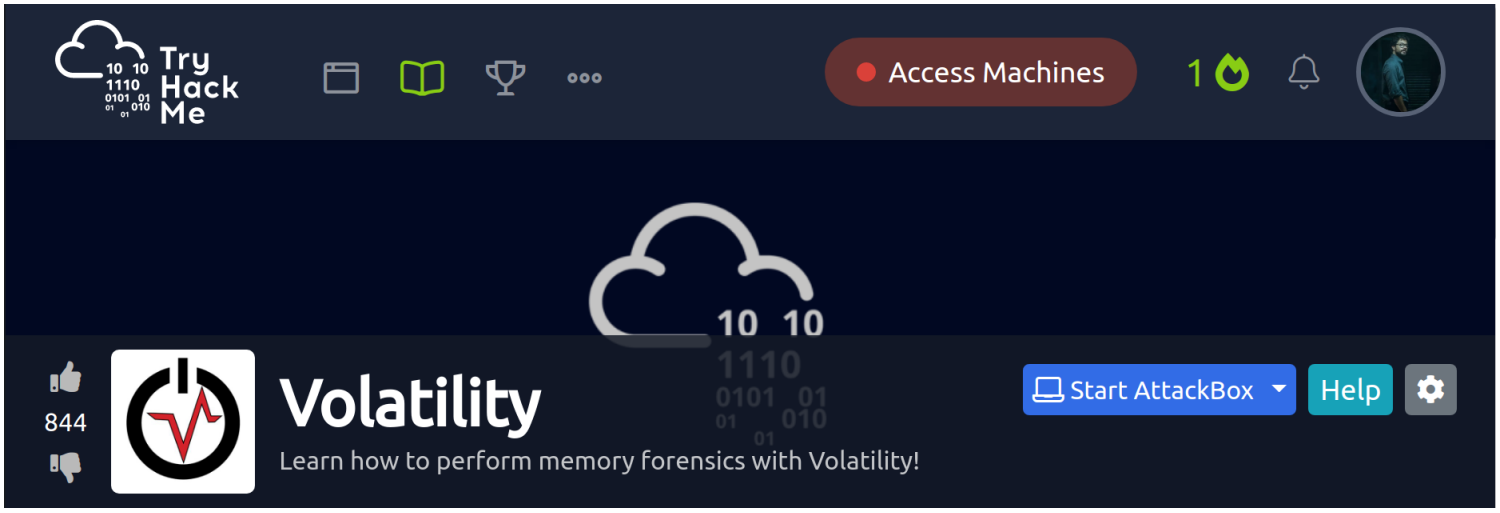
Volatility - Memory Forensics

okay , so here we will be looking at a tool named volatility that is used to perform memory forensics , original resource to this document is tryhackme.com :



lets get to it .

Introduction to volatility

Volatility is a memory forensic tool , it is the gold standard tool for forensics in incident response , its functionality can also be expanded via plugins that it offers .

to install :

there was weird issue with downloading either it was my browser or the site not functioning properly itself , i found a workaround and used axel to get the job done :

```
┌──(root㉿kali)-[/home/kali/digital-forensics/volatility3-1.0.0]
└─# axel http://downloads.volatilityfoundation.org/releases/2.6/volatility_2.6_lin64_standalone.zip
Initializing download: http://downloads.volatilityfoundation.org/releases/2.6/volatility_2.6_lin64_standalone.zip
File size: 14.0551 Megabyte(s) (14737820 bytes)
Opening output file volatility_2.6_lin64_standalone.zip
Starting download

[100%] [.....] [ 560.1KB/s] [00:00]

Downloaded 14.0551 Megabyte(s) in 25 second(s). (560.11 KB/s)
```

