



What Malware Authors Don't Want you to Know - *Evasive Hollow Process Injection*

Monnappa K A





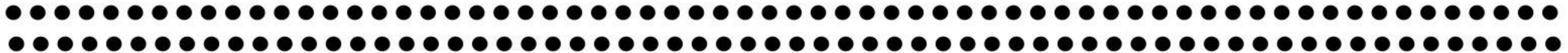
Who AM I

Monnappa K A

- Info Security Investigator - Cisco CSIRT**
- Co-founder Cysinfo Security Community**
- Author of Limon Sandbox**
- Winner of Volatility Plugin Contest 2016**
- Conferences - Black Hat, FIRST, 4SICS**
- Articles - eForensics, Hakin9, Hack Insight**

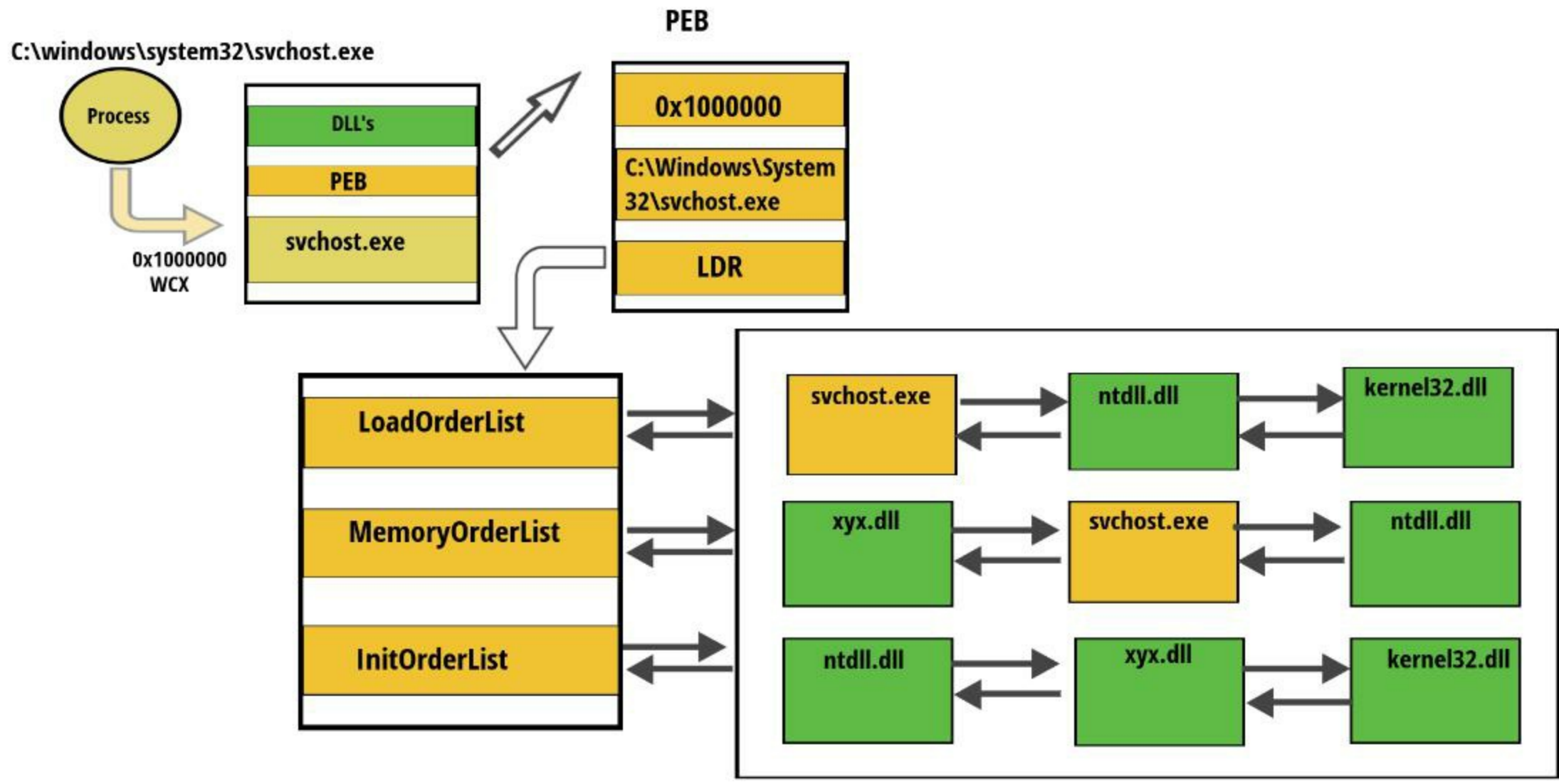


Process Memory Internals



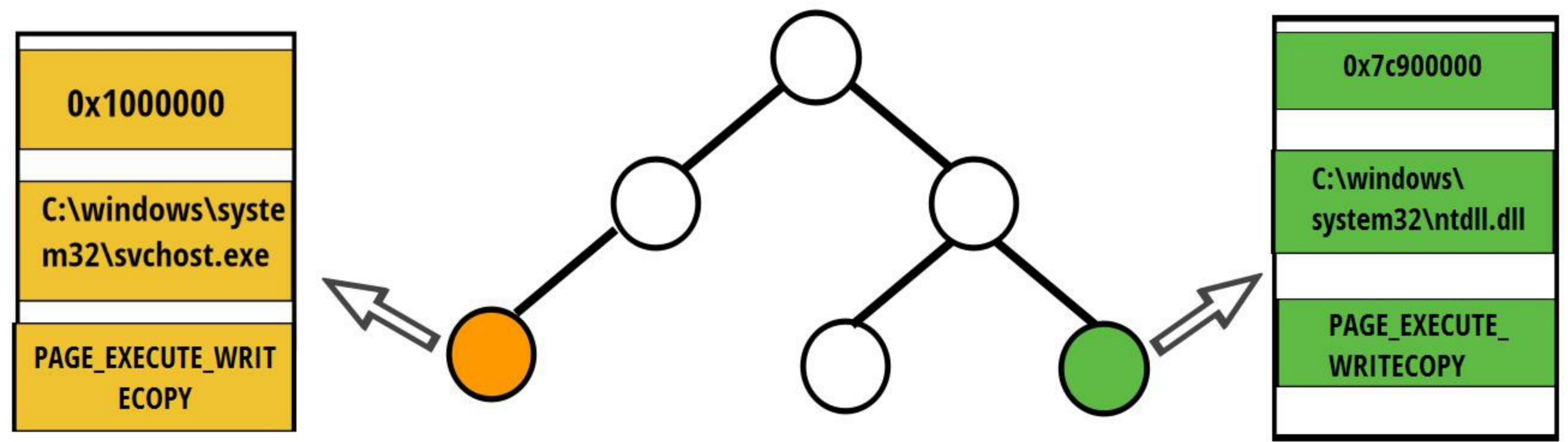


Process Memory

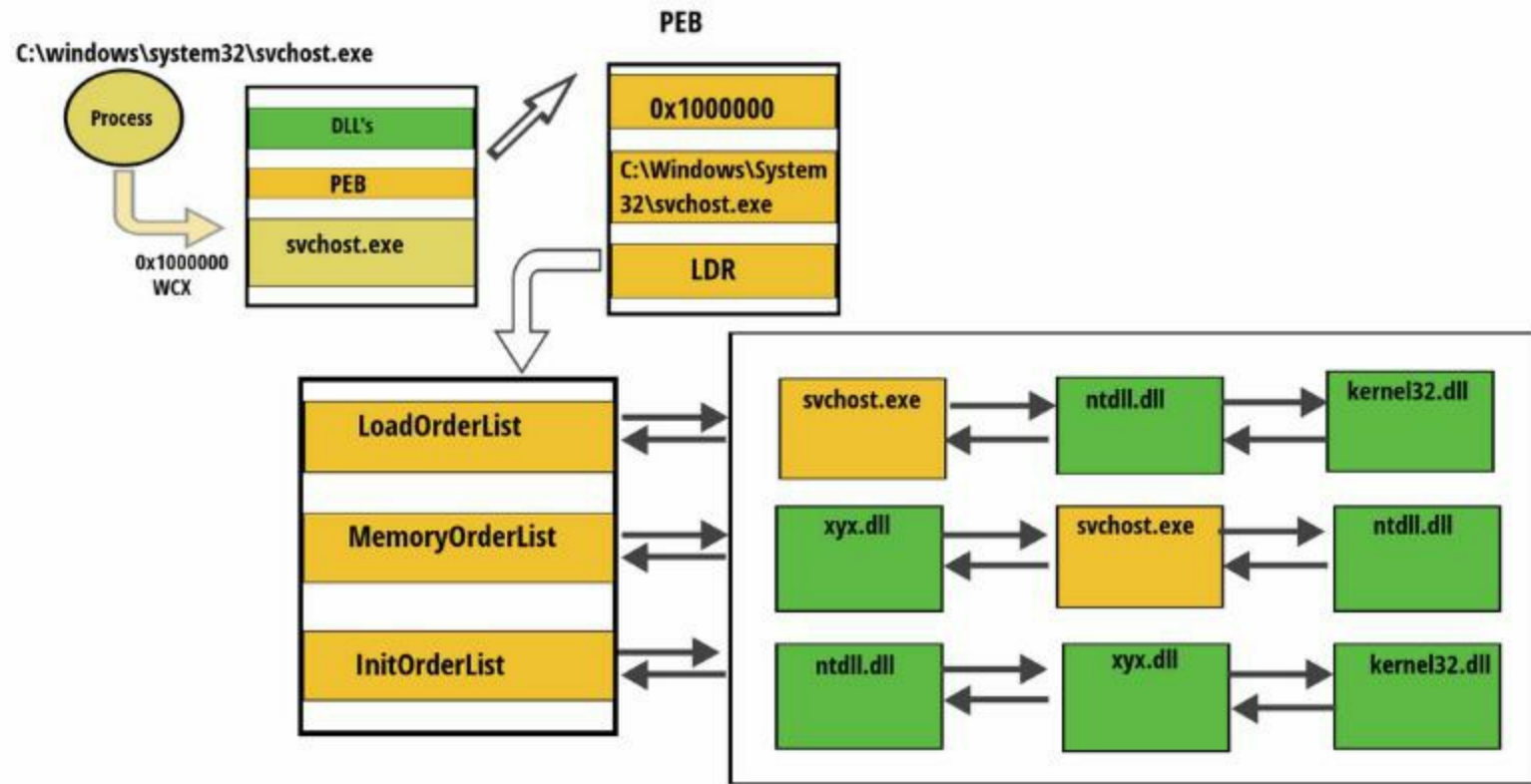


User Memory

Kernel Memory



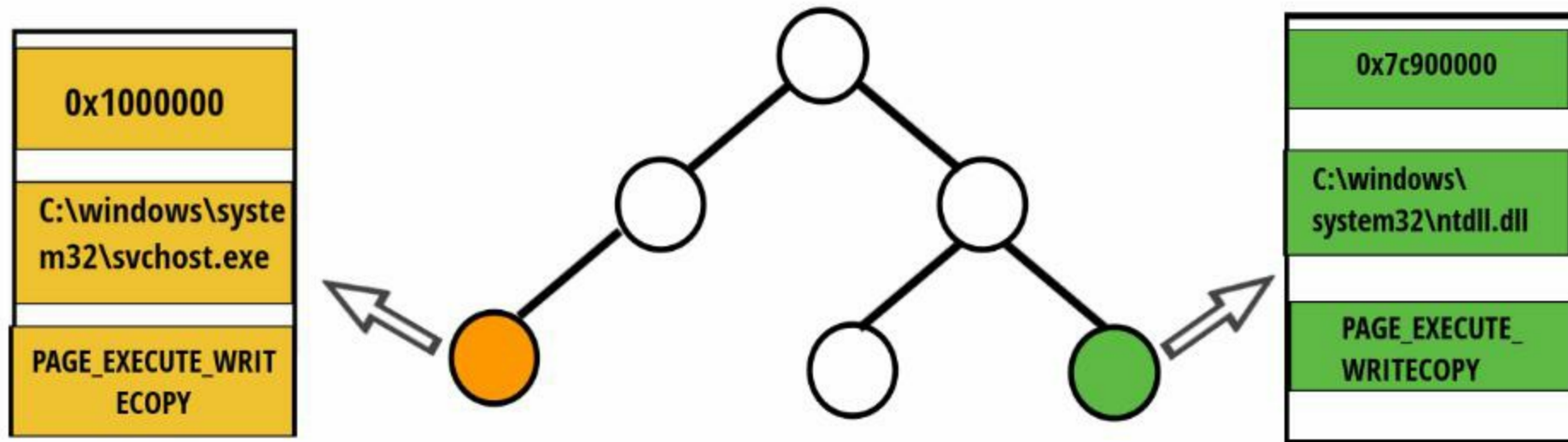
Dlllist Volatility plugin displays loaded modules by walking the InLoadOrderModuleList



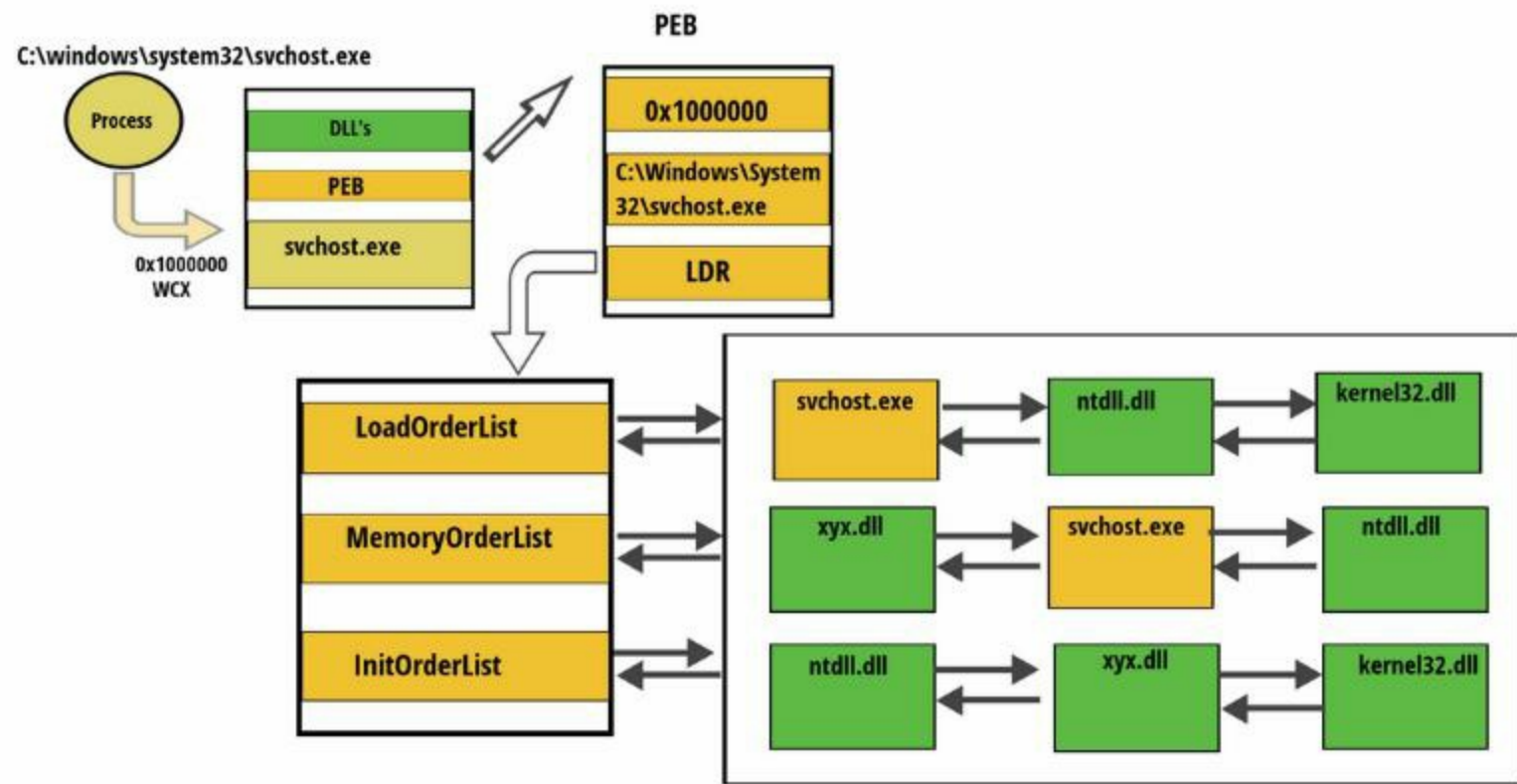
```

root@kratos:~/Volatility# python vol.py -f mem_image.vmem dlllist -p 884
Volatility Foundation Volatility Framework 2.5
*****
svchost.exe pid:      884
Command line : C:\WINDOWS\system32\svchost -k DcomLaunch
Service Pack 3
    
```

