Unpacking Hancitor malware

muha2xmad.github.io/unpacking/hancitor/

January 8, 2022

Muhammad Hasan Ali

Malware Analysis learner

1 minute read

As-salamu Alaykum

Introduction

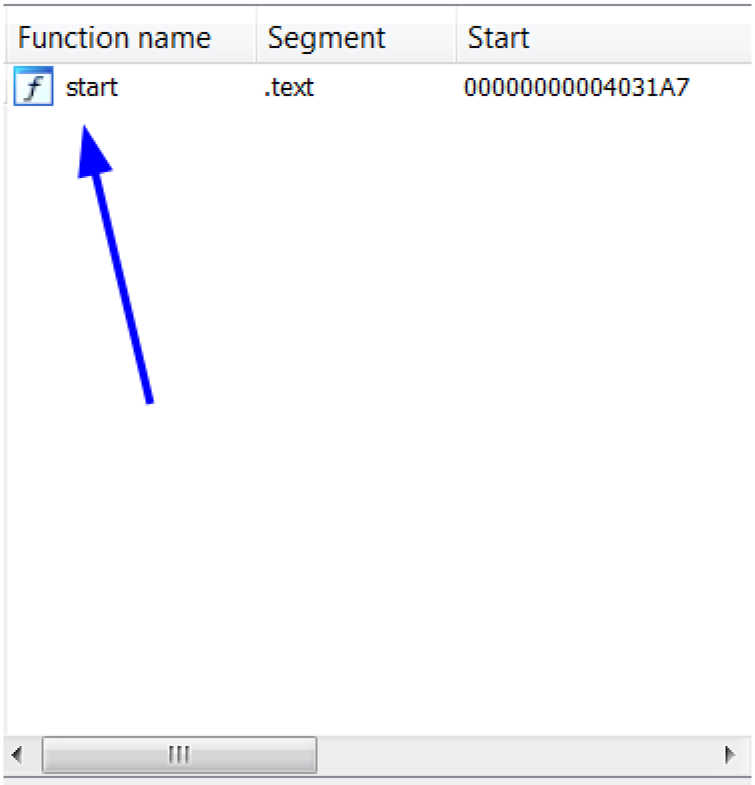
Hancitor is an information stealer and malware downloader used by a threat actor designated as MAN1, Moskalvzapoe or TA511. In a threat brief from 2018, we noted Hancitor was relatively unsophisticated, but it would remain a threat for years to come. Approximately three years later, Hancitor remains a threat and has evolved to use tools like Cobalt Strike. In recent months, this actor began using a network ping tool to help enumerate the Active Directory (AD) environment of infected hosts. This blog illustrates how the threat actor behind Hancitor uses the network ping tool, so security professionals can better identify and block its use. 1

MD5: FF9D0327538AB33468A8AD2142EFF416

Packed indicators

If we open it in `DiE` or `pestudio` we will notice that it’s **Not packed**.

Another way to detect packing is to open it in `IDA` and if you see less number of functions then It’s packed.



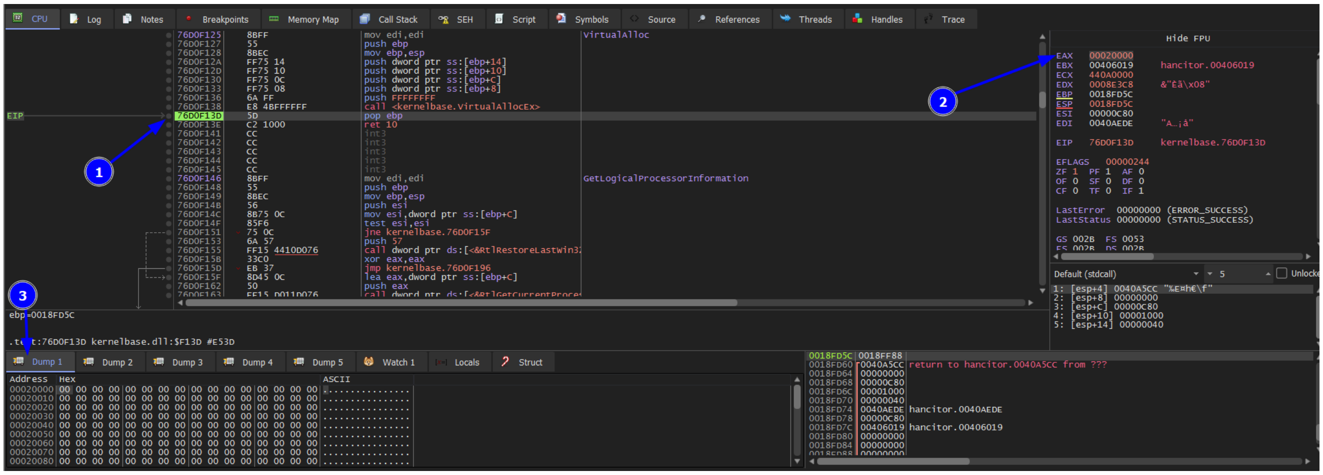
Function name	Segment	Start
start	.text	00000000004031A7

Figure(1):

Unpacking process

How unpacking works? well, packing process depands on the packer. So each packer has different unpacking routine.