now unzip the file :

```
┌──(root㉿kali)-[/home/kali/digital-forensics]
└─# unzip volatility_2.6_lin64_standalone.zip
Archive:  volatility_2.6_lin64_standalone.zip
   creating: volatility_2.6_lin64_standalone/
  inflating: volatility_2.6_lin64_standalone/AUTHORS.txt
  inflating: volatility_2.6_lin64_standalone/CREDITS.txt
  inflating: volatility_2.6_lin64_standalone/LEGAL.txt
  inflating: volatility_2.6_lin64_standalone/LICENSE.txt
  inflating: volatility_2.6_lin64_standalone/README.txt
  inflating: volatility_2.6_lin64_standalone/volatility_2.6_lin64_standalone
```

and see if it runs properly :

```
┌──(root㉿kali)-[/home/kali/digital-forensics/volatility_2.6_lin64_standalone]
└─# ./volatility_2.6_lin64_standalone -h
Volatility Foundation Volatility Framework 2.6
Usage: Volatility - A memory forensics analysis platform.

Options:
  -h, --help            list all available options and their default values.
                        Default values may be set in the configuration file
                        (/etc/volatilityrc)
  --conf-file=/root/.volatilityrc
                        User based configuration file
  -d, --debug           Debug volatility
  --plugins=PLUGINS     Additional plugin directories to use (colon separated)
  --info                Print information about all registered objects
  --cache-directory=/root/.cache/volatility
```

and it works like a charm .

Obtaining Memory Samples

now , to get started with using the tool we need some memory samples and to obtain those samples there can be 2 situations whether the machine is offline or online (means a live machine)

in case of online/live machines we can use some tools such as :

• FTK Imager - Link
• Redline - Link *Requires registration but Redline has a very nice GUI
• DumpIt.exe
• win32dd.exe / win64dd.exe - *Has fantastic psexec support, great for IT departments if your EDR solution doesn't support this

these tools outputs a (.raw) file that we can analyze , this .raw file contains the image of system memory .

then , in case of offline machine we can copy the %SystemDrive%/hiberfil.sys file from the drive , until the drive is not encrypted .

well , this hiberfil.sys file is a compressed memory image in windows systems , created from previous boot . to improve loading times in windows , this file can be used for forensics .

things get a bit interesting when talking about virtual machines , we can look for these files in the VM's directory :

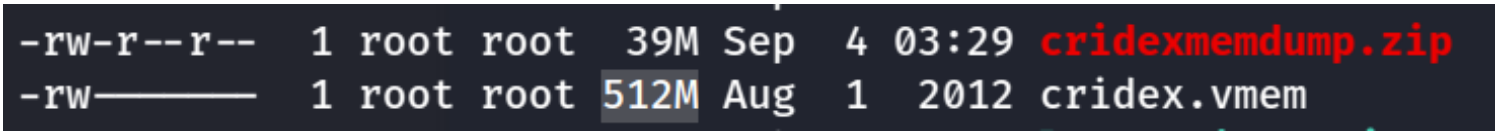
• VMware - .vmem file
• Hyper-V - .bin file
• Parallels - .mem file
• VirtualBox - .sav file *This is only a partial memory file. You'll need to dump memory like a normal bare-metal system for this hypervisor

these files can easily be copied without any disturbance to the actual running VM , preserving its forensic integrity .

Examining Our Patient

okay , so there is a task that we will perform here to have a more practical and better understanding of how to use this tool and understand the underlying concepts .

so there is a memory sample given in the task that we will analyze .



unzip the cridexmemdump.zip to get the cridex.vmem file .

now lets start the forensic activity .

first we have to figure out what profile do we need to use , profile determines how our file is to be treated and as each version of windows is different we need to find the right profile for our case :