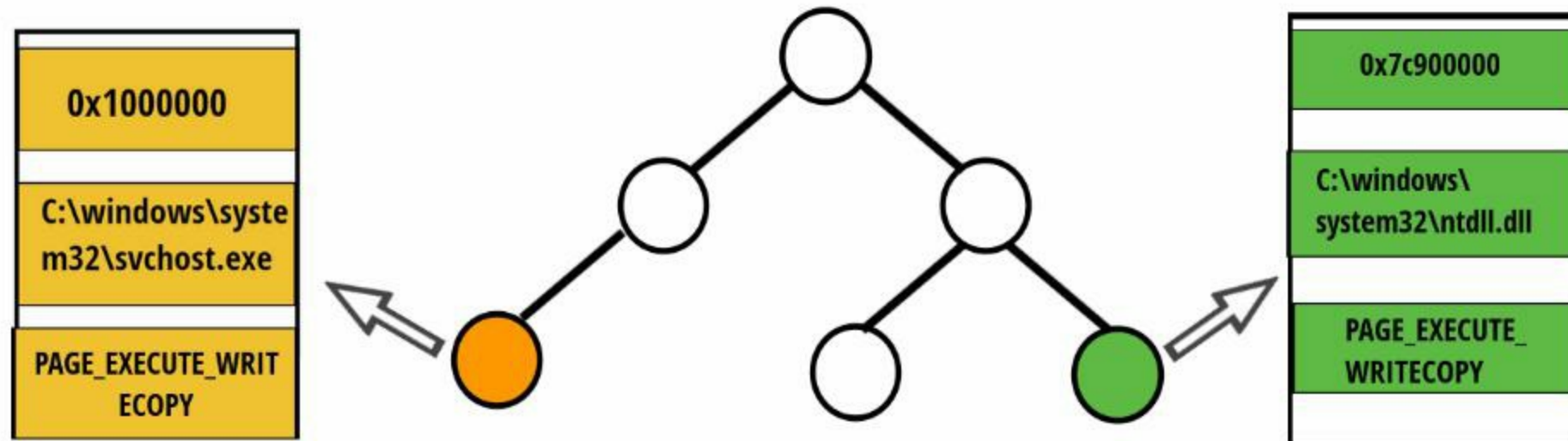
Base	Size	LoadCount	Path
0x01000000	0x6000	0xffff	C:\WINDOWS\system32\svchost.exe
0x7c900000	0xaf000	0xffff	C:\WINDOWS\system32\ntdll.dll
0x7c800000	0xf6000	0xffff	C:\WINDOWS\system32\kernel32.dll
0x77dd0000	0x9b000	0xffff	C:\WINDOWS\system32\ADVAPI32.dll
0x77e70000	0x92000	0xffff	C:\WINDOWS\system32\RPCRT4.dll
0x77fe0000	0x11000	0xffff	C:\WINDOWS\system32\Secur32.dll
0x5cb70000	0x26000	0x1	C:\WINDOWS\system32\ShimEng.dll
0x6f880000	0x1ca000	0x1	C:\WINDOWS\AppPatch\AcGeneral.DLL
0x7e410000	0x91000	0x39	C:\WINDOWS\system32\USER32.dll
0x77f10000	0x49000	0x28	C:\WINDOWS\system32\GDI32.dll
0x76b40000	0x2d000	0x2	C:\WINDOWS\system32\WINMM.dll



ldrmodules volatility plugin compares the PEB list with data in the VAD

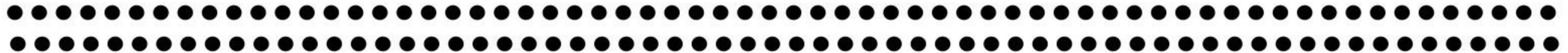


```
root@kratos:~/Volatility# python vol.py -f mem_image.vmem ldrmodules -p 884
Volatility Foundation Volatility Framework 2.5
Pid      Process                Base                InLoad  InInit  InMem  MappedPath
-----
884      svchost.exe            0x01000000         True    False   True   \WINDOWS\system32\svchost.exe
884      svchost.exe            0x7c900000         True    True    True   \WINDOWS\system32\ntdll.dll
884      svchost.exe            0x74f70000         True    True    True   \WINDOWS\system32\icaapi.dll
884      svchost.exe            0x76f60000         True    True    True   \WINDOWS\system32\wldap32.dll
884      svchost.exe            0x77c00000         True    True    True   \WINDOWS\system32\version.dll
884      svchost.exe            0x5ad70000         True    True    True   \WINDOWS\system32\uxtheme.dll
884      svchost.exe            0x76c90000         True    True    True   \WINDOWS\system32\imagehlp.dll
884      svchost.exe            0x76bc0000         True    True    True   \WINDOWS\system32\regapi.dll
884      svchost.exe            0x77dd0000         True    True    True   \WINDOWS\system32\advapi32.dll
884      svchost.exe            0x77a80000         True    True    True   \WINDOWS\system32\crypt32.dll
884      svchost.exe            0x77be0000         True    True    True   \WINDOWS\system32\msacm32.dll
884      svchost.exe            0x68000000         True    True    True   \WINDOWS\system32\rsaenh.dll
884      svchost.exe            0x76e10000         True    True    True   \WINDOWS\system32\adsldpc.dll
```





Introduction to Hollow Process Injection





Code Injection



Injects code into the legitimate process



Forces legitimate process to execute malicious code



Performs malicious actions within the context of legitimate process



Hollow Process Injection

Code Injection technique which replaces the executable section of the running process with malicious executable



Malware Creates a legitimate process in the suspended state



Process executable section is freed, reallocated and copied with malicious executable



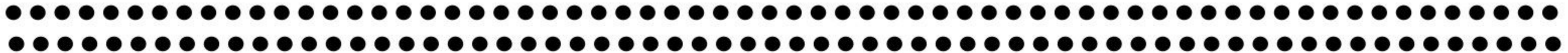
Suspended thread's start address is pointed to the malicious executable's address of entry point and thread is resumed

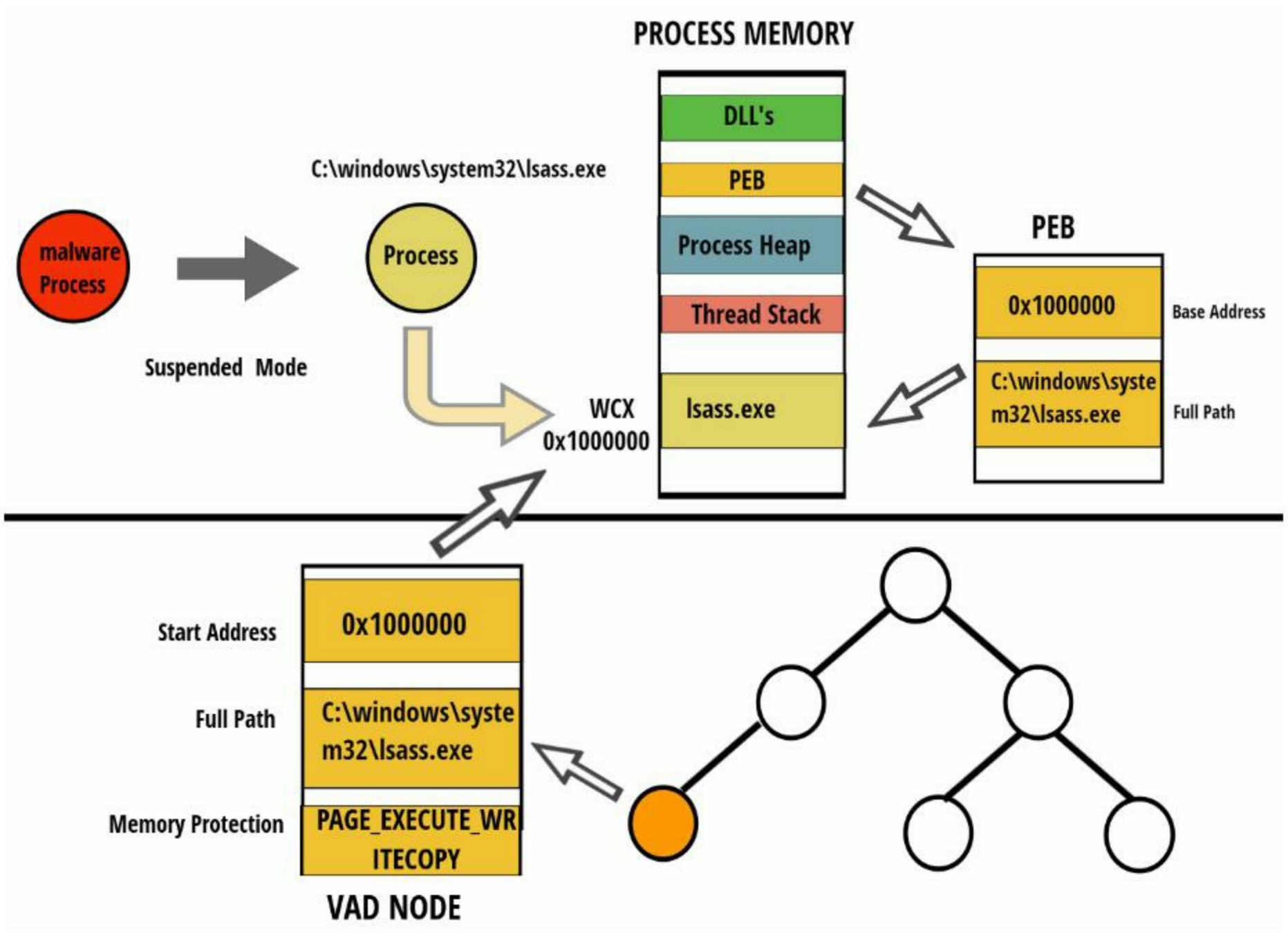


Used to disguise malware process as legitimate process

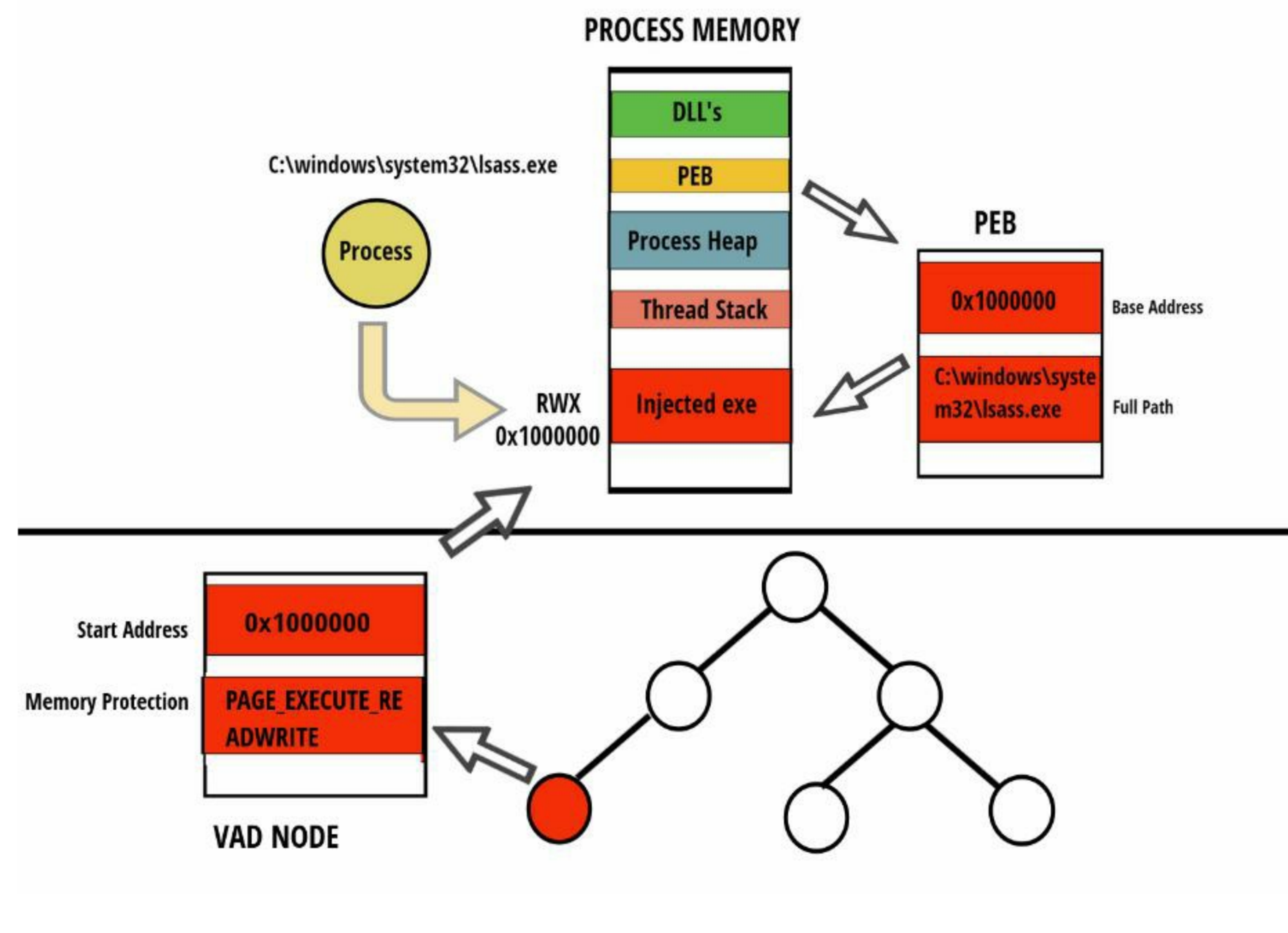


Hollow Process Injection - Stuxnet





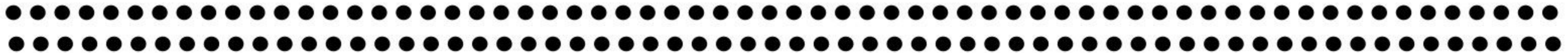
BEFORE HOLLOWING



AFTER HOLLOWING



***Detecting Hollow Process Injection
Using Memory Forensics - Stuxnet***



lsass.exe process (pid 868 and pid 1928) was not started by winlogon.exe or wininit.exe but these processes were started by services.exe (pid 668)

```
root@kratos:~/Volatility# python vol.py -f stuxnet.vmem pslist | grep -i lsass
Volatility Foundation Volatility Framework 2.5
0x81e70020 lsass.exe          680    624    19     342    0     0 2010-10-29
17:08:54 UTC+0000
0x81c498c8 lsass.exe          868    668     2     23     0     0 2011-06-03
04:26:55 UTC+0000
0x81c47c00 lsass.exe          1928   668     4     65     0     0 2011-06-03
04:26:55 UTC+0000
root@kratos:~/Volatility# python vol.py -f stuxnet.vmem pslist -p 668
Volatility Foundation Volatility Framework 2.5
Offset(V)  Name                PID  PPID  Thds  Hnds  Sess  Wow64  Start
-----
Exit
-----
0x82073020 services.exe ←      668 ← 624   21   431   0     0 2010-10-29
17:08:54 UTC+0000
```

```
root@kratos:~/Volatility# python vol.py -f stuxnet.vmem pslist -p 624
Volatility Foundation Volatility Framework 2.5
Offset(V)  Name                PID  PPID  Thds  Hnds  Sess  Wow64  Start
-----
Exit
-----
0x81da5650 winlogon.exe ←     624 ← 376   19   570   0     0 2010-10-29
17:08:54 UTC+0000
```



Comparing Dlllist and Ldrmodules plugin shows the discrepancy

```
root@kratos:~/Volatility# python vol.py -f stuxnet.vmem dlllist -p 868
Volatility Foundation Volatility Framework 2.5
*****
lsass.exe pid:      868
Command line : "C:\WINDOWS\system32\lsass.exe"
Service Pack 3

Base          Size    LoadCount Path
-----
0x01000000    0x6000    0xffff C:\WINDOWS\system32\lsass.exe ←
0x7c900000    0xaf000   0xffff C:\WINDOWS\system32\ntdll.dll
0x7c800000    0xf6000   0xffff C:\WINDOWS\system32\kernel32.dll
0x77dd0000    0x9b000   0xffff C:\WINDOWS\system32\ADVAPI32.dll
0x77e70000    0x92000   0xffff C:\WINDOWS\system32\RPCRT4.dll
0x77fe0000    0x11000   0xffff C:\WINDOWS\system32\Secur32.dll
0x7e410000    0x91000   0xffff C:\WINDOWS\system32\USER32.dll
0x77f10000    0x49000   0xffff C:\WINDOWS\system32\GDI32.dll
```

```
root@kratos:~/Volatility# python vol.py -f stuxnet.vmem ldrmodules -p 868
Volatility Foundation Volatility Framework 2.5
Pid      Process          Base          InLoad InInit InMem MappedPath
-----
868      lsass.exe        0x00080000    False  False False
868      lsass.exe        0x7c900000    True   True  True  \WINDOWS\system32\ntdll.d
ll
868      lsass.exe        0x77e70000    True   True  True  \WINDOWS\system32\rpcrt4.
dll
868      lsass.exe        0x7c800000    True   True  True  \WINDOWS\system32\kernel3
2.dll
868      lsass.exe        0x77fe0000    True   True  True  \WINDOWS\system32\secur32
.dll
868      lsass.exe        0x7e410000    True   True  True  \WINDOWS\system32\user32.
dll
868      lsass.exe        0x01000000    True   False True
868      lsass.exe        0x77f10000    True   True  True  \WINDOWS\system32\gdi32.d
ll
868      lsass.exe        0x77dd0000    True   True  True  \WINDOWS\system32\advapi3
2.dll
```



