RW-)   C:\Users\garci
   80: File  (RW-)   C:\Windows\WinSxS\amd64_microsoft.windows.common-controls_6595b64144ccf1df_6.0.19041.1110_none_60b5254171f9507e
   B8: File  (R-D)   C:\Windows\System32\en-US\notepad.exe.mui
  174: Section       \BaseNamedObjects\__ComCatalogCache__
  1AC: File  (R-D)   C:\Windows\SystemResources\notepad.exe.mun
  22C: Section       \Sessions\1\BaseNamedObjects\windows_shell_global_counters
  230: Section       \Windows\Theme2823998743
  260: Section       \Sessions\1\Windows\Theme1121871550
  264: File  (R-D)   C:\Windows\Fonts\StaticCache.dat
  310: Section       \BaseNamedObjects\__ComCatalogCache__
  314: File  (R--)   C:\Windows\Registration\R00000000000d.clb
  32C: File  (RW-)   C:\Windows\WinSxS\amd64_microsoft.windows.common-controls_6595b64144ccf1df_6.0.19041.1110_none_60b5254171f9507e
```

If you want to list all executable and dynamic link libraries (DLL files) that are loaded into processes, the command to do this is **listdlls.** Also, you can also search for a particular process as well. To do this, you use the **listdlls** followed by the process that you want. This output below is the executable and DLL files for notepad.exe. Here is the command I used **listdlls notepad.exe**



```
Sysinternals

------------------------------------------------------------------------------
notepad.exe pid: 13628
Command line: "C:\WINDOWS\system32\notepad.exe"