we will use -f to specify the file and use imageinfo plugin to look for suggested profiles :

```
┌──(root㉿kali)-[/home/kali/digital-forensics/volatility_2.6_lin64_standalone]
└─# ./volatility_2.6_lin64_standalone -f cridex.vmem imageinfo
Volatility Foundation Volatility Framework 2.6
INFO    : volatility.debug    : Determining profile based on KDBG search...
          Suggested Profile(s) : WinXPSP2x86, WinXPSP3x86 (Instantiated with WinXPSP2x86)
                     AS Layer1 : IA32PagedMemoryPae (Kernel AS)
                     AS Layer2 : FileAddressSpace (/home/kali/digital-forensics/volatility_2.6_lin64_standalone/cridex.vmem)
                      PAE type : PAE
                           DTB : 0x2fe000L
                          KDBG : 0x80545ae0L
          Number of Processors : 1
     Image Type (Service Pack) : 3
                KPCR for CPU 0 : 0xffdff000L
             KUSER_SHARED_DATA : 0xffdf0000L
           Image date and time : 2012-07-22 02:45:08 UTC+0000
     Image local date and time : 2012-07-21 22:45:08 -0400
```

now we got two profiles here and we need to find the correct one , so the first one we will be using to list the processes from the memory dump :

```
┌──(root㉿kali)-[/home/kali/digital-forensics/volatility_2.6_lin64_standalone]
└─# ./volatility_2.6_lin64_standalone -f cridex.vmem --profile=WinXPSP2x86 pslist
Volatility Foundation Volatility Framework 2.6
Offset(V)  Name                    PID   PPID   Thds     Hnds   Sess  Wow64 Start                          Exit
---------- -------------------- ------ ------ ------ -------- ------ ------ ------------------------------ ------------------------------
0x823c89c8 System                    4      0     53      240 ------      0
0x822f1020 smss.exe                368      4      3       19 ------      0 2012-07-22 02:42:31 UTC+0000
0x822a0598 csrss.exe               584    368      9      326      0      0 2012-07-22 02:42:32 UTC+0000
0x82298700 winlogon.exe            608    368     23      519      0      0 2012-07-22 02:42:32 UTC+0000
0x81e2ab28 services.exe            652    608     16      243      0      0 2012-07-22 02:42:32 UTC+0000
0x81e2a3b8 lsass.exe               664    608     24      330      0      0 2012-07-22 02:42:32 UTC+0000
0x82311360 svchost.exe             824    652     20      194      0      0 2012-07-22 02:42:33 UTC+0000
0x81e29ab8 svchost.exe             908    652      9      226      0      0 2012-07-22 02:42:33 UTC+0000
0x823001d0 svchost.exe            1004    652     64     1118      0      0 2012-07-22 02:42:33 UTC+0000
0x821dfda0 svchost.exe            1056    652      5       60      0      0 2012-07-22 02:42:33 UTC+0000
0x82295650 svchost.exe            1220    652     15      197      0      0 2012-07-22 02:42:35 UTC+0000
0x821dea70 explorer.exe           1484   1464     17      415      0      0 2012-07-22 02:42:36 UTC+0000
0x81eb17b8 spoolsv.exe            1512    652     14      113      0      0 2012-07-22 02:42:36 UTC+0000
0x81e7bda0 reader_sl.exe          1640   1484      5       39      0      0 2012-07-22 02:42:36 UTC+0000
0x820e8da0 alg.exe                 788    652      7      104      0      0 2012-07-22 02:43:01 UTC+0000
0x821fcda0 wuauclt.exe            1136   1004      8      173      0      0 2012-07-22 02:43:46 UTC+0000
0x8205bda0 wuauclt.exe            1588   1004      5      132      0      0 2012-07-22 02:44:01 UTC+0000
```

and it works so our profile is WinXPSP2x86

we can also view network connections at the time of image creation with netscan :

```
┌──(root㉿kali)-[/home/kali/digital-forensics/volatility_2.6_lin64_standalone]
└─# ./volatility_2.6_lin64_standalone -f cridex.vmem --profile=WinXPSP2x86 netscan
Volatility Foundation Volatility Framework 2.6
ERROR   : volatility.debug    : This command does not support the profile WinXPSP2x86
```

however in our case the host does not support it but it can still be useful where it does support