Malfind Volatility plugin shows suspicious memory protection (PAGE_EXECUTE_READWRITE) at address 0x1000000

```
root@kratos:~/Volatility# python vol.py -f stuxnet.vmem malfind -p 868
Volatility Foundation Volatility Framework 2.5
Process: lsass.exe Pid: 868 Address: 0x1000000
Vad Tag: Vad Protection: PAGE_EXECUTE_READWRITE
Flags: CommitCharge: 2, Protection: 6

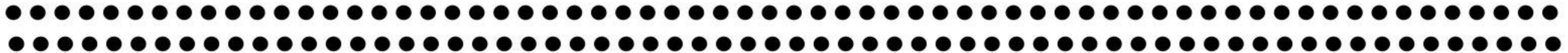
0x01000000  4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00  MZ.....
0x01000010  b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00  .....@.....
0x01000020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
0x01000030  00 00 00 00 00 00 00 00 00 00 00 00 00 d0 00 00  .....

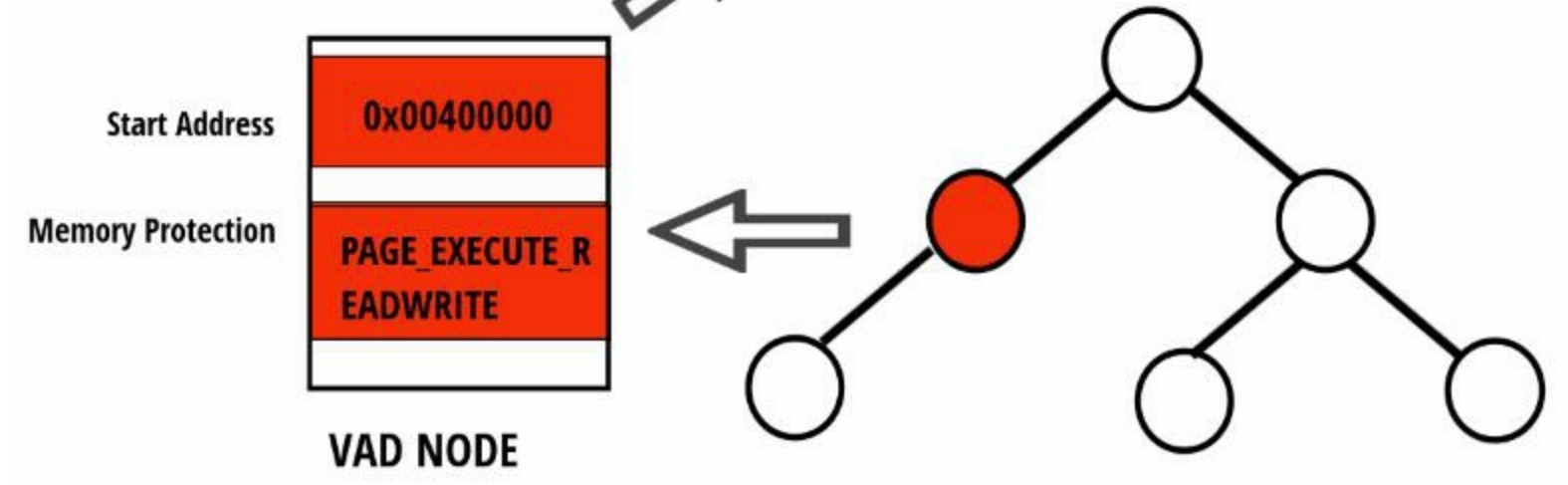
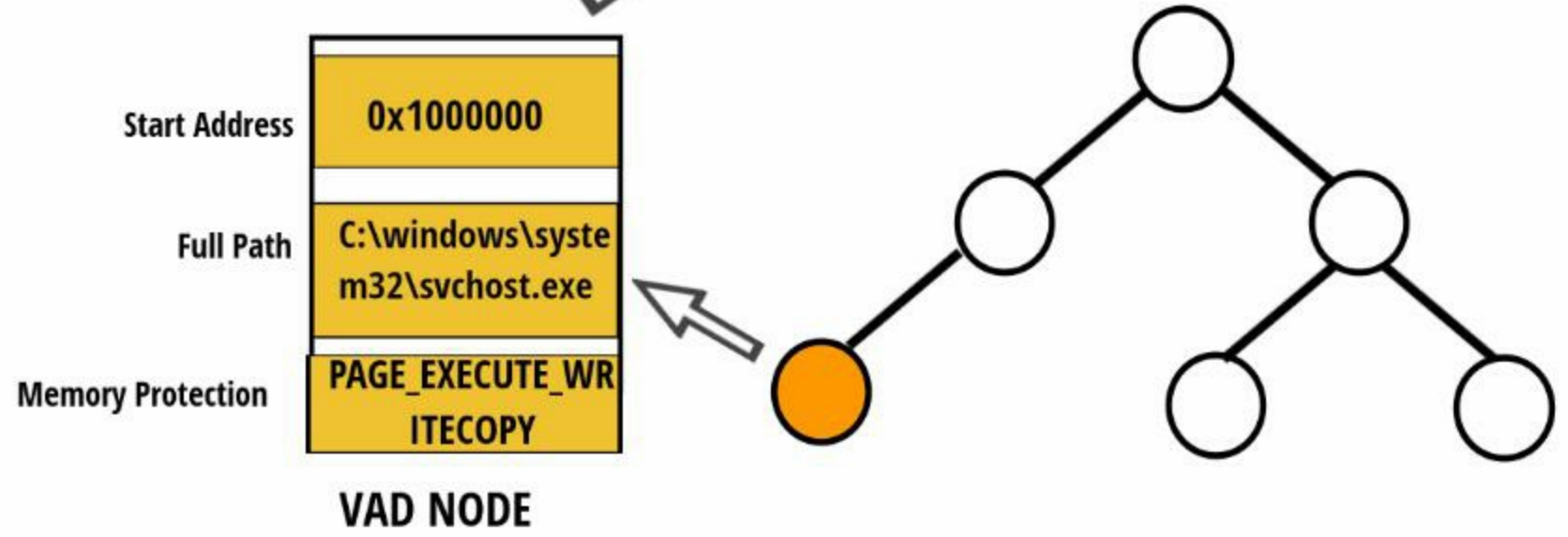
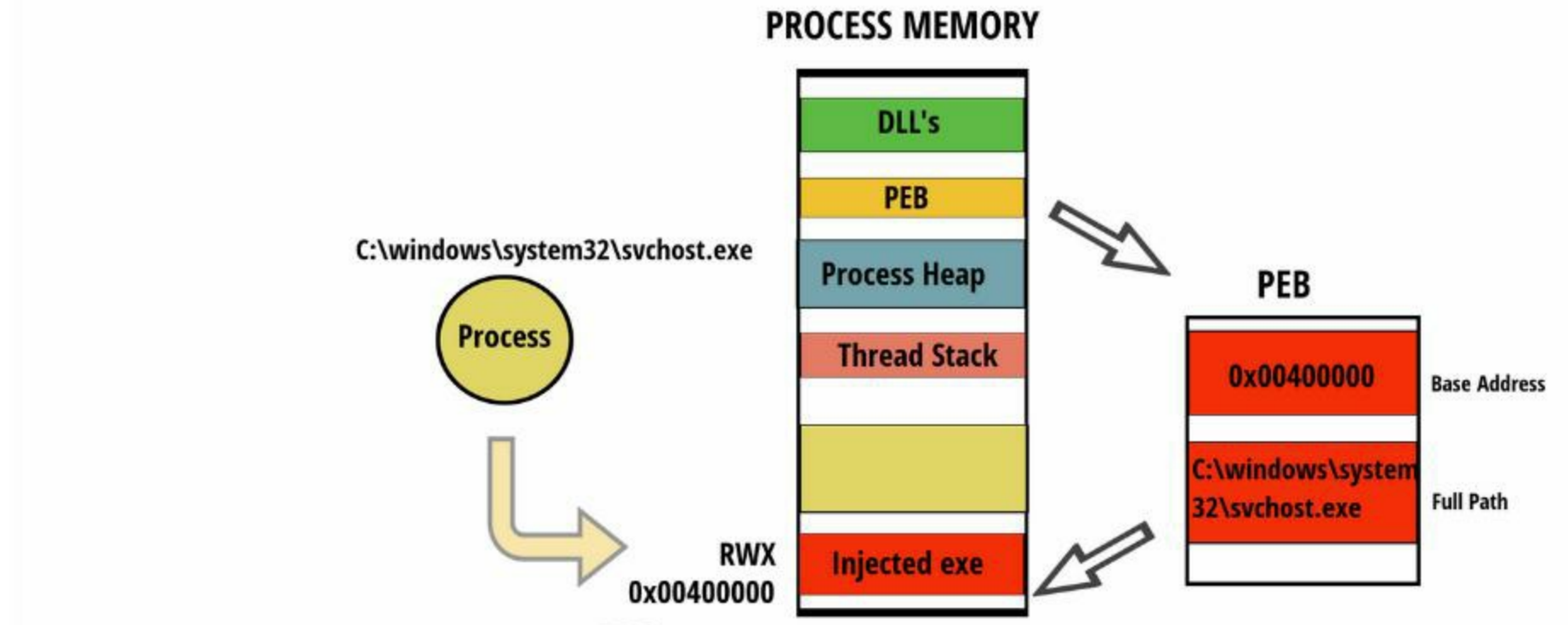
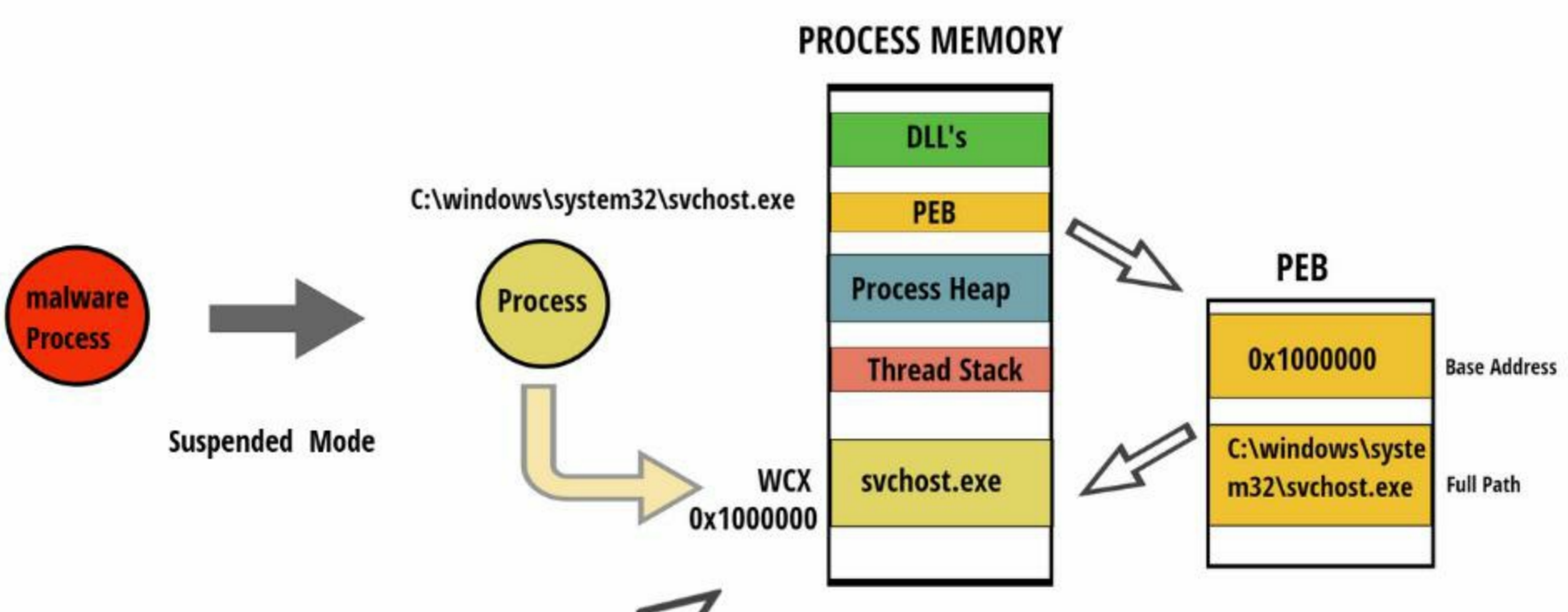
0x01000000 4d      DEC EBP
0x01000001 5a      POP EDX
0x01000002 90      NOP
```





Hollow Process Injection - Skeeyah



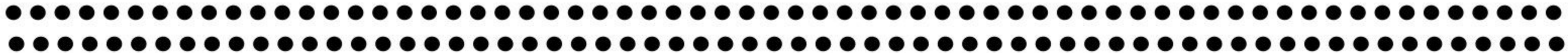


BEFORE HOLLOWING

AFTER HOLLOWING



***Detecting Hollow Process Injection
using Memory Forensics - Skeeyah***



svchost.exe process (pid 1824) was not started by services.exe

```
root@kratos:~/Volatility# python vol.py -f infected.vmem pslist | grep -i svchost
Volatility Foundation Volatility Framework 2.5
0x815cfaa0 svchost.exe      876    696    20    202    0    0 2016-05-10
06:47:25 UTC+0000
0x818c5a78 svchost.exe      960    696    9     227    0    0 2016-05-10
06:47:25 UTC+0000
0x8181e558 svchost.exe     1044   696    68    1227   0    0 2016-05-10
06:47:25 UTC+0000
0x818c7230 svchost.exe     1104   696    5     59     0    0 2016-05-10
06:47:25 UTC+0000
0x81743da0 svchost.exe     1144   696    15    210    0    0 2016-05-10
06:47:25 UTC+0000
0x817ba390 svchost.exe     1824   1768    1     26     0    0 2016-05-12
14:43:43 UTC+0000
```

```
root@kratos:~/Volatility# python vol.py -f infected.vmem pslist -p 696
Volatility Foundation Volatility Framework 2.5
Offset(V)  Name                PID  PPID  Thds  Hnds  Sess  Wow64  Start
-----
Exit
-----
0x8186c980 services.exe ←      696 ← 652   16   264    0     0 2016-05-10
06:47:24 UTC+0000
root@kratos:~/Volatility# python vol.py -f infected.vmem pslist -p 1768
Volatility Foundation Volatility Framework 2.5
ERROR : volatility.debug : Cannot find PID 1768. If its terminated or unlinked, u
se psscan and then supply --offset=OFFSET ←
root@kratos:~/Volatility#
```

Comparing Dlllist and Ldrmodules plugin shows the discrepancy

```
root@kratos:~/Volatility# python vol.py -f infected.vmem dlllist -p 1824
Volatility Foundation Volatility Framework 2.5
*****
svchost.exe pid: 1824
Command line : "C:\WINDOWS\system32\svchost.exe"
Service Pack 3

Base          Size  LoadCount Path
-----
0x00400000    0x7000    0xffff C:\WINDOWS\system32\svchost.exe
0x7c900000    0xaf000    0xffff C:\WINDOWS\system32\ntdll.dll
0x7c800000    0xf6000    0xffff C:\WINDOWS\system32\kernel32.dll
0x7e410000    0x91000    0xffff C:\WINDOWS\system32\USER32.dll
0x77f10000    0x49000    0xffff C:\WINDOWS\system32\GDI32.dll
0x5cb70000    0x26000     0x1 C:\WINDOWS\system32\ShimEng.dll
0x6f880000    0x1ca000   0x1 C:\WINDOWS\AppPatch\AcGenral.DLL
0x77dd0000    0x9b000    0x18 C:\WINDOWS\system32\ADVAPI32.dll
0x77e70000    0x92000    0xa C:\WINDOWS\system32\RPCRT4.dll
0x77fe0000    0x11000    0x5 C:\WINDOWS\system32\Secur32.dll
0x76b40000    0x2d000    0x2 C:\WINDOWS\system32\WINMM.dll
0x774e0000    0x13d000   0x2 C:\WINDOWS\system32\ole32.dll
0x77c10000    0x58000    0x9 C:\WINDOWS\system32\msvcrt.dll
0x77120000    0x8b000    0x1 C:\WINDOWS\system32\OLEAUT32.dll
```

```
root@kratos:~/Volatility# python vol.py -f infected.vmem ldrmodules -p 1824
Volatility Foundation Volatility Framework 2.5
Pid      Process          Base          InLoad InInit InMem MappedPath
-----
1824     svchost.exe      0x7c900000    True   True   True  \WINDOWS\system32\ntdll.dll
1824     svchost.exe      0x7c800000    True   True   True  \WINDOWS\system32\kernel32.dll
1824     svchost.exe      0x773d0000    True   True   True  \WINDOWS\WinSxS\x86_Microsoft.W
-Controls_6595b64144ccf1df_6.0.2600.5512_x-ww_35d4ce83\comctl32.dll
1824     svchost.exe      0x77f60000    True   True   True  \WINDOWS\system32\shlwapi.dll
1824     svchost.exe      0x769c0000    True   True   True  \WINDOWS\system32\userenv.dll
1824     svchost.exe      0x77dd0000    True   True   True  \WINDOWS\system32\advapi32.dll
1824     svchost.exe      0x77be0000    True   True   True  \WINDOWS\system32\msacm32.dll
1824     svchost.exe      0x77c00000    True   True   True  \WINDOWS\system32\version.dll
1824     svchost.exe      0x76b40000    True   True   True  \WINDOWS\system32\winmm.dll
1824     svchost.exe      0x77e70000    True   True   True  \WINDOWS\system32\rpcrt4.dll
1824     svchost.exe      0x6f880000    True   True   True  \WINDOWS\AppPatch\AcGenral.dll
1824     svchost.exe      0x774e0000    True   True   True  \WINDOWS\system32\ole32.dll
1824     svchost.exe      0x7e410000    True   True   True  \WINDOWS\system32\user32.dll
1824     svchost.exe      0x77f10000    True   True   True  \WINDOWS\system32\gdi32.dll
1824     svchost.exe      0x77120000    True   True   True  \WINDOWS\system32\oleaut32.dll
1824     svchost.exe      0x5cb70000    True   True   True  \WINDOWS\system32\shimeng.dll
1824     svchost.exe      0x76390000    True   True   True  \WINDOWS\system32\imm32.dll
1824     svchost.exe      0x7c9c0000    True   True   True  \WINDOWS\system32\shell32.dll
1824     svchost.exe      0x77c10000    True   True   True  \WINDOWS\system32\msvcrt.dll
1824     svchost.exe      0x5ad70000    True   True   True  \WINDOWS\system32\uxtheme.dll
1824     svchost.exe      0x5d090000    True   True   True  \WINDOWS\system32\comctl32.dll
1824     svchost.exe      0x77fe0000    True   True   True  \WINDOWS\system32\secur32.dll
root@kratos:~/Volatility#
```



Malfind Volatility plugin shows suspicious memory protection (PAGE_EXECUTE_READWRITE) at address 0x400000

```
root@kratos:~/Volatility# python vol.py -f infected.vmem malfind -p 1824
Volatility Foundation Volatility Framework 2.5
Process: svchost.exe Pid: 1824 Address: 0x400000 ←
Vad Tag: VadS Protection: PAGE_EXECUTE_READWRITE
Flags: CommitCharge: 7, MemCommit: 1, PrivateMemory: 1, Protection: 6

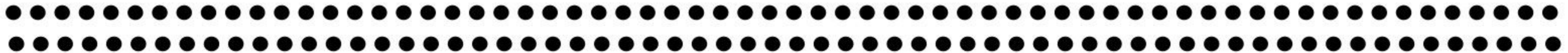
0x00400000  4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00  MZ.....
0x00400010  b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00  .....@.....
0x00400020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
0x00400030  00 00 00 00 00 00 00 00 00 00 00 00 00 e0 00 00 00  .....

0x00400000  4d          DEC EBP
0x00400001  5a          POP EDX
```





***Automating Process Hollow Detection
using HollowFind Volatility Plugin***



Detecting Stuxnet's process hollowing using hollowfind plugin

```
root@kratos:~/Volatility# python vol.py -f stuxnet.vmem hollowfind
Volatility Foundation Volatility Framework 2.5
Hollowed Process Information:
Process: lsass.exe PID: 868
Parent Process: services.exe PPID: 668
Creation Time: 2011-06-03 04:26:55 UTC+0000
Process Base Name(PEB): lsass.exe
Command Line(PEB): "C:\WINDOWS\system32\lsass.exe"
Hollow Type: Invalid EXE Memory Protection and Process Path Discrepancy

VAD and PEB Comparison:
Base Address(VAD): 0x1000000
Process Path(VAD):
Vad Protection: PAGE_EXECUTE_READWRITE
Vad Tag: Vad

Base Address(PEB): 0x1000000
Process Path(PEB): C:\WINDOWS\system32\lsass.exe
Memory Protection: PAGE_EXECUTE_READWRITE
Memory Tag: Vad
```

```
Similar Processes:
lsass.exe(868) Parent:services.exe(668) Start:2011-06-03 04:26:55 UTC+0000
lsass.exe(680) Parent:winlogon.exe(624) Start:2010-10-29 17:08:54 UTC+0000
lsass.exe(1928) Parent:services.exe(668) Start:2011-06-03 04:26:55 UTC+0000

Suspicious Memory Regions:
0x80000(PE Found) Protection: PAGE_EXECUTE_READWRITE Tag: Vad
0x1000000(PE Found) Protection: PAGE_EXECUTE_READWRITE Tag: Vad
```