So in this sample we will begin by setting two breakpoints in `VirtualAlloc` and `VirtualProtect` . Then run `f9` to hit the first breakpoint and then run again to hit the 2nd one which is `VirtualAlloc` . Then step over `f8` to the call of virtualalloc. Then dump `EAX` .



```
76D0F125   8BFF          mov edi,edi                                  VirtualAlloc
76D0F127   55            push ebp
76D0F128   8BEC          mov ebp,esp
76D0F12A   FF75 14       push dword ptr ss:[ebp+14]
76D0F12D   FF75 10       push dword ptr ss:[ebp+10]
76D0F130   FF75 0C       push dword ptr ss:[ebp+C]
76D0F133   FF75 08       push dword ptr ss:[ebp+8]
76D0F136   6A FF         push FFFFFFFF
76D0F138   E8 4BFFFFFF   call <kernelbase.VirtualAllocEx>
EIP  76D0F13D   5D       pop ebp
76D0F13E   C2 1000       ret 10
76D0F141   CC            int3
76D0F142   CC            int3
76D0F143   CC            int3
76D0F144   CC            int3
76D0F145   CC            int3
76D0F146   8BFF          mov edi,edi                                  GetLogicalProcessorInformation
76D0F148   55            push ebp
76D0F149   8BEC          mov ebp,esp
76D0F14B   56            push esi
76D0F14C   8B75 0C       mov esi,dword ptr ss:[ebp+C]
76D0F14F   85F6          test esi,esi
76D0F151   75 0C         jne kernelbase.76D0F15F
76D0F153   6A 57         push 57
76D0F155   FF15 4410D076 call dword ptr ds:[<&RtlRestoreLastWin3
76D0F15B   33C0          xor eax,eax
76D0F15D   EB 37         jmp kernelbase.76D0F196
76D0F15F   8D45 0C       lea eax,dword ptr ss:[ebp+C]
76D0F162   50            push eax
76D0F163   FF15 D011D076 call dword ptr ds:[<&RtlGetCurrentProce

ebp=0018FD5C

.text:76D0F13D kernelbase.dll:$F13D #E53D
```

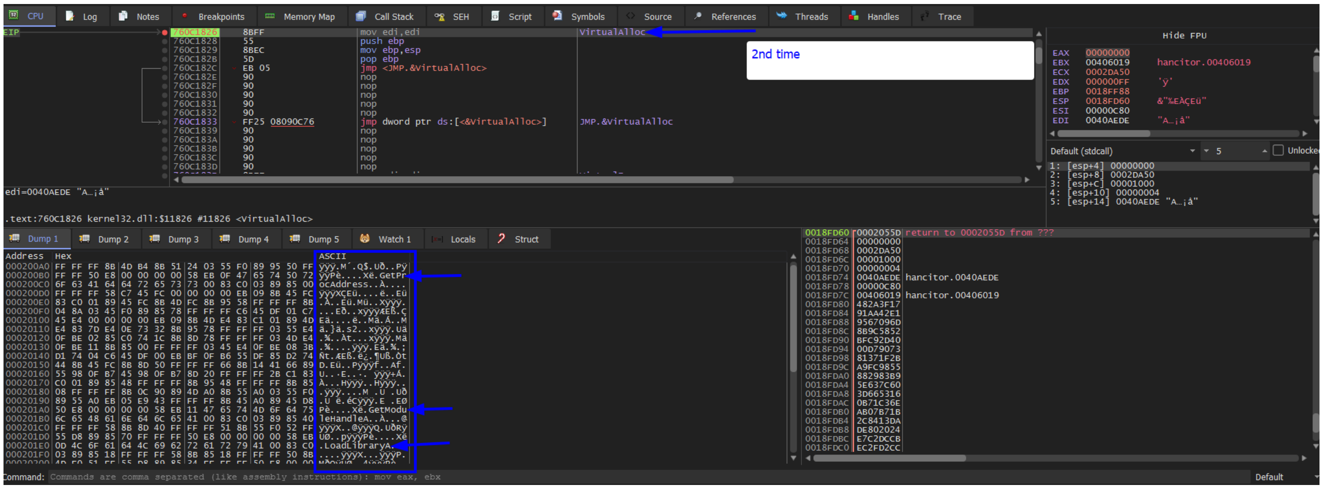
```
EAX     00020000
EBX     00406019     hancitor.00406019
ECX     440A0000
EDX     0008E3C8     &"Èã\x08"
EBP     0018FD5C
ESP     0018FD5C
ESI     00000C80
EDI     0040AEDE     "A…¡â"

EIP     76D0F13D     kernelbase.76D0F13D

EFLAGS    00000244
ZF 1   PF 1   AF 0
OF 0   SF 0   DF 0
CF 0   TF 0   IF 1

LastError   00000000 (ERROR_SUCCESS)
LastStatus  00000000 (STATUS_SUCCESS)

GS 002B   FS 0053
ES 002B   DS 002B

Default (stdcall)   5   Unlocked
1: [esp+4] 0040A5CC "‰E¤h€\f"
2: [esp+8] 00000000
3: [esp+C] 00000C80
4: [esp+10] 00001000
5: [esp+14] 00000040
```

```
Address   Hex                                               ASCII
00020000  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00020010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00020020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00020030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00020040  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00020050  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00020060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00020070  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00020080  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