malware can hide itself in associated processes , we can look for hidden processes with psxview command :

```
┌──(root㉿kali)-[/home/kali/digital-forensics/volatility_2.6_lin64_standalone]
└─# ./volatility_2.6_lin64_standalone -f cridex.vmem --profile=WinXPSP2x86 psxview
Volatility Foundation Volatility Framework 2.6
Offset(P)  Name                    PID pslist psscan thrdproc pspcid csrss session deskthrd ExitTime
---------- -------------------- ------ ------ ------ -------- ------ ----- ------- -------- --------
0x02498700 winlogon.exe            608 True   True   True     True   True  True    True
0x02511360 svchost.exe             824 True   True   True     True   True  True    True
0x022e8da0 alg.exe                 788 True   True   True     True   True  True    True
0x020b17b8 spoolsv.exe            1512 True   True   True     True   True  True    True
0x0202ab28 services.exe            652 True   True   True     True   True  True    True
0x02495650 svchost.exe            1220 True   True   True     True   True  True    True
0x0207bda0 reader_sl.exe          1640 True   True   True     True   True  True    True
0x025001d0 svchost.exe            1004 True   True   True     True   True  True    True
0x02029ab8 svchost.exe             908 True   True   True     True   True  True    True
0x023fcda0 wuauclt.exe            1136 True   True   True     True   True  True    True
0x0225bda0 wuauclt.exe            1588 True   True   True     True   True  True    True
0x0202a3b8 lsass.exe               664 True   True   True     True   True  True    True
0x023dea70 explorer.exe           1484 True   True   True     True   True  True    True
0x023dfda0 svchost.exe            1056 True   True   True     True   True  True    True
0x024f1020 smss.exe                368 True   True   True     True   False False   False
0x025c89c8 System                    4 True   True   True     True   False False   False
0x024a0598 csrss.exe               584 True   True   True     True   False True    True
```

to have more focused and refined search of injected processes we can run ldrmodules , which shows are processes that are inint , inmem and inload , if any of these column is false it is likely an injected malware :

```
┌──(root㉿kali)-[/home/kali/digital-forensics/volatility_2.6_lin64_standalone]
└─# ./volatility_2.6_lin64_standalone -f cridex.vmem --profile=WinXPSP2x86 ldrmodules
Volatility Foundation Volatility Framework 2.6
Pid      Process              Base       InLoad InInit InMem MappedPath
-------- -------------------- ---------- ------ ------ ----- ----------
       4 System               0x7c900000 False  False  False \WINDOWS\system32\ntdll.dll
     368 smss.exe             0x48580000 True   False  True  \WINDOWS\system32\smss.exe
     368 smss.exe             0x7c900000 True   True   True  \WINDOWS\system32\ntdll.dll
     584 csrss.exe            0x00460000 False  False  False \WINDOWS\Fonts\vgasys.fon
     584 csrss.exe            0x4a680000 True   False  True  \WINDOWS\system32\csrss.exe
     584 csrss.exe            0x75b40000 True   True   True  \WINDOWS\system32\csrsrv.dll
     584 csrss.exe            0x75b50000 True   True   True  \WINDOWS\system32\basesrv.dll
     584 csrss.exe            0x7e720000 True   True   True  \WINDOWS\system32\sxs.dll
     584 csrss.exe            0x77e70000 True   True   True  \WINDOWS\system32\rpcrt4.dll
     584 csrss.exe            0x7c800000 True   True   True  \WINDOWS\system32\kernel32.dll
     584 csrss.exe            0x77dd0000 True   True   True  \WINDOWS\system32\advapi32.dll
     584 csrss.exe            0x77fe0000 True   True   True  \WINDOWS\system32\secur32.dll
     584 csrss.exe            0x7e410000 True   True   True  \WINDOWS\system32\user32.dll
```