Detecting Skeeyah's process hollowing using hollowfind plugin

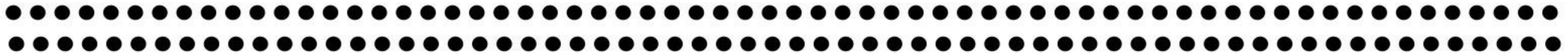
```
root@kratos:~/Volatility# python vol.py -f infected.vmem hollowfind
Volatility Foundation Volatility Framework 2.5
Hollowed Process Information:
  Process: svchost.exe PID: 1824 PPID: 1768
  Process Base Name(PEB): svchost.exe
  Hollow Type: No VAD Entry For Process Executable ←
VAD and PEB Comparison:
  Base Address(VAD): 0x0
  Process Path(VAD): NA ←
  Vad Protection: NA
  Vad Tag: NA
  Base Address(PEB): 0x400000 ←
  Process Path(PEB): C:\WINDOWS\system32\svchost.exe ←
  Memory Protection: PAGE_EXECUTE_READWRITE ←
  Memory Tag: VadS
0x00400000  4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00  MZ.....
0x00400010  b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00  .....@.....
0x00400020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
0x00400030  00 00 00 00 00 00 00 00 00 00 00 00 00 e0 00 00 00  .....
```

```
Similar Processes:
svchost.exe(1824) Parent:NA(1768) Start:2016-05-12 14:43:43 UTC+0000
svchost.exe(960) Parent:services.exe(696) Start:2016-05-10 06:47:25 UTC+0000
svchost.exe(1104) Parent:services.exe(696) Start:2016-05-10 06:47:25 UTC+0000
svchost.exe(1144) Parent:services.exe(696) Start:2016-05-10 06:47:25 UTC+0000
svchost.exe(876) Parent:services.exe(696) Start:2016-05-10 06:47:25 UTC+0000
svchost.exe(1044) Parent:services.exe(696) Start:2016-05-10 06:47:25 UTC+0000
```

```
Suspicious Memory Regions: ←
  0x400000(PE Found) Protection: PAGE_EXECUTE_READWRITE Tag: VadS
-----
```




***Evasive Hollow Process Injection -
Taidoor***



Malware allocates a memory in the remote process (svchost.exe) at address 0x00400000 but does not unmap (hollow out) the memory at the address 0x01000000

Assembly code snippets:

```
00401877 push    MEM_COMMIT or MEM_RESERVE
0040187C push    ecx
0040187D push    edx
0040187E push    eax
0040187F call    [esp+3DCh+addr_virtualallocex] ; VirtualAllocEx
00401883 mov     ebx, eax
00401885 test   ebx, ebx
00401887 jz     loc_40193C

0040188D mov     eax, [esp+3C8h+addr_writeprocessmemory]
00401891 test   eax, eax
00401893 jz     short loc_4018A4

00401895 mov     ecx, [ebp+IMAGE_NT_HEADERS.OptionalHeader.SizeOf
```

Registers:

- EAX 00000034
- EBX 7C839725 kernel32.dll:k
- ECX 00005000
- EDX 00400000 ppsx.exe:00400000
- ESI 00350000 debug019:00350
- EDI 00380000 debug022:00380
- EBP 003500F0 debug019:00350
- ESP 0012FA90 Stack[00000428

Stack view:

Address	Value	Comment
0012FA90	00000034	
0012FA94	00400000	ppsx.exe:00400000
0012FA98	00005000	
0012FA9C	00003000	
0012FAA0	00000040	



Malware writes the PE file to inject into the remote process (svchost.exe) at the allocated address 0x00400000

The screenshot displays a debugger interface with several panels:

- Assembly View:** Shows assembly instructions from address 00401895 to 004018AA. The instruction at 004018A2, `call eax ; WriteProcessMemory`, is highlighted with a red box. A green arrow points from this instruction to the memory view below.
- Registers:** Located on the right, it shows the state of various registers. `EBX 00400000` is highlighted in green, indicating the target address for the `WriteProcessMemory` call.
- Memory View (Hex View-1):** Located at the bottom left, it shows a hex dump of memory starting at address 00350000. A red box highlights the first few lines of the dump, which correspond to the PE file header.
- Stack View:** Located at the bottom right, it shows the stack frame. The address `0012FA94` contains the value `00400000`, which is highlighted in green and pointed to by a red arrow. This value matches the `EBX` register value.

```
00401895 mov     ecx, [ebp+IMAGE_NT_HEADERS.OptionalHeader.SizeOfHeaders]
00401898 mov     edx, [esp+3C8h+susp_proc_handle]
0040189C push    0
0040189E push    ecx           ; size of headers
0040189F push    esi           ; decrypted pe
004018A0 push    ebx           ; address where data will be written
004018A1 push    edx           ; suspended process handle
004018A2 call    eax           ; WriteProcessMemory
004018A4
004018A4 loc_4018A4:
004018A4 xor     edi, edi
004018A6 cmp     [ebp+IMAGE_NT_HEADERS.FileHeader.NumberOfSections], di
004018AA jbe    short loc_4018F9
```

100.00% (200,18889) (1253,5) 000018A2 004018A2: hollow_process_injection+7B2 (Synchronized with EIP)

Decimal	Hex	State
1064	428	Ready

Address	Value	Comment
0012FA90	00000034	
0012FA94	00400000	ppsx.exe:00400000
0012FA98	00350000	debug019:00350000
0012FA9C	00000400	
0012FAA0	00000000	
0012FAA4	0012FF84	Stack[00000428]:0012FF84
0012FAA8	EAD4ECE7	

```
00350000 4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 MZ.....*...
00350010 B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 +.....@.....
00350020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00350030 00 00 00 00 00 00 00 00 00 00 00 00 F0 00 00 00 .....=...
00350040 0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68 ...!..!-!+.L-!Th
00350050 69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F is*program*canno
00350060 74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20 t*be*run*in*DOS
```

Malware then overwrites the PEB.ImageBaseAddress of svchost.exe with the new address (0x00400000), this changes the base address of svchost.exe from 0x01000000 to 0x00400000 (this address contains the injected executable)

00401906 push 4
00401908 push eax
00401909 mov eax, [esi+CONTEXT._Ebx] ; PEB of remote process
0040190F add eax, 8 ; PEB+8 -> ImageBaseAddress
00401912 push eax
00401913 push ecx
00401914 call [esp+3DCh+addr_writeprocessmemory] ; modifies the ImageBaseAddress
00401918 mov edx, [ebp+IMAGE_NT_HEADERS.OptionalHeader.AddressOfEntryPoint] ; edx contains address of entry point
0040191B mov eax, [esp+3C8h+addr_setthreadcontext] ; modifies address of entry
0040191F add edx, ebx ; edx contains address of entry point
00401921 test eax, eax
00401923 mov [esi+CONTEXT._Eax], edx ; modifies address of entry
00401929 jz short loc_401933

EAX 7FFDE008
EBX 00400000 ppsx.exe:00400000
ECX 00000034
EDX 00000003
ESI 00380000 debug022:00380000
EDI 00000003
EBP 003500F0 debug019:003500F0
ESP 0012FA90 Stack[00000428]:0012FF84
EIP 00401914 hollow_process_injection+824 (Synchronized with EIP)

Hex View-1
00350124 00 00 40 00 00 10 00 00 00 02 00 00 04 00 00 00 ..@.....
00350134 00 00 00 00 04 00 00 00 00 00 00 00 00 50 00 00P..
00350144 00 04 00 00 00 00 00 00 02 00 00 00 00 00 10 00
00350154 00 10 00 00 00 00 10 00 00 10 00 00 00 00 00 00
00350164 10 00 00 00 00 00 00 00 00 00 00 00 A8 32 00 00;2..
00350174 8C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 i.....

Stack view
0012FA90 00000034
0012FA94 7FFDE008
0012FA98 00350124 debug019:00350124
0012FA9C 00000004
0012FAA0 00000000
0012FAA4 0012FF84 Stack[00000428]:0012FF84

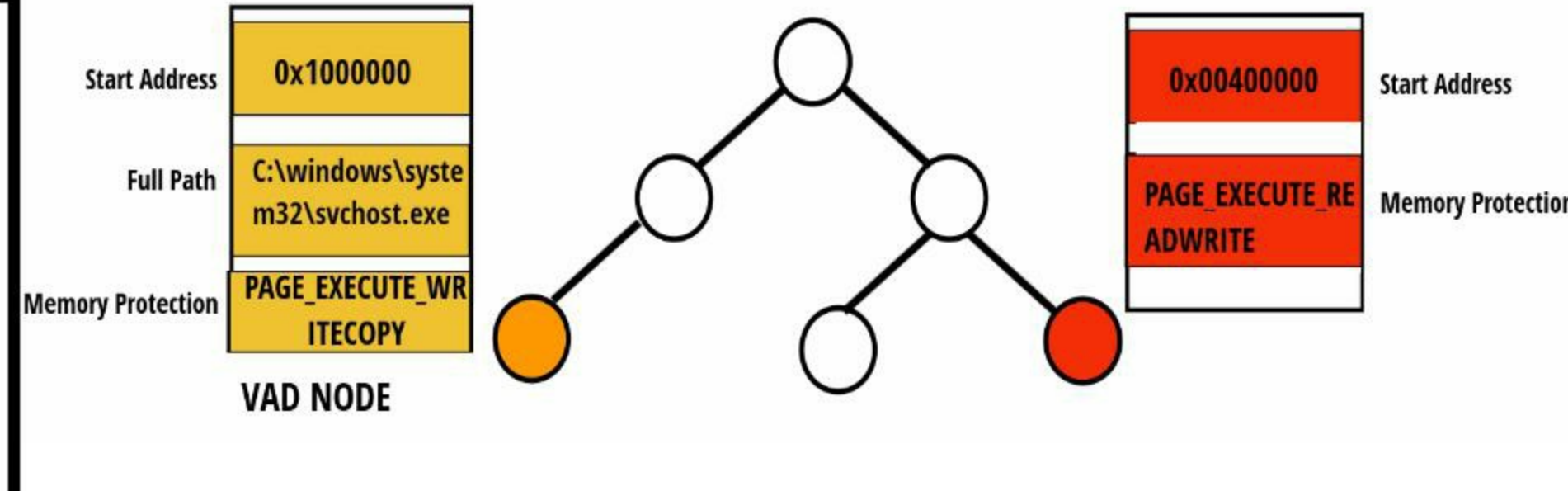
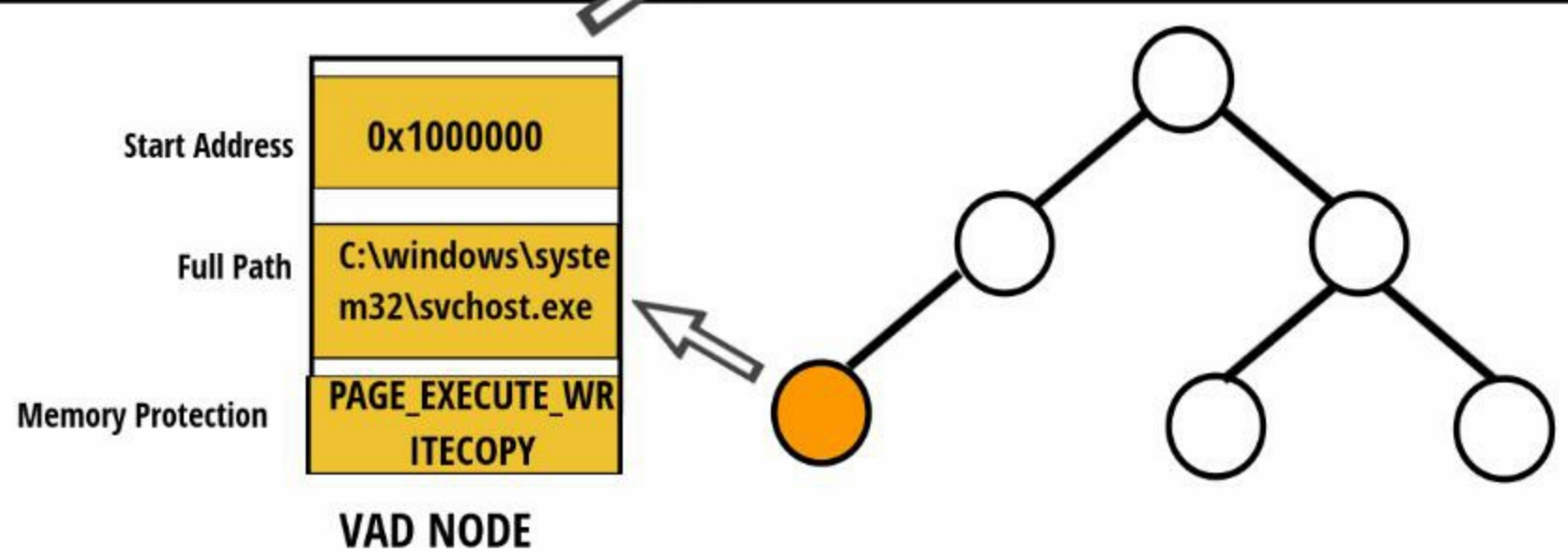
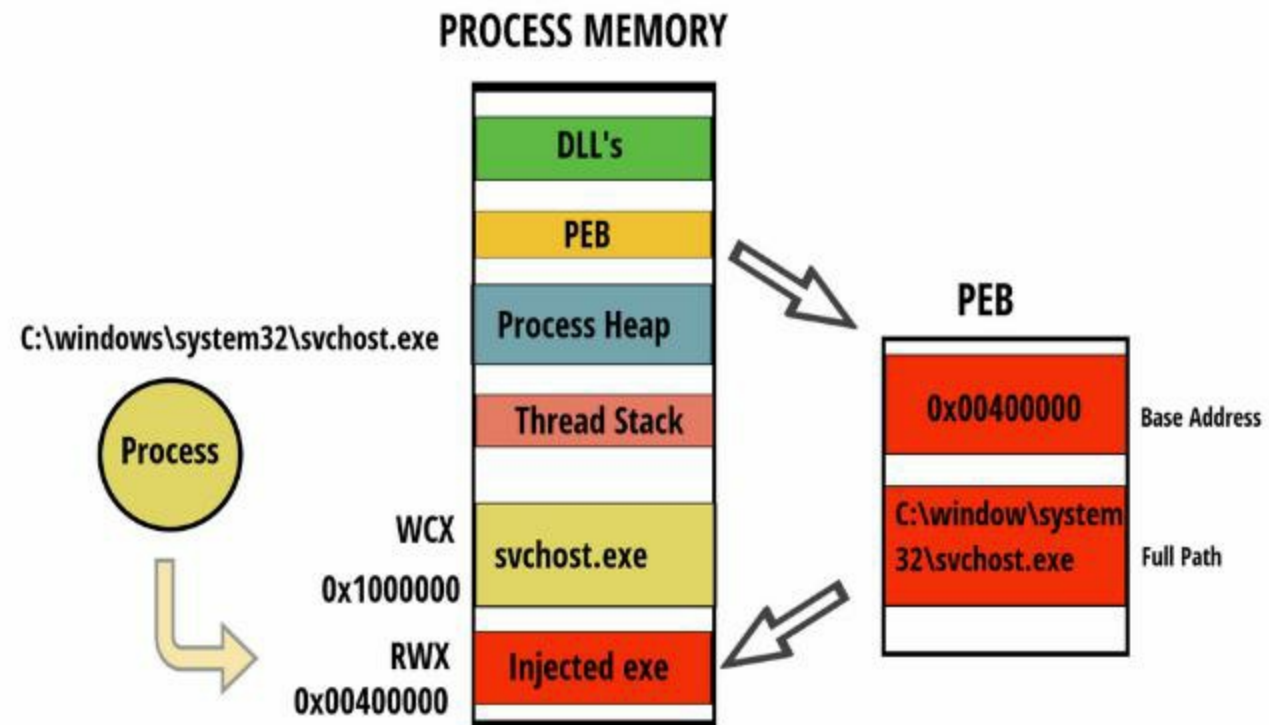
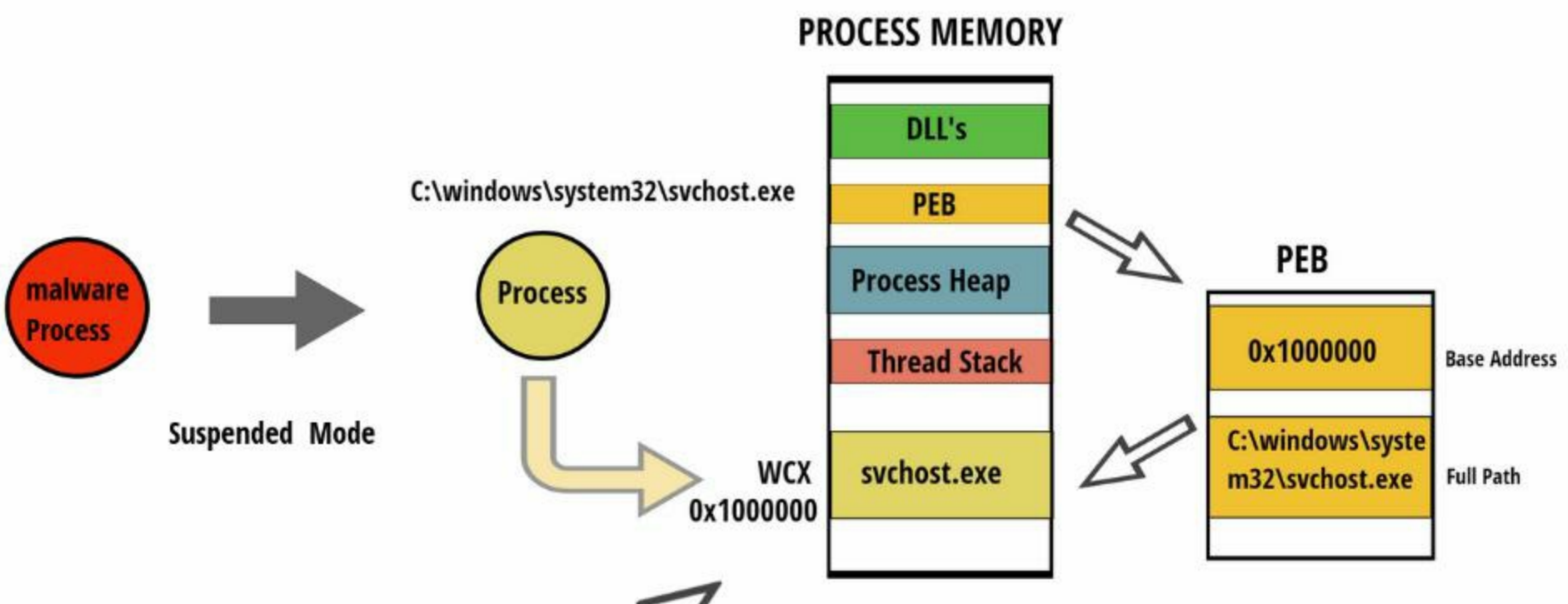


Malware then changes the start address of the suspended thread to the address of entry point of the injected executable by setting CONTEXT._Eax and using SetThreadContext api and then it resumes the thread

```
0040191B mov     eax, [esp+3C8h+addr_setthreadcontext]
0040191F add     edx, ebx           ; edx contains address of entry point
00401921 test    eax, eax
00401923 mov     [esi+CONTEXT._Eax], edx ; modifies address of entry point
00401929 jz     short loc_401933
```

```
0040192B mov     ecx, [esp+3C8h+var_3A8]
0040192F push   esi
00401930 push   ecx           ; hthread
00401931 call   eax           ; SetThreadContext
```

```
00401933
00401933 loc_401933:
00401933 mov     edx, [esp+3C8h+var_3A8]
00401937 push   edx           ; hthread
00401938 call   [esp+3CCh+addr_resumthread] ; ResumeThread
```