Base                Size      Path
0x00000000d9b20000  0x38000   C:\WINDOWS\system32\notepad.exe
0x00000000cf010000  0x1f8000  C:\WINDOWS\SYSTEM32\ntdll.dll
0x00000000ce7e0000  0xbd000   C:\WINDOWS\System32\KERNEL32.DLL
0x00000000ccdf0000  0x2d2000  C:\WINDOWS\System32\KERNELBASE.dll
0x00000000cda80000  0x2b000   C:\WINDOWS\System32\GDI32.dll
0x00000000ccd20000  0x22000   C:\WINDOWS\System32\win32u.dll
0x00000000cca60000  0x10f000  C:\WINDOWS\System32\gdi32full.dll
0x00000000ccd50000  0x9d000   C:\WINDOWS\System32\msvcp_win.dll
0x00000000cc960000  0x100000  C:\WINDOWS\System32\ucrtbase.dll
0x00000000cd2a0000  0x19d000  C:\WINDOWS\System32\USER32.dll
0x00000000cd520000  0x354000  C:\WINDOWS\System32\combase.dll
0x00000000cdd10000  0x125000  C:\WINDOWS\System32\RPCRT4.dll
0x00000000cd9b0000  0xad000   C:\WINDOWS\System32\shcore.dll
0x00000000cef30000  0x9e000   C:\WINDOWS\System32\msvcrt.dll
0x00000000b3170000  0x29a000  C:\WINDOWS\WinSxS\amd64_microsoft.windows.common-controls_6595b64144ccf1df_6.0.19041.1110_none_60b5254171f9507e\COMCTL32.dll
0x00000000cef00000  0x30000   C:\WINDOWS\System32\IMM32.DLL
0x00000000ccb70000  0x82000   C:\WINDOWS\System32\bcryptPrimitives.dll
0x00000000ceda0000  0xae000   C:\WINDOWS\System32\ADVAPI32.dll
0x00000000cee60000  0x9c000   C:\WINDOWS\System32\sechost.dll
0x00000000ca620000  0x12000   C:\WINDOWS\SYSTEM32\kernel.appcore.dll
0x00000000ca140000  0x9e000   C:\WINDOWS\system32\uxtheme.dll
0x00000000cd1f0000  0xaf000   C:\WINDOWS\System32\clbcatq.dll
0x00000000bc7d0000  0xf4000   C:\Windows\System32\MrmCoreR.dll
0x00000000cde40000  0x743000  C:\WINDOWS\System32\SHELL32.dll
0x00000000ca840000  0x791000  C:\WINDOWS\SYSTEM32\windows.storage.dll
0x00000000cc1e0000  0x30000   C:\WINDOWS\system32\Wldp.dll
0x00000000cd880000  0x55000   C:\WINDOWS\System32\shlwapi.dll
0x00000000cd0d0000  0x115000  C:\WINDOWS\System32\MSCTF.dll
0x00000000cdc40000  0xcd000   C:\WINDOWS\System32\OLEAUT32.dll
0x00000000b2150000  0xac000   C:\WINDOWS\system32\TextShaping.dll
0x00000000c5a60000  0xdd000   C:\Windows\System32\efswrt.dll
0x00000000beb40000  0x1d000   C:\Windows\System32\MPR.dll
0x00000000c82b0000  0x154000  C:\WINDOWS\SYSTEM32\wintypes.dll
0x00000000c7120000  0x200000  C:\Windows\System32\twinapi.appcore.dll
0x00000000aaff0000  0x66000   C:\Windows\System32\oleacc.dll
0x00000000bc580000  0xf9000   C:\WINDOWS\SYSTEM32\textinputframework.dll
0x00000000c9610000  0x35e000  C:\WINDOWS\System32\CoreUIComponents.dll
0x00000000c9970000  0xf2000   C:\WINDOWS\System32\CoreMessaging.dll
0x00000000cb8a0000  0x33000   C:\WINDOWS\SYSTEM32\ntmarta.dll
0x00000000cd8e0000  0x6b000   C:\WINDOWS\System32\WS2_32.dll
0x0000000077320000  0x12b000  C:\Users\garci\AppData\Local\Screencast-O-Matic-v2\SOMNative-x64-3.0.68.dll
0x00000000cdab0000  0x12a000  C:\WINDOWS\System32\ole32.dll
0x00000000aafc0000  0x27000   C:\WINDOWS\SYSTEM32\WINMM.dll
0x00000000ca230000  0x2f000   C:\WINDOWS\SYSTEM32\dwmapi.dll
```

The last part of the lab was about RAM Acquisition. RAM can be acquired during the live acquisition. In other words when it is powered on. I used Belkasoft RAM Capturer to capture the memory dump of my RAM.



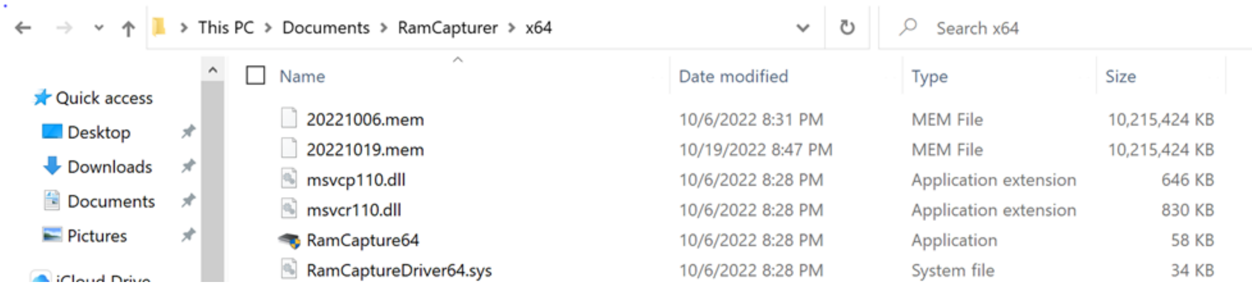
Belkasoft Live RAM Capturer

Select output folder path:

C:\Users\garci\Documents\RamCapturer\x64

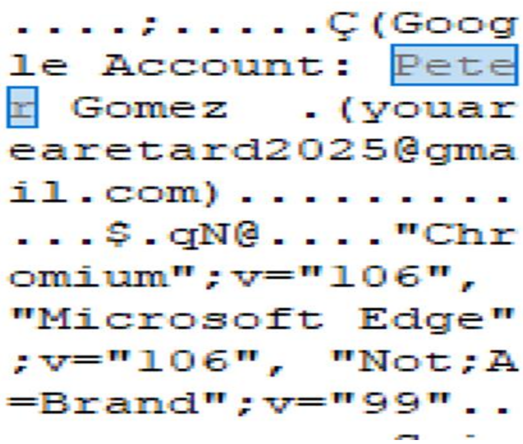
Loading device driver ...Physical Memory Page Size = 4096Total Physical Memory Size = 9976 MB

The dump is located in the **Document > RamCapturer>x64** folder. The file is 20221019.mem.



Name	Date modified	Type	Size
20221006.mem	10/6/2022 8:31 PM	MEM File	10,215,424 KB
20221019.mem	10/19/2022 8:47 PM	MEM File	10,215,424 KB
msvcp110.dll	10/6/2022 8:28 PM	Application extension	646 KB
msvcr110.dll	10/6/2022 8:28 PM	Application extension	830 KB
RamCapture64	10/6/2022 8:28 PM	Application	58 KB
RamCaptureDriver64.sys	10/6/2022 8:28 PM	System file	34 KB

I created a new Gmail account before starting this part of the lab. I used the name peter. I moved the 20221019.mem file and placed it in the HxD program. Here below I searched the mem file using the name "peter". Below shows the email that I used to create the account and the web browser I utilized to create the Gmail account.



```
....;.....Ç(Goog
le Account: Pete
r Gomez  .(youar
earetard2025@gma
il.com).........
...$.qN@...."Chr
omium";v="106",
"Microsoft Edge"
;v="106", "Not;A
=Brand";v="99"..
```