to find malware and dump the code use malfind :

```
┌──(root㉿kali)-[/home/kali/digital-forensics/volatility_2.6_lin64_standalone]
└─# ./volatility_2.6_lin64_standalone -f cridex.vmem --profile=WinXPSP2x86 malfind -D .
Volatility Foundation Volatility Framework 2.6
Process: csrss.exe Pid: 584 Address: 0x7f6f0000
Vad Tag: Vad  Protection: PAGE_EXECUTE_READWRITE
Flags: Protection: 6

0x7f6f0000  c8 00 00 00 91 01 00 00 ff ee ff ee 08 70 00 00   .............p..
0x7f6f0010  08 00 00 00 00 fe 00 00 00 00 10 00 00 20 00 00   ................
0x7f6f0020  00 02 00 00 00 20 00 00 8d 01 00 00 ff ef fd 7f   ................
0x7f6f0030  03 00 08 06 00 00 00 00 00 00 00 00 00 00 00 00   ................

0x7f6f0000 c8000000        ENTER 0x0, 0x0
0x7f6f0004 91              XCHG ECX, EAX
0x7f6f0005 0100            ADD [EAX], EAX
0x7f6f0007 00ff            ADD BH, BH
0x7f6f0009 ee              OUT DX, AL
0x7f6f000a ff              DB 0xff
0x7f6f000b ee              OUT DX, AL
0x7f6f000c 087000          OR [EAX+0x0], DH
```

-D specifies the directory to dump the code , which i used “.” to denote current directory

to list all the dll's in the memory use dlllist :

```
┌──(root㉿kali)-[/home/kali/digital-forensics/volatility_2.6_lin64_standalone]
└─# ./volatility_2.6_lin64_standalone -f cridex.vmem --profile=WinXPSP2x86 dlllist
Volatility Foundation Volatility Framework 2.6
************************************************************************
System pid:      4
Unable to read PEB for task.
************************************************************************
smss.exe pid:    368
Command line : \SystemRoot\System32\smss.exe


Base             Size  LoadCount Path
---------- ---------- ---------- ----
0x48580000     0xf000     0xffff \SystemRoot\System32\smss.exe
0x7c900000    0xaf000     0xffff C:\WINDOWS\system32\ntdll.dll
************************************************************************
csrss.exe pid:    584
Command line : C:\WINDOWS\system32\csrss.exe ObjectDirectory=\Windows SharedSection=1024,307
Dll=winsrv:UserServerDllInitialization,3 ServerDll=winsrv:ConServerDllInitialization,2 Profi
Service Pack 3

Base             Size  LoadCount Path
---------- ---------- ---------- ----
0x4a680000     0x5000     0xffff \??\C:\WINDOWS\system32\csrss.exe
0x7c900000    0xaf000     0xffff C:\WINDOWS\system32\ntdll.dll
0x75b40000     0xb000     0xffff C:\WINDOWS\system32\CSRSRV.dll
0x75b50000    0x10000        0x3 C:\WINDOWS\system32\basesrv.dll
```

to dump all the dll's related to a process we need to specify --pid and dlldump module with -D to provide the directory where to dump :

```
┌──(root㉿kali)-[/home/kali/digital-forensics/volatility_2.6_lin64_standalone]
└─# ./volatility_2.6_lin64_standalone -f cridex.vmem --profile=WinXPSP2x86 --pid=584 dlldump -D .
Volatility Foundation Volatility Framework 2.6
Process(V) Name             Module Base Module Name          Result
---------- ---------------- ----------- -------------------- ------
0x822a0598 csrss.exe        0x04a680000 csrss.exe            OK: module.584.24a0598.4a680000.dll
0x822a0598 csrss.exe        0x07c900000 ntdll.dll            OK: module.584.24a0598.7c900000.dll
0x822a0598 csrss.exe        0x075b40000 CSRSRV.dll           OK: module.584.24a0598.75b40000.dll
0x822a0598 csrss.exe        0x077f10000 GDI32.dll            OK: module.584.24a0598.77f10000.dll
0x822a0598 csrss.exe        0x07e720000 sxs.dll              OK: module.584.24a0598.7e720000.dll
0x822a0598 csrss.exe        0x077e70000 RPCRT4.dll           OK: module.584.24a0598.77e70000.dll
0x822a0598 csrss.exe        0x077dd0000 ADVAPI32.dll         OK: module.584.24a0598.77dd0000.dll
0x822a0598 csrss.exe        0x077fe0000 Secur32.dll          OK: module.584.24a0598.77fe0000.dll
0x822a0598 csrss.exe        0x075b50000 basesrv.dll          OK: module.584.24a0598.75b50000.dll
0x822a0598 csrss.exe        0x07c800000 KERNEL32.dll         OK: module.584.24a0598.7c800000.dll
0x822a0598 csrss.exe        0x07e410000 USER32.dll           OK: module.584.24a0598.7e410000.dll
0x822a0598 csrss.exe        0x075b60000 winsrv.dll           OK: module.584.24a0598.75b60000.dll
```