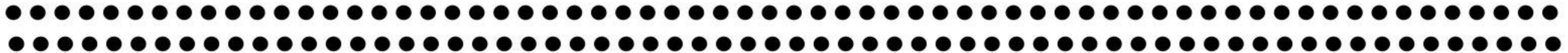


BEFORE HOLLOWING

AFTER HOLLOWING



***Demo1 - Taidoor's Process Hollowing
Confuses Analysis Tools***



Hollowfind Plugin Detects Taidoor's Hollow Process Injection

```
Hollowed Process Information:
Process: svchost.exe PID: 2020 PPID: 2012
Process Base Name(PEB): svchost.exe
Hollow Type: Process Base Address and Memory Protection Discrepancy

VAD and PEB Comparison:
Base Address(VAD): 0x1000000 ←
Process Path(VAD): \WINDOWS\system32\svchost.exe
Vad Protection: PAGE_EXECUTE_WRITECOPY ←
Vad Tag: Vad

Base Address(PEB): 0x400000 ←
Process Path(PEB): C:\WINDOWS\system32\svchost.exe
Memory Protection: PAGE_EXECUTE_READWRITE ←
Memory Tag: VadS

0x00400000 4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00 MZ.....
0x00400010 b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 .....@.....
0x00400020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0x00400030 00 00 00 00 00 00 00 00 00 00 00 00 f0 00 00 00 .....
```

Similar Processes:

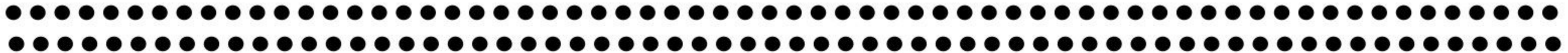
```
svchost.exe(2020) Parent:NA(2012) Start:2016-04-09 15:36:18 UTC+0000
svchost.exe(960) Parent:services.exe(572) Start:2016-04-03 18:44:53 UTC+0000
svchost.exe(1064) Parent:services.exe(572) Start:2016-04-03 18:44:55 UTC+0000
svchost.exe(832) Parent:services.exe(572) Start:2016-04-03 18:44:53 UTC+0000
svchost.exe(748) Parent:services.exe(572) Start:2016-04-03 18:44:53 UTC+0000
svchost.exe(892) Parent:services.exe(572) Start:2016-04-03 18:44:53 UTC+0000
```

Suspicious Memory Regions:

```
0x400000(PE Found) Protection: PAGE_EXECUTE_READWRITE Tag: VadS
```



***Evasive Hollow Process Injection -
Kuluoz***



Kuluoz creates svchost.exe process in the suspended mode which loads svchost.exe at address 0x120000

```
012F1E85 lea    eax, [ebp+ProcessInformation]
012F1E89 push   edx          ; lpProcessInformation
012F1E8A lea    eax, [ebp+StartupInfo]
012F1E8D push   eax          ; lpStartupInfo
012F1E8E push   0            ; lpCurrentDirectory
012F1E90 push   0            ; lpEnvironment
012F1E92 push   CREATE_SUSPENDED ; dwCreationFlags
012F1E94 push   0            ; bInheritHandles
012F1E96 push   0            ; lpThreadAttributes
012F1E98 push   0            ; lpProcessAttributes
012F1E9A push   offset CommandLine ; "svchost.exe"
012F1E9F push   0            ; lpApplicationName
012F1EA1 call   CreateProcessA ←
012F1EA7 mov    ecx, [ebp+ProcessInformation.hProcess]
012F1EAD mov    [ebp+sus_proc_handle], ecx
012F1EB3 mov    edx, [ebp+ProcessInformation.hThread]
012F1EB9 mov    [ebp+var_9C], edx
012F1EBF push   0
```


Code injected into the remote process (svchost.exe) at address 0x60000 with RWX memory protection

The screenshot displays Process Hacker's interface. On the left, a list of processes is shown, with **svchost.exe** highlighted. The right pane shows the memory view for **svchost.exe (2928)**. A red box highlights the memory region from **0x60000** to **0x71000**, which is mapped with **RWX** protection. The injected code is shown in both hex and ASCII views.

Base address	Type	Size	Protection	Use	Total WS	Private WS	Share...
0x10000	Private	128 kB	RW		8 kB	8 kB	
0x30000	Mapped	16 kB	R		16 kB		
0x40000	Mapped	4 kB	R		4 kB		
0x50000	Private	4 kB	RW		4 kB	4 kB	
0x60000	Mapped	68 kB	RWX		32 kB		
0x60000	Mapped: Commit	68 kB	RWX		32 kB		
0x120000	Image	32 kB	WCX	C:\Windows\System32\svchost.exe	20 kB		
0x17000					4 kB	4 kB	
0x77690					24 kB	12 kB	
0x778f0							
0x7ffb0					32 kB		
0x7ffde0					4 kB	4 kB	
0x7ffdf0					4 kB	4 kB	
0x7ffe0							

```
00000000 55 8b ec 81 ec ac 00 00 00 53 56 57 60 fc 33 d2 U.....SVW`.3.
00000010 64 8b 15 30 00 00 00 8b 52 0c 8b 52 14 8b 72 28 d..0....R..R..r(
00000020 6a 18 59 33 ff 33 c0 ac 3c 61 7c 02 2c 20 c1 cf j.Y3.3..<a|., ..
00000030 0d 03 f8 e2 f0 81 ff 5b bc 4a 6a 8b 5a 10 8b 12 .....[.Jj.2...
00000040 75 db 89 5d d8 61 8b 4d d8 8b 41 3c 8b 44 08 78 u..].a.M..A<.D.x
00000050 03 c1 8b 50 10 8b 70 1c 8b 78 24 89 55 e8 8b 50 ...P..p..x$.U..P
00000060 20 03 d1 03 f9 03 f1 33 db 89 55 a0 89 7d f4 c7 .....3..U..}..
00000070 85 5c ff ff ff 47 65 74 50 c7 85 60 ff ff ff 72 .\...GetP..`....r
00000080 6f 63 41 c7 85 64 ff ff ff 64 64 72 65 66 c7 85 ocA..d...ddref..
00000090 68 ff ff ff 73 73 88 9d 6a ff ff ff 89 5d fc 8b h...ss..j....}..
000000a0 45 fc 8b 04 82 03 c1 8d 95 5c ff ff ff c6 45 ef E.....\....E.
000000b0 01 89 45 f8 2b d0 c7 45 f0 0e 00 00 00 8b 45 f8 ..E.+..E.....E.
000000c0 8a 00 3a c3 74 0b 8b 7d f8 3a 04 3a 8b 7d f4 74 ...:t..}...:}.t
000000d0 03 88 5d ef ff 45 f8 ff 4d f0 75 e1 38 5d ef 75 ..].E..M.u.8}.u
000000e0 08 ff 45 fc 8b 55 a0 eb b6 8b 45 fc 0f b7 04 47 ..E..U....E....G
000000f0 0f b7 55 e8 2b c2 8b 74 86 04 8d 85 6c ff ff ff ..U.+..t...l...
00000100 50 03 f1 51 c7 85 6c ff ff ff 47 65 74 4d c7 85 P..Q..l...GetM..
00000110 70 ff ff ff 6f 64 75 6c c7 85 74 ff ff ff 65 48 p...odul..t...eH
00000120 61 6e c7 85 78 ff ff 64 6c 65 41 88 9d 7c ff an..x...dleA..|.
00000130 ff ff ff d6 89 85 54 ff ff ff 8d 45 90 50 ff 75 .....T....E.P.u
00000140 d8 c7 45 90 4c 6f 61 64 c7 45 94 4c 69 62 72 c7 ..E.Load.E.Libr.
00000150 45 98 61 72 79 41 88 5d 9c ff d6 89 85 58 ff ff E.aryA.]....X..
00000160 ff 8d 45 a4 50 ff 75 d8 c7 45 a4 45 78 69 74 c7 ..E.P.u..E.Exit.
00000170 45 a8 50 72 6f 63 c7 45 ac 65 73 73 00 ff d6 8d E.Proc.E.ess....
```

Malware copies the executable content of svchost.exe and It then unmaps the section in the svchost.exe where its executable is loaded (i.e 0x120000)

```
012F21A0 loc_12F21A0:
012F21A0 nop
012F21A1 mov     edx, [ebp+baseaddr_svchost]
012F21A7 push   edx
012F21A8 mov     eax, [ebp+sus_proc_handle]
012F21AF push   eax
012F21AF call   [ebp+Ntunmapviewofsection] ; NtUnmapViewOfSection
012F21B2 mov     [ebp+var_C4], eax
012F21B8 mov     ecx, [ebp+size_of_image]
012F21BB mov     [ebp+sect_size], ecx
```

Stack view

0018FC70	00000060
0018FC74	00120000
0018FC78	7FFDE008 TIB[
0018FC7C	00000005
0018FC80	00008000
0018FC84	00000000
0018FC88	00000000
0018FC8C	00000060
0018FC90	0000005C
0018FC94	00000B70
0018FC98	00000F94

General Statistics Performance Threads Token Modules Memory Environment Handles Job GPU Disk and Network Comment

Hide free regions

Strings... Refresh

Base address	Type	Size	Protection	Use	Total WS	Private WS	Sharea
> 0x10000	Private	128 kB	RW		8 kB	8 kB	
> 0x30000	Mapped	16 kB	R		16 kB		
> 0x40000	Mapped	4 kB	R		4 kB		
> 0x50000	Private	4 kB	RW		4 kB	4 kB	
> 0x60000	Mapped	68 kB	RWX		68 kB		
> 0x170000	Private	256 kB	RW	Stack (thread 3988)	4 kB	4 kB	
> 0x77690000	Image	1,288 kB	WCX	C:\Windows\System32\ntdll.dll	24 kB	12 kB	
> 0x778f0000	Image	4 kB	WCX	C:\Windows\System32\apisetschema.dll			
> 0x7ffb0000	Mapped	140 kB	R		32 kB		
> 0x7ffde000	Private	4 kB	RW	PEB	4 kB	4 kB	
> 0x7ffdf000	Private	4 kB	RW	TEB (thread 3988)	4 kB	4 kB	
> 0x7ffe0000	Private	64 kB	R	USER_SHARED_DATA			

Malware modifies the copied exe content, then maps the section (which contains patched svchost.exe) into the remote process (svchost.exe) at the same address 0x120000 with read, write, execute (RWX) protection. At this point first 7 bytes of address of entry point of svchost.exe is modified.

```
012F21D3 lea    eax, [ebp+sect_size]
012F21D9 push   eax
012F21DA push   0
012F21DC push   0
012F21DE push   0
012F21E0 lea    ecx, [ebp+view_base_addr]
012F21E6 push   ecx
012F21E7 mov    edx, [ebp+sus_proc_handle]
012F21ED push   edx
012F21EE mov    eax, [ebp+var_A0]
012F21F4 push   eax
012F21F9 call   [ebp+ntmapviewofsection] ; NtMapViewOfSection maps
012F21FB mov    [ebp+var_C4], eax
012F2201 mov    [ebp+EventAttributes.nLength], 0Ch
012F220B mov    [ebp+EventAttributes.lpSecurityDescriptor], 0
012F2215 mov    [ebp+EventAttributes.bInheritHandle], 1
012F221F mov    ecx, dword_12E10D8
012F2225 mov    dword ptr [ebp+Name], ecx
012F222B mov    edx, dword_12E10DC
```

Hex View-1

0018FDE8	00	00	12	00	00	00	5A	00	00	00	00	00	00	00	00	00	00	00	00Z.....
0018FDF8	00	00	00	00	00	80	00	00	44	00	00	00	00	00	00	00	00	00	00Ç..D.....
0018FE08	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00



Below screen shot shows the patched executable injected into the remote process (svchost.exe) at the address 0x120000 and once the patched executable is injected the suspended thread is resumed

The screenshot displays the Process Explorer interface. On the left, a list of processes is shown, with 'svchost.exe' highlighted. The main window shows the 'Memory' tab for a selected 'svchost.exe' process. A table lists memory regions, with a red box highlighting the region at address 0x120000, which is mapped and has RWX protection. Below this, a memory dump window shows the contents of the injected code, which is a valid PE header and the beginning of a program that displays an error message: 'is program cannot be run in DOS mode'.

Base address	Type	Size	Protection	Use	Total WS	Private
0x10000	Private	128 kB	RW		8 kB	
0x30000	Mapped	16 kB	R		16 kB	
0x40000	Mapped	4 kB	R		4 kB	
0x50000	Private	4 kB	RW		4 kB	
0x60000	Mapped	68 kB	RWX		68 kB	
0x120000	Mapped	32 kB	RWX		32 kB	
0x120000	Mapped: Commit	32 kB	RWX		32 kB	
0x170000	Private	256 kB	RW	Stack (thread 3988)	4 kB	
0x770000	Private				24 kB	1
0x770000	Private				32 kB	
0x770000	Private				4 kB	
0x770000	Private				4 kB	



Comparing address of entry point of the legitimate `svchost.exe` (on the left) and the patched `svchost.exe` (on the right) process shows the difference in the 7 bytes at the address of entry point, whereas all other bytes are same. These 7 bytes turn out to be 3 instructions which will redirect the control flow to the malicious code that was injected before (at address `0x60000`)

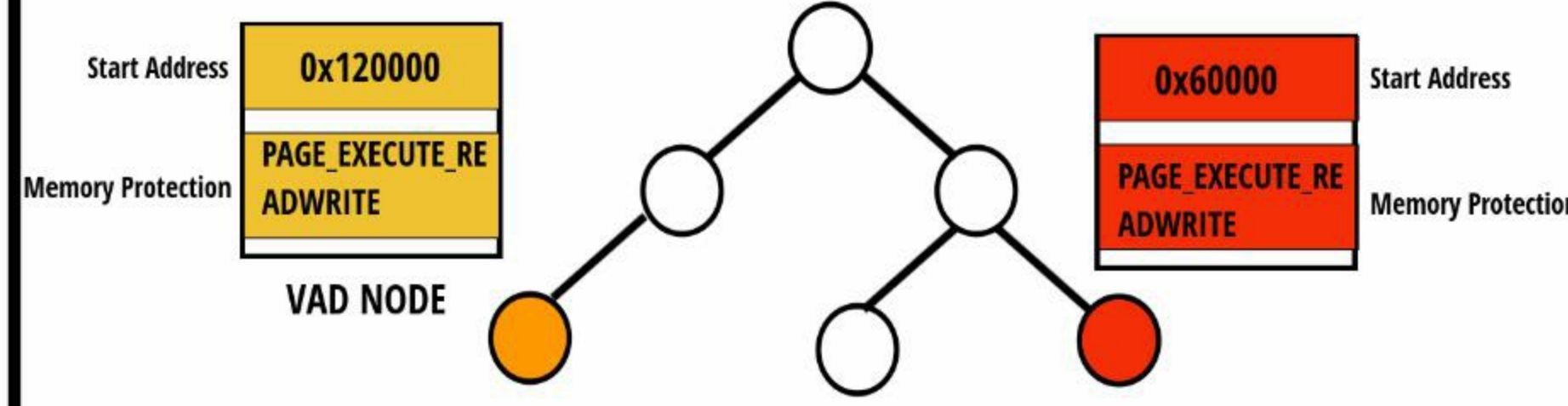
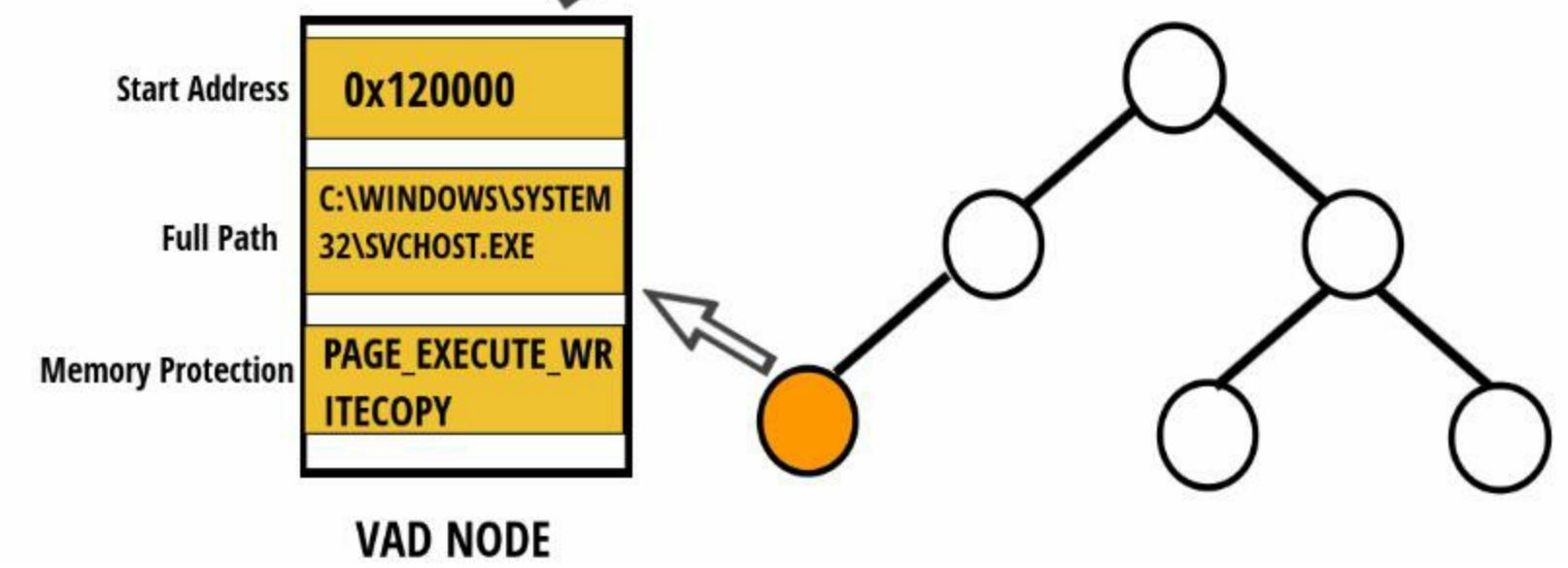
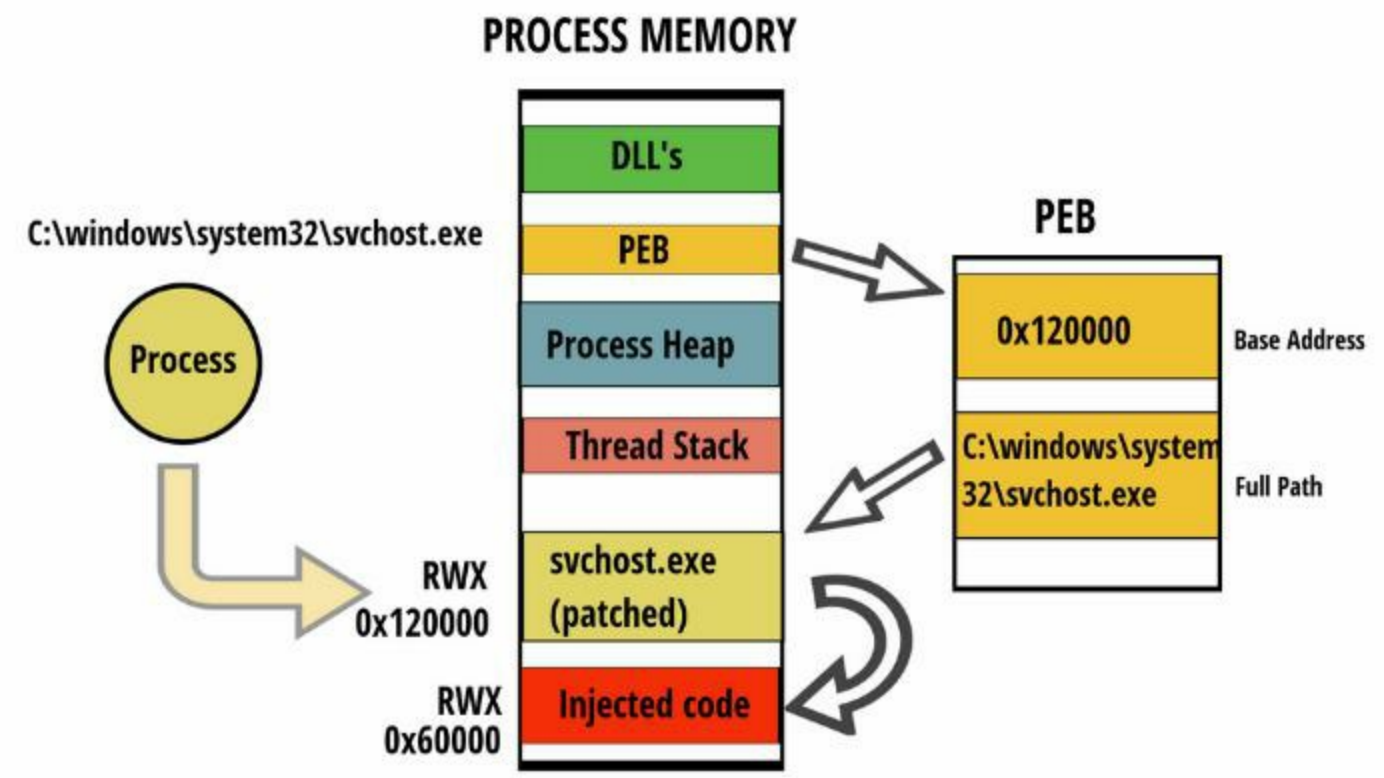
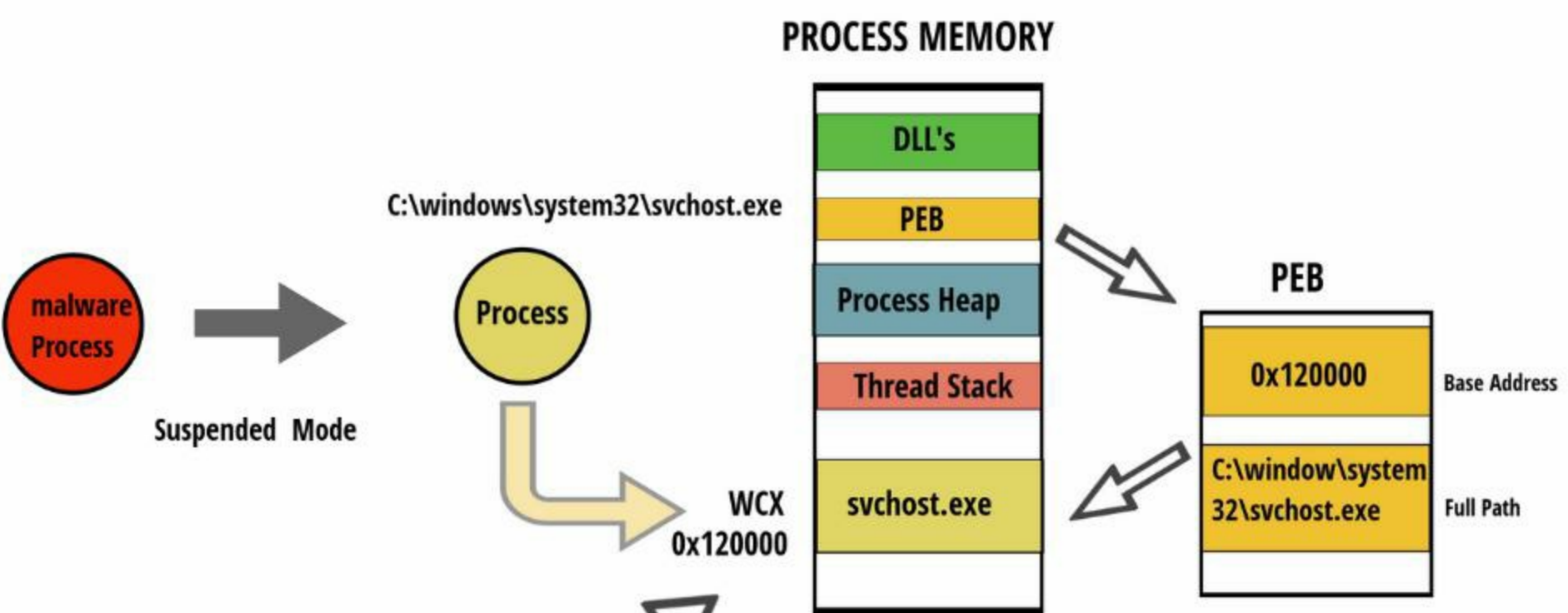
```
00001100 90 90 90 90 e8 d3 fc ff ff 6a 10 68 f8 21 12 00 .....j.h.!...
00001110 e8 75 fc ff ff 33 db 89 5d fc 64 a1 18 00 00 00 .u...3..].d....
00001120 8b 70 04 89 5d e4 bf 80 50 12 00 53 56 57 ff 15 .p..].P..SVW..
00001130 70 10 12 00 3b c3 0f 85 bc 0f 00 00 33 f6 46 a1 p...;.....3.F.
00001140 68 50 12 00 3b c6 0f 84 cb 0f 00 00 a1 68 50 12 hP..;.....hP.
00001150 00 85 c0 75 78 89 35 68 50 12 00 68 f0 21 12 00 ...ux.5hP..h.!...
00001160 68 e4 21 12 00 e8 71 ff ff ff 59 59 85 c0 0f 85 h.!...q...YY....
00001170 b0 0f 00 00 a1 68 50 12 00 3b c6 75 1b 68 e0 21 .....hP..;u.h.!
00001180 12 00 68 d8 21 12 00 e8 f8 fb ff ff 59 59 c7 05 ..h.!.....YY..
00001190 68 50 12 00 02 00 00 00 39 5d e4 75 08 53 57 ff hP.....9].u.SW.
000011a0 15 78 10 12 00 39 1d 58 55 12 00 0f 85 84 0f 00 .x...9.XU.....
000011b0 00 ff 35 bc 50 12 00 ff 35 c0 50 12 00 ff 35 b8 ..5.P...5.P...5.
000011c0 50 12 00 e8 64 02 00 00 e9 8a 0f 00 00 89 35 a4 P...d.....5.
000011d0 55 12 00 eb 9f 90 90 90 00 00 00 00 39 1d 12 00 U.....9...
000011e0 00 00 00 00 00 00 00 00 19 22 12 00 71 1d 12 00 .....".q...
000011f0 00 00 00 00 90 90 90 90 fe ff ff ff 00 00 00 00 .....
00001200 d0 ff ff ff 00 00 00 00 fe ff ff ff 6e 31 12 00 .....n1..
00001210 82 31 12 00 90 90 90 90 b8 4d 5a 00 00 66 39 .l.....MZ..f9
00001220 05 00 00 12 00 0f 85 8f 00 00 00 a1 3c 00 12 00 .....<...
00001230 8d 80 00 00 12 00 81 38 50 45 00 00 75 7c 0f b7 .....8PE..u|..
00001240 48 18 81 f9 0b 01 00 00 0f 85 70 0f 00 00 83 78 H.....p....x
00001250 74 0e 76 66 33 c9 39 88 e8 00 00 00 0f 95 c1 8b t.vf3.9.....
00001260 c1 6a 01 a3 48 50 12 00 e8 ce 01 00 00 50 ff 15 .j..HP.....P..
```

```
00002100 90 90 90 90 90 68 00 00 06 00 c3 68 f8 21 12 00 .....h.....h.!...
00002110 e8 75 fc ff ff 11 33 db 89 5d fc 64 a1 18 00 00 00 .u...3..].d....
00002120 8b 70 04 89 5d e4 bf 80 50 12 00 53 56 57 ff 15 .p..].P..SVW..
00002130 70 10 12 00 3b c3 0f 85 bc 0f 00 00 33 f6 46 a1 p...;.....3.F.
00002140 68 50 12 00 3b c6 0f 84 cb 0f 00 00 a1 68 50 12 hP..;.....hP.
00002150 00 85 c0 75 78 89 35 68 50 12 00 68 f0 21 12 00 ...ux.5hP..h.!...
00002160 68 e4 21 12 00 e8 71 ff ff ff 59 59 85 c0 0f 85 h.!...q...YY....
00002170 b0 0f 00 00 a1 68 50 12 00 3b c6 75 1b 68 e0 21 .....hP..;u.h.!
00002180 12 00 68 d8 21 12 00 e8 f8 fb ff ff 59 59 c7 05 ..h.!.....YY..
00002190 68 50 12 00 02 00 00 00 39 5d e4 75 08 53 57 ff hP.....9].u.SW.
000021a0 15 78 10 12 00 39 1d 58 55 12 00 0f 85 84 0f 00 .x...9.XU.....
000021b0 00 ff 35 bc 50 12 00 ff 35 c0 50 12 00 ff 35 b8 ..5.P...5.P...5.
000021c0 50 12 00 e8 64 02 00 00 e9 8a 0f 00 00 89 35 a4 P...d.....5.
000021d0 55 12 00 eb 9f 90 90 90 00 00 00 00 39 1d 12 00 U.....9...
000021e0 00 00 00 00 00 00 00 00 19 22 12 00 71 1d 12 00 .....".q...
000021f0 00 00 00 00 90 90 90 90 fe ff ff ff 00 00 00 00 .....
00002200 d0 ff ff ff 00 00 00 00 fe ff ff ff 6e 31 12 00 .....n1..
00002210 82 31 12 00 90 90 90 90 b8 4d 5a 00 00 66 39 .l.....MZ..f9
00002220 05 00 00 12 00 0f 85 8f 00 00 00 a1 3c 00 12 00 .....<...
00002230 8d 80 00 00 12 00 81 38 50 45 00 00 75 7c 0f b7 .....8PE..u|..
00002240 48 18 81 f9 0b 01 00 00 0f 85 70 0f 00 00 83 78 H.....p....x
00002250 74 0e 76 66 33 c9 39 88 e8 00 00 00 0f 95 c1 8b t.vf3.9.....
00002260 c1 6a 01 a3 48 50 12 00 e8 ce 01 00 00 50 ff 15 .j..HP.....P..
```

```
root@localhost:~# rasm2 -d 906800000600c3
```

```
nop
push 0x60000
ret
```



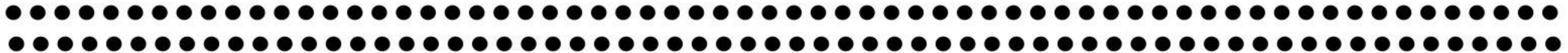


BEFORE HOLLOWING

AFTER HOLLOWING



***Demo2 - Kuluoz's Process Hollowing
Diverts Analysis***



Hollowfind plugin detects Kuluoz's Hollow Process Injection

Hollowed Process Information:

Process: svchost.exe PID: 3056 PPID: 3040
Process Base Name(PEB): svchost.exe
Hollow Type: Invalid EXE Memory Protection and Process Path Discrepancy

VAD and PEB Comparison:

Base Address(VAD): 0xa00000
Process Path(VAD):
Vad Protection: PAGE_EXECUTE_READWRITE
Vad Tag: Vad

Base Address(PEB): 0xa00000
Process Path(PEB): C:\Windows\system32\svchost.exe
Memory Protection: PAGE_EXECUTE_READWRITE
Memory Tag: Vad

Disassembly(Entry Point):

```
0x00a02104 90      NOP
0x00a02105 6800000600  PUSH DWORD 0x60000
0x00a0210a c3      RET
0x00a0210b 68f821a000  PUSH DWORD 0xa021f8
0x00a02110 e875fcffff  CALL 0xa01d8a
```

Similar Processes:

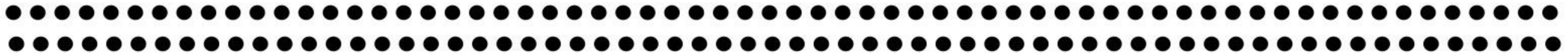
```
svchost.exe(3056) Parent:order.exe(3040) Start:2016-05-11 07:31:52 UTC+0000
svchost.exe(1152) Parent:services.exe(496) Start:2016-05-11 06:35:30 UTC+0000
svchost.exe(1068) Parent:services.exe(496) Start:2016-05-11 06:35:30 UTC+0000
svchost.exe(1328) Parent:services.exe(496) Start:2016-05-11 06:35:30 UTC+0000
svchost.exe(624) Parent:services.exe(496) Start:2016-05-11 06:35:29 UTC+0000
svchost.exe(712) Parent:services.exe(496) Start:2016-05-11 06:35:29 UTC+0000
svchost.exe(764) Parent:services.exe(496) Start:2016-05-11 06:35:29 UTC+0000
svchost.exe(876) Parent:services.exe(496) Start:2016-05-11 06:35:30 UTC+0000
svchost.exe(916) Parent:services.exe(496) Start:2016-05-11 06:35:30 UTC+0000
```

Suspicious Memory Regions:

```
0x60000(No PE/Possibly Code) Protection: PAGE_EXECUTE_READWRITE Tag: Vad
0x310000(No PE/Possibly Code) Protection: PAGE_EXECUTE_READWRITE Tag: VadS
0xa00000(PE Found) Protection: PAGE_EXECUTE_READWRITE Tag: Vad
```



***Evasive Hollow Process Injection -
Modifying Kuluoz to be Stealthy***





Steps to Make Kuluoz Stealthy

- **Instead of creating svchost.exe in the suspended mode, created explorer.exe in the suspended mode, the reason is because explorer.exe is normally started by userinit.exe and it terminates itself which means userinit.exe will not show up in the process listing (and will not show as parent for explorer.exe). So if malware starts explorer.exe, injects code and terminates itself, it can become hard to tell based on the parent child relationship**
- **Kuluoz injects code into the remote process using NtMapViewOfSection with read, write, execute(RWX) permission but if we can some how map that memory section containing malicious code with PAGE_EXECUTE_WRITECOPY protection we should be able to bypass the malfind plugin thereby hiding the memory region containing malicious code**

As per API documentation flag `PAGE_EXECUTE_WRITECOPY` is not supported by memory allocation API's like `VirtualAllocEx` but it turns out that `PAGE_EXECUTE_WRITECOPY` protection can be set by using the native api like `NtMapViewOfSection`, so Kuluoz code was modified to do that.



The image shows a screenshot of the Windows API documentation for memory protection constants. The left sidebar lists categories like 'Memory Management Functions', 'Memory Management Registry Keys', 'Memory Management Structures', 'Memory Protection Constants', and 'Memory Management Tracing Events'. The main content area shows a table with two entries. The entry for 'PAGE_EXECUTE_WRITECOPY' (0x80) is highlighted with a red box. The text for this entry states that it enables execute, read-only, or copy-on-write access and that it is not supported by VirtualAlloc or VirtualAllocEx. The entry for 'PAGE_EXECUTE_READWRITE' (0x40) is also visible above it.

PAGE_EXECUTE_READWRITE 0x40	Enables execute, read-only, or read/write access to the committed region of pages. Windows Server 2003 and Windows XP: This attribute is not supported by the CreateFileMapping function until Windows XP with SP2 and Windows Server 2003 with SP1.
PAGE_EXECUTE_WRITECOPY 0x80	Enables execute, read-only, or copy-on-write access to a mapped view of a file mapping object. An attempt to write to a committed copy-on-write page results in a private copy of the page being made for the process. The private page is marked as PAGE_EXECUTE_READWRITE , and the change is written to the new page. This flag is not supported by the VirtualAlloc or VirtualAllocEx functions. Windows Vista, Windows Server 2003, and Windows XP: This attribute is not supported by the CreateFileMapping function until Windows Vista with SP1 and Windows Server 2008.