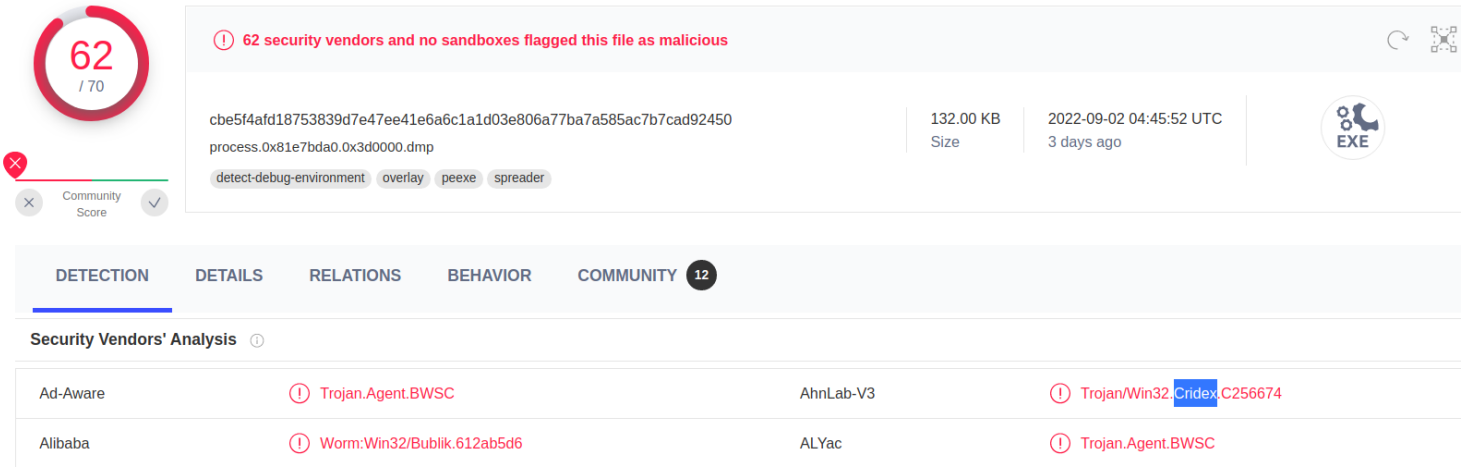
and we are done for collecting some samples which we can further analyze.

Post Actions

now we have some dll's and files which we think are malicious , to confirm that we can use online malware scanners to test our hypothesis :

we can use VirusTotal and Hybrid Analysis : VirusTotal - Link Hybrid Analysis - Link

once analyzing all file one by one we found a .dmp file that is malicious and is the malware named cridex which infected the PC :



and we have successfully found the malware and our task is done for now .

Extra Resources

Interested in going further? Here's a slew of awesome resources (both paid and free in no particular order) to check out and learn more!
AlienVault Open Threat Exchange (OTX) - An open-source threat tracking system. Create pulses based on your malware analysis work and check out the work of others. Link

SANS 408 - Windows Forensic Analysis Link

"Memory Forensics with Vol(a|u)tility" - A great talk on learning the basics of Volatility and the GUI plugin VolUtility made by @chupath1ngee
"The Art of Memory Forensics" - Link

MemLabs - A collection of CTF-style memory forensic labs
Link