Kuluoz malwares sample was modified to create explorer.exe in the suspended mode instead of svchost.exe. The explorer.exe was loaded at base address 0x570000 with the PAGE_EXECUTE_WRITECOPY(WCX)

```
00D81E83 lea    edx, [ebp+ProcessInformation]
00D81E89 push   edx          ; lpProcessInformation
00D81E8A lea    eax, [ebp+StartupInfo]
00D81E8D push   eax          ; lpStartupInfo
00D81E8E push   0            ; lpCurrentDirectory
00D81E90 push   0            ; lpEnvironment
00D81E92 push   CREATE_SUSPENDED ; dwCreationFlags
00D81E94 push   0            ; bInheritHandles
00D81E96 push   0            ; lpThreadAttributes
00D81E98 push   0            ; lpProcessAttributes
00D81E9A push   offset CommandLine ; "explorer.exe"
00D81E9F push   0            ; lpApplicationName
00D81EA1 call   CreateProcessA
00D81EA7 mov    ecx, [ebp+ProcessInformation.hProcess]
00D81EAD mov    [ebp+sus_proc_handle], ecx
00D81EB3 mov    edx, [ebp+ProcessInformation.hThread]
00D81EB9 mov    [ebp+var_9C], edx
00D81EBF push  0
00D81EC1 push  18h
00D81EC3 lea    [ebp+var_10], [ebp+var_9C]
00D81EC6 push  [ebp+var_10]
00D81EC7 push  [ebp+var_10]
00D81EC9 mov    [ebp+var_10], [ebp+var_9C]
00D81ECF push  [ebp+var_10]
```

Stack view

0020FD60	00000000
0020FD64	00D710CC
0020FD68	00000000
0020FD6C	00000000
0020FD70	00000000
0020FD74	00000004
0020FD78	00000000
0020FD7C	00000000
0020FD80	0020FF10 S
0020FD84	0020FD9C S
0020FD88	00000000
0020FD8C	00000005
0020FD90	00000000
0020FD94	00000000
0020FD98	00000000
0020FD9C	00000000
0020FDA0	00000000

UNKNOWN 0020FD64: (Synchr

Hex View-1

00D710CC	65 78 70 6C 6F 72 65 72 2E 65 78 65 00 6E 65 6F	explorer.exe neo
00D710DC	64 76 33 47 00 00 00 00 00 00 00 00 00 00 00 00	uv3G.....
00D710EC	00 00 00 00 55 8B EC 81 EC AC 00 00 00 53 56 57Ui8.8¼...SVW

kuluoz was allowed to inject code into remote process (explorer.exe) at address 0x60000 but instead of allowing the malware to map the section with RWX protection, it was modified to map the section with writecopy(WCX) protection by changing the constant value to 0x80.

```
00D81FC9 push 0
00D81FCB push 1
00D81FCD lea ecx, [ebp+sect_size]
00D81FD3 push ecx
00D81FD4 push 0
00D81FD6 push 0
00D81FD8 push 0
00D81FDA lea edx, [ebp+view_base_addr]
00D81FE0 push edx
00D81FE1 mov eax, [ebp+sus_proc_handle]
00D81FE7 push eax
00D81FE8 mov ecx, [ebp+handle_section]
00D81FEB push ecx
00D81FEC call [ebp+ntmapviewofsection] ; NtMapViewOfSection maps in remote p
00D81FF2 mov [ebp+var_C4], eax
00D81FF8 mov edx, [ebp+view_base_addr]
00D81FFE mov [ebp+rem_view_base_addr], edx
00D82001 push PAGE_EXECUTE_READWRITE ; flProtect
00D82003 Hex View-1
00D82008 0020FD84 80 00 00 00 08 30 FD 7F 05 00 00 00
00D8200D 0020FD94 00 00 00 00 00 00 00 00 60 00 00 00
```

```
0020FD64 00000060
0020FD68 0020FEF8 Stack[
0020FD6C 00000000
0020FD70 00000000
0020FD74 00000000
0020FD78 0020FDAC Stack[
0020FD7C 00000001
0020FD80 00000000
0020FD84 00000080
0020FD88 77FD5008
0020FD8C 00000005
0020FD90 00010AA0 debug0
0020FD94 00000000
0020FD98 00000000
0020FD9C 00000060
0020FDA0 0000005C
```

Base address	Type	Size	Protection	Use
> 0x10000	Private	128 kB	RW	
> 0x30000	Mapped	16 kB	R	
> 0x40000	Mapped	8 kB	R	
> 0x50000	Private	4 kB	RW	
▲ 0x60000	Mapped	68 kB	WCX	
0x60000	Mapped: Commit	68 kB	WCX	
> 0x1e000	Private	36 kB	RW	
> 0x57000	Private	36 kB	RW	
> 0x77690	Private	36 kB	RW	
> 0x778f0	Private	36 kB	RW	
> 0x7ffb0	Private	36 kB	RW	
> 0x7ffd30	Private	36 kB	RW	
> 0x7ffdf0	Private	36 kB	RW	
> 0x7ffe00	Private	36 kB	RW	

Hex View-1

0020FD84	80	00	00	00	08	30	FD	7F	05	00	00	00
0020FD94	00	00	00	00	00	00	00	00	60	00	00	00

explorer.exe (2588) (0x60000 - 0x71000)

00000000	55	8b	ec	81	ec	ac	00	00	00	53	56	57	60	fc	33	d2	U.....SVW`.3.
00000010	64	8b	15	30	00	00	00	8b	52	0c	8b	52	14	8b	72	28	d..0....R..R..r(
00000020	6a	18	59	33	ff	33	c0	ac	3c	61	7c	02	2c	20	c1	cf	j.Y3.3..<a ., ..
00000030	0d	03	f8	e2	f0	81	ff	5b	bc	4a	6a	8b	5a	10	8b	12[.Jj.Z...
00000040	75	db	89	5d	d8	61	8b	4d	d8	8b	41	3c	8b	44	08	78	u..].a.M..A<.D.x
00000050	03	c1	8b	50	10	8b	70	1c	8b	78	24	89	55	e8	8b	50	...P..p..x\$.U..P
00000060	20	03	d1	03	f9	03	f1	33	db	89	55	a0	89	7d	f4	c73..U..}..
00000070	85	5c	ff	ff	ff	47	65	74	50	c7	85	60	ff	ff	ff	72	\ GetP ` r



Malware was allowed to copy the content of explorer.exe, patch the copied content, hollow out explorer.exe and write the patched content back into remote process (explorer.exe) at 0x570000. At this point first 7 bytes of address of entry point of explorer.exe is modified

Assembly code snippet:

```

00D821CD push PAGE_EXECUTE_READWRITE
00D821CF push 0
00D821D1 push 1
00D821D3 lea eax, [ebp+sect_size]
00D821D9 push eax
00D821DA push 0
00D821DC push 0
00D821DE push 0
00D821E0 lea ecx, [ebp+view_base_addr]
00D821E6 push ecx
00D821E7 mov edx, [ebp+sus_proc_handle]
00D821ED push edx
00D821EE mov eax, [ebp+var_A0]
00D821F4 push eax
00D821F5 call [ebp+ntmapviewofsection] ; NtMapViewOfSection maps into r
00D821FB mov [ebp+var_C4], eax
00D82201 mov [ebp+EventAttributes.nLength], 0
00D8220B mov [ebp+EventAttributes.lpSecurityDe
00D82215 mov [ebp+EventAttributes.bInheritHan

```

Stack view:

Address	Value
0020FD60	00000064
0020FD64	00000060
0020FD68	0020FEF8
0020FD6C	00000000
0020FD70	00000000
0020FD74	00000000
0020FD78	0020FDAC
0020FD7C	00000001
0020FD80	00000000
0020FD84	00000040
0020FD88	7FFD3008
0020FD8C	00000005
0020FD90	00281000
0020FD94	00000000
0020FD98	00000000
0020FD9C	00000060

Memory map table:

Base address	Type	Size	Protection	Use
0x10000	Private	128 kB	RW	
0x30000	Mapped	16 kB	R	
0x40000	Mapped	8 kB	R	
0x50000	Private	4 kB	RW	
0x60000	Mapped	68 kB	WCX	
0x1e0000	Private	256 kB	RW	Stack (thread 2580)
0x570000	Mapped	2,564 kB	RWX	
0x570000	Mapped: Commit	2,564 kB	RWX	

Hex View-1:

```

0020FEF8 00 00 57 00 00 00 4E 00 00 00 00 00 00 00 00 00
0020FF08 00 00 00 00 00 10 28 00 44 00 00 00 00 00 00 00
0020FF18 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

```

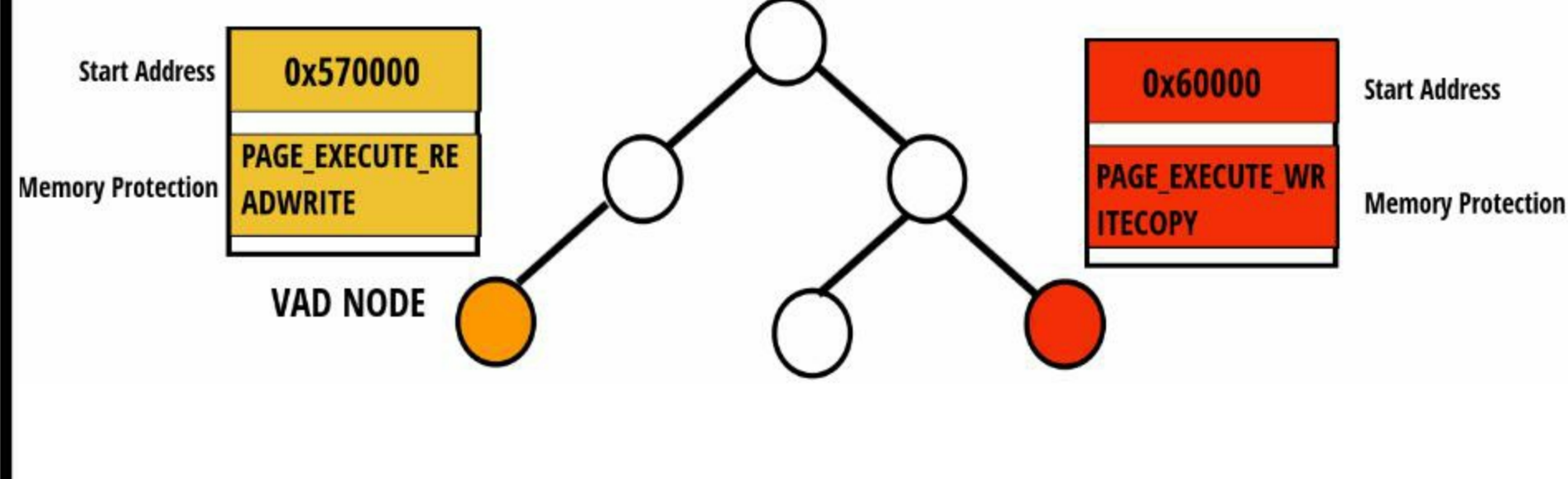
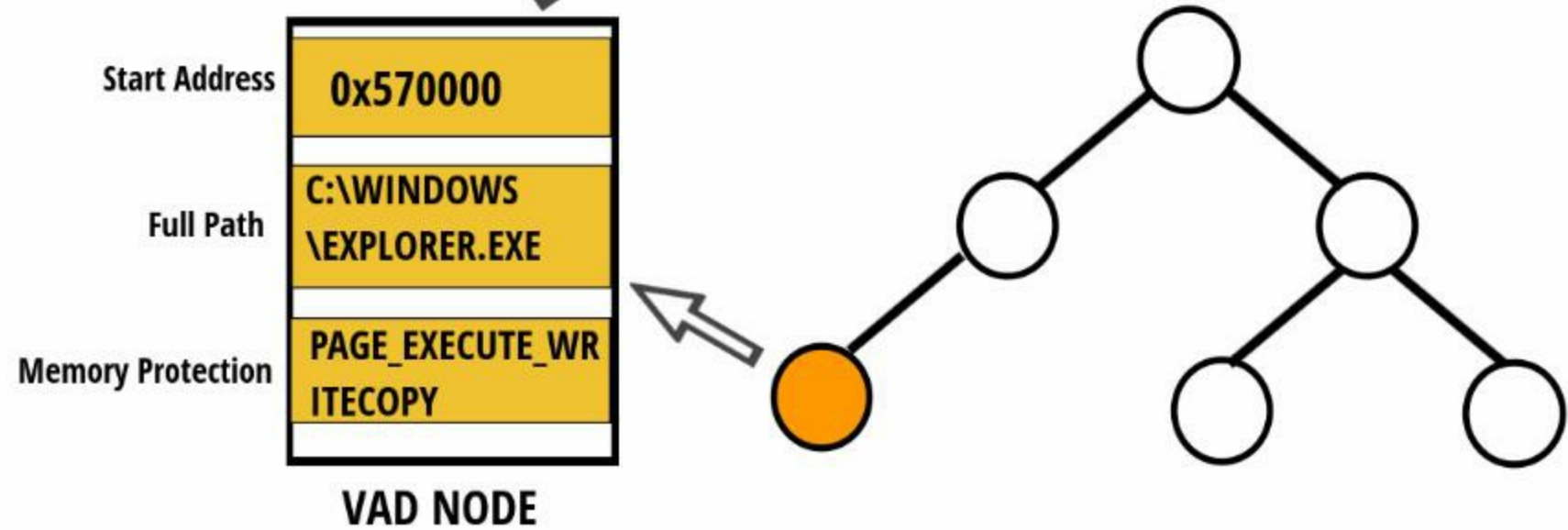
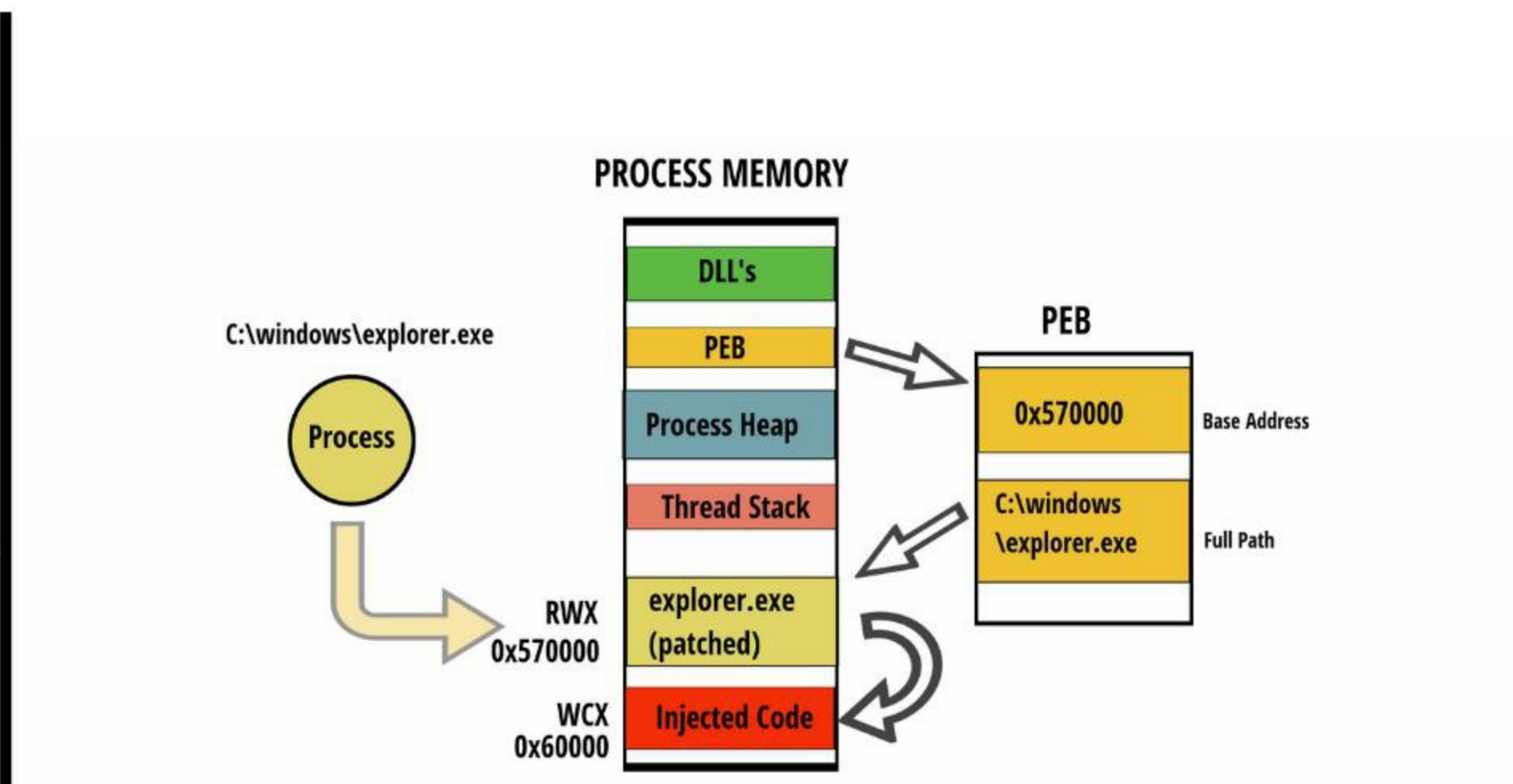
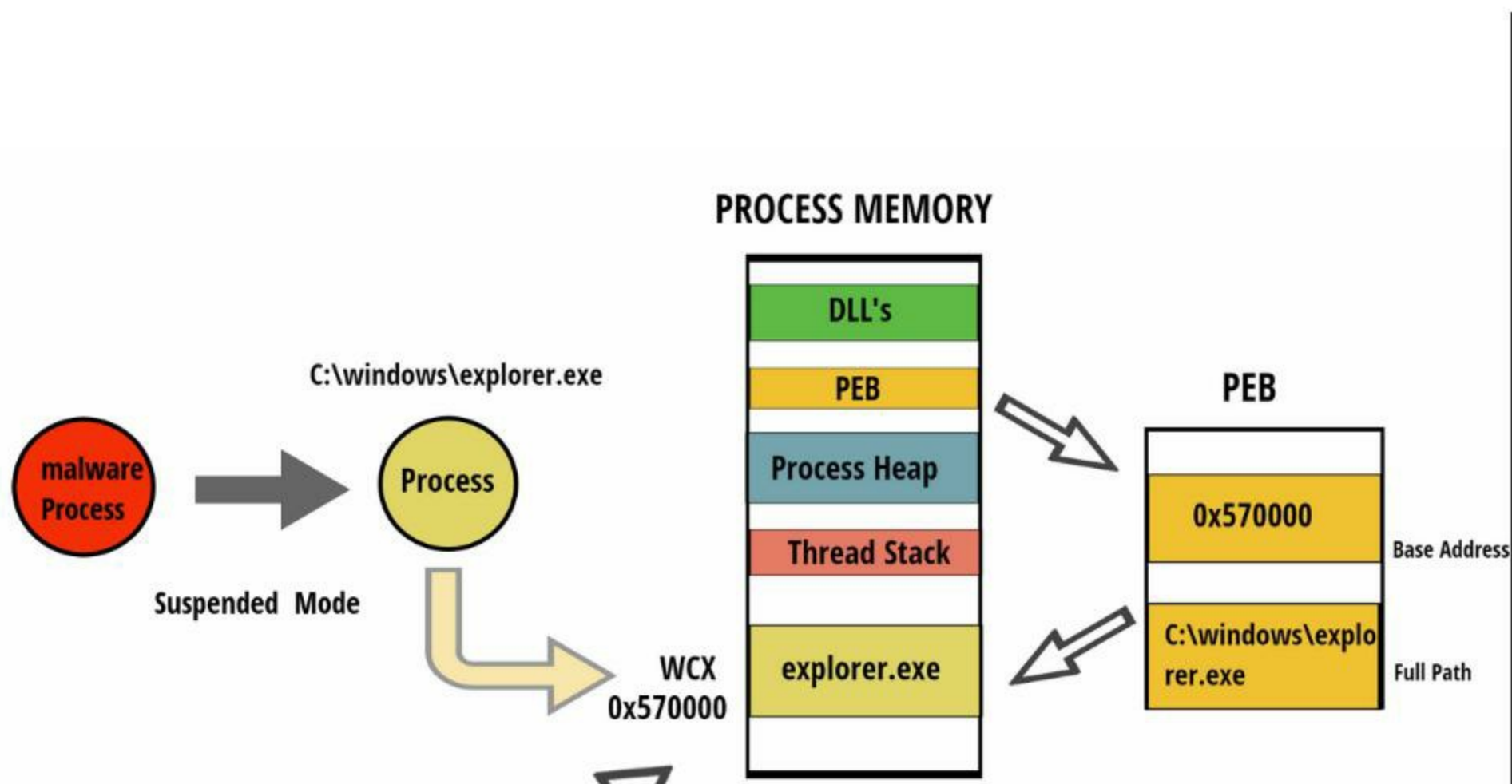
explorer.exe (2588) (0x570000 - 0x7f1000) memory dump:

```

00000000 4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00 MZ.....
00000001 b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 .....@.....
00000002 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000003 00 00 00 00 00 00 00 00 00 00 00 00 00 d8 00 00 00 .....
00000004 0e 1f ba 0e 00 b4 09 cd 21 b8 01 4c cd 21 54 68 .....!.L.!Th

```



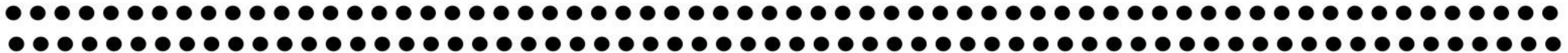


BEFORE HOLLOWING

AFTER HOLLOWING



***Demo3 - Kuluoz's modified Process
Hollowing Hides from Forensic Tools***



Hollowfind plugin detects Kuluoz's Modified Hollow Process Injection

Hollowed Process Information:

Process: explorer.exe PID: 2588 PPID: 160

Process Base Name(PEB): explorer.exe

Hollow Type: Invalid EXE Memory Protection and Process Path Discrepancy

VAD and PEB Comparison:

Base Address(VAD): 0x570000

Process Path(VAD):

Vad Protection: PAGE_EXECUTE_READWRITE

Vad Tag: Vad

Base Address(PEB): 0x570000

Process Path(PEB): C:\Windows\explorer.exe

Memory Protection: PAGE_EXECUTE_READWRITE

Memory Tag: Vad

Disassembly(Entry Point):

```
0x005a0efa 90          NOP
0x005a0efb 6800000600 PUSH DWORD 0x60000
0x005a0f00 c3          RET
0x005a0f01 6830105a00 PUSH DWORD 0x5a1030
0x005a0f06 e8c11d0000 CALL 0x5a2ccc
```

Similar Processes:

explorer.exe(2588) Parent:msorder.exe(160) Start:2016-06-26 10:04:34 UTC+0000

explorer.exe(1348) Parent:NA(1328) Start:2016-06-24 13:28:21 UTC+0000

Suspicious Memory Regions:

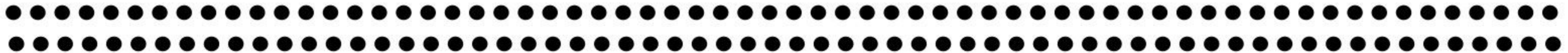
0x60000(No PE/Possibly Code) Protection: PAGE_EXECUTE_WRITECOPY Tag: Vad

0x370000(No PE/Possibly Code) Protection: PAGE_EXECUTE_READWRITE Tag: VadS

0x570000(PE Found) Protection: PAGE_EXECUTE_READWRITE Tag: Vad

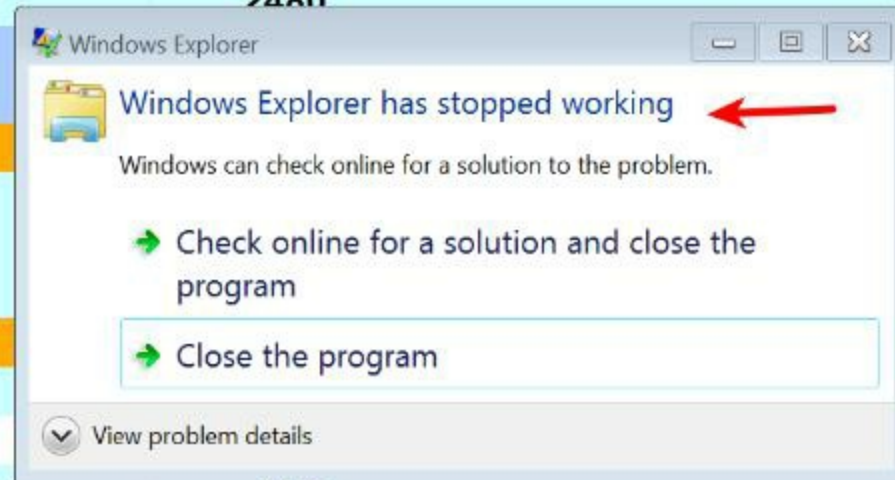


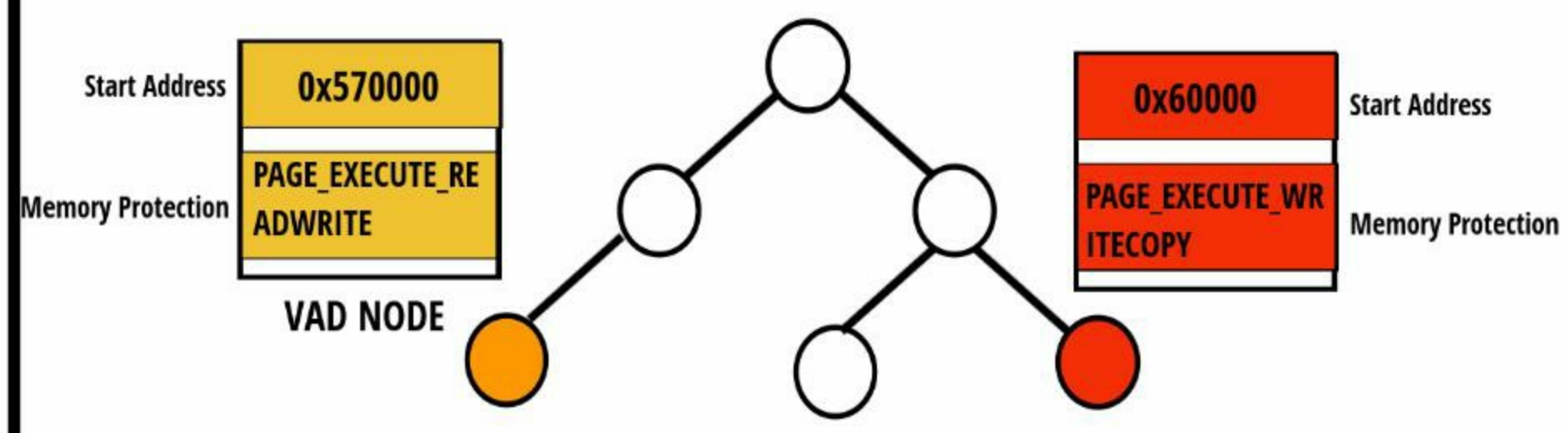
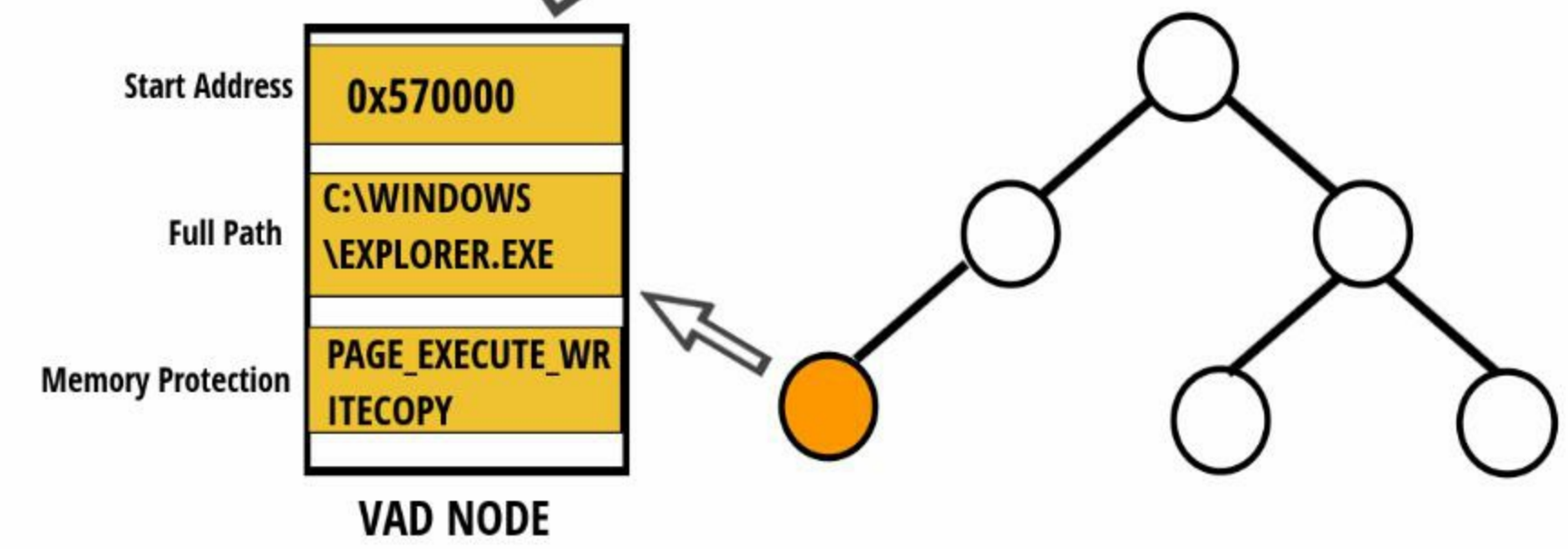
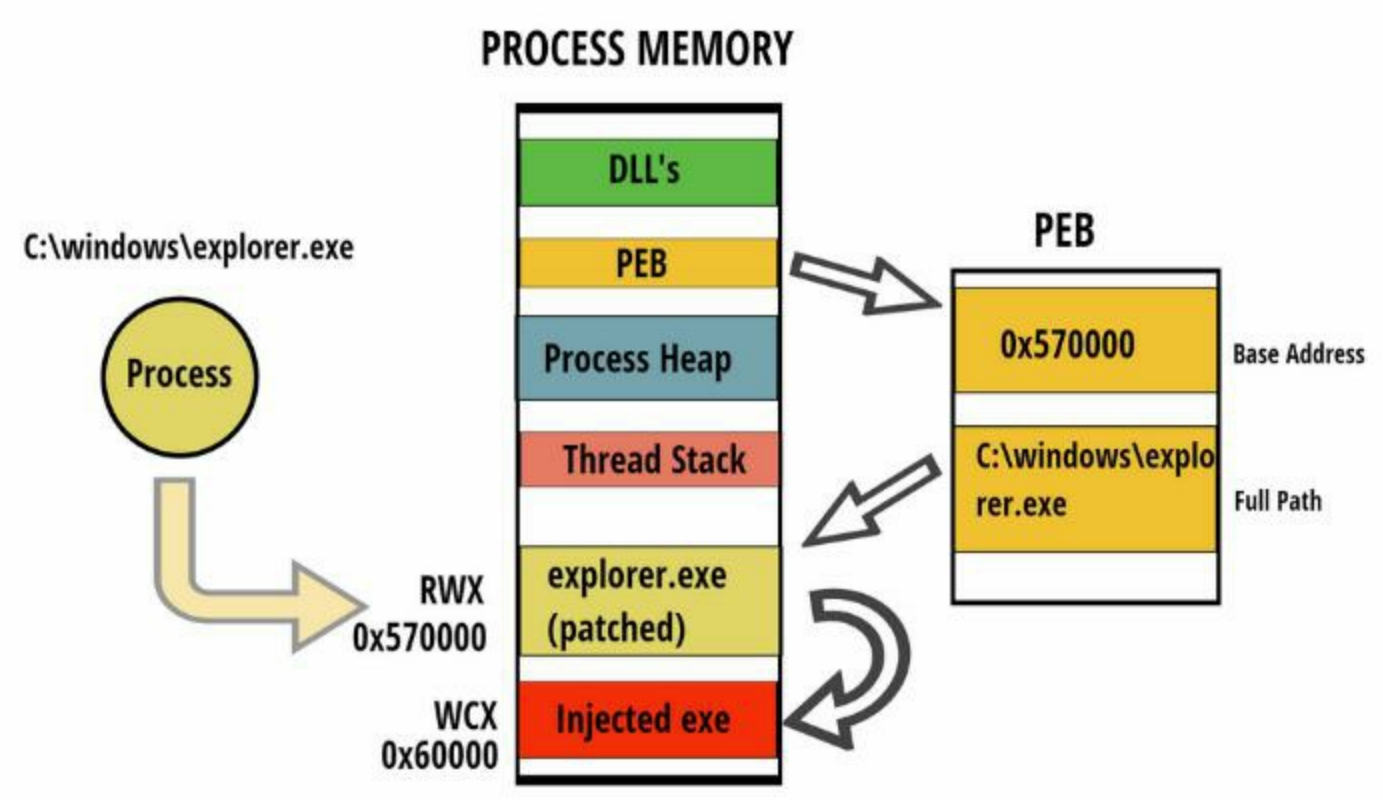
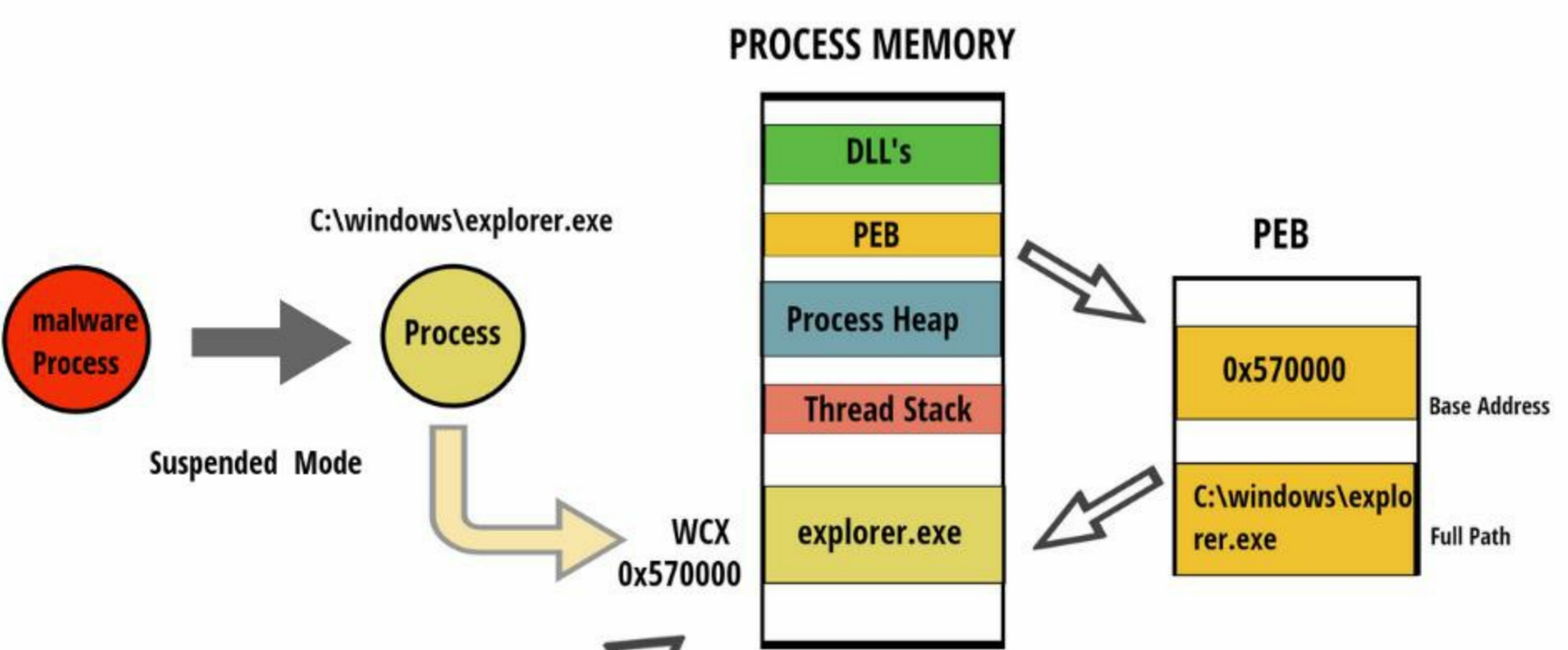
***Evasive Hollow Process Injection -
Kronos***



Kronos performs similar hollow process redirection technique as kuluoZ, this sample hollows out the explorer.exe process, patches the address of entry point and attempts to redirect execution flow inside an executable (instead of code) which was injected with PAGE_EXECUTE_WRITECOPY protections. While testing this executable the explorer.exe crashed as shown below, but still the memory image was taken for further analysis.

Name	PID	CPU	I/O total ...	Private bytes	User name	Description
svchost.exe	1876			1.19 MB	... \NETWORK S	Host Process for Wi
dllhost.exe	2088			2.86 MB	NT ... \SYSTEM	COM Surrogate
msdtc.exe	2256			2.5 MB	... \NETWORK S	Microsoft Distribute
SearchIndexer.exe	2480			15.99 MB	NT ... \SYSTEM	Microsoft Windows
SearchProtocolHost.exe				2.16 MB	NT ... \SYSTEM	Microsoft Windows
SearchFilterHost.exe				1.28 MB	NT ... \SYSTEM	Microsoft Windows
taskhost.exe				2.6 MB	WIN-T9... \test	Host Process for Wi
wmpnetwk.exe				8.27 MB	... \NETWORK S	Windows Media Play
svchost.exe				3.12 MB	... \LOCAL SER\	Host Process for Wi
svchost.exe				776 kB	NT ... \SYSTEM	Host Process for Wi
WerFault.exe				4.23 MB	WIN-T9... \test	Windows Problem R
WmiApSrv.exe				1.14 MB	NT ... \SYSTEM	WMI Performance R
taskhost.exe				4.94 MB	... \LOCAL SER\	Host Process for Wi
lsass.exe	504			2.76 MB	NT ... \SYSTEM	Local Security Autho
lsmd.exe	512			1.18 MB	NT ... \SYSTEM	Local Session Mana
csrss.exe	404	0.10		17.07 MB	NT ... \SYSTEM	Client Server Runtim
conhost.exe	2708			620 kB	WIN-T9... \test	Console Window Ho
winlogon.exe	452			2.3 MB	NT ... \SYSTEM	Windows Logon App
explorer.exe	1364	0.03		31.48 MB	WIN-T9... \test	Windows Explorer
vmtoolsd.exe	1560	1.59	174.41 k...	5.3 MB	WIN-T9... \test	VMware Tools Core
ProcessHacker.exe	3908	0.81		8.69 MB	WIN-T9... \test	Process Hacker
explorer.exe	860			1.39 MB	WIN-T9... \test	Windows Explorer





BEFORE HOLLOWING

AFTER HOLLOWING



***Demo4 - Kronos modified Process
Hollowing Hides from Forensic Tools***








Black Hat Sound Bytes

Key Take Aways:

- ⚙️ Attackers find new ways to bypass, confuse & divert analysis***
- ⚙️ Tools don't work always as expected***
- ⚙️ Understanding the working of these stealth techniques will help better detect & counter such attacks***



References

-  <https://cysinfo.com/detecting-deceptive-hollowing-techniques/>
-  http://www.volatilityfoundation.org/#!releases/component_71401
-  <https://www.trustwave.com/Resources/SpiderLabs-Blog/Analyzing-Malware-Hollow-Processes/>
-  <http://mnin.blogspot.in/2011/06/examining-stuxnets-footprint-in-memory.html>
-  <http://journeyintoir.blogspot.in/2015/02/process-hollowing-meets-cuckoo-sandbox.html>

THANK YOU



@monnappa22



<http://www.youtube.com/c/MonnappaKA>



monnappa22@gmail.com



<https://www.cysinfo.com>