```
0018FD5C  0018FF88
0018FD60  0040A5CC  return to hancitor.0040A5CC from ???
0018FD64  00000000
0018FD68  00000C80
0018FD6C  00001000
0018FD70  00000040
0018FD74  0040AEDE  hancitor.0040AEDE
0018FD78  00000C80
0018FD7C  00406019  hancitor.00406019
0018FD80  00000000
0018FD84  00000000
0018FD88  00000000
```

Figure(2):

Then run again to get to virtualalloc to the 2nd time and see what dump has.



```
EIP  760C1826   8BFF          mov edi,edi                                  VirtualAlloc
760C1828   55                 push ebp
760C1829   8BEC               mov ebp,esp
760C182B   5D                 pop ebp
760C182C   EB 05              jmp <JMP.&VirtualAlloc>
760C182E   90                 nop
760C182F   90                 nop
760C1830   90                 nop
760C1831   90                 nop
760C1832   90                 nop
760C1833   FF25 08090C76      jmp dword ptr ds:[<&VirtualAlloc>]           JMP.&VirtualAlloc
760C1839   90                 nop
760C183A   90                 nop
760C183B   90                 nop
760C183C   90                 nop
760C183D   90                 nop

edi=0040AEDE "A…¡â"

.text:760C1826 kernel32.dll:$11826 #11826 <VirtualAlloc>
```

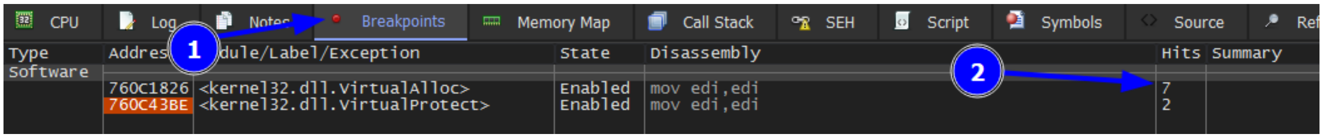
```
EAX     00000000
EBX     00406019     hancitor.00406019
ECX     0002DA50
EDX     000000FF     'ÿ'
EBP     0018FF88
ESP     0018FD60     &"‰EÀÇEü"
ESI     00000C80
EDI     0040AEDE     "A…¡â"

Default (stdcall)   5   Unlocked
1: [esp+4] 00000000
2: [esp+8] 0002DA50
3: [esp+C] 00001000
4: [esp+10] 00000004
5: [esp+14] 0040AEDE "A…¡â"
```

```
Address   Hex                                               ASCII
000200A0  FF FF FF 8B 4D B4 8B 51 24 03 55 F0 89 95 50 FF  ÿÿÿ.M´.Q$.Uð..Pÿ
000200B0  FF FF 50 E8 00 00 00 00 58 EB 0F 47 65 74 50 72  ÿÿPè....Xë.GetPr
000200C0  6F 63 41 64 64 72 65 73 73 00 83 C0 03 89 85 00  ocAddress..À....
000200D0  FF FF FF 58 C7 45 FC 00 00 00 00 EB 09 8B 45 FC  ÿÿÿXÇEü....ë..Eü
000200E0  83 C0 01 89 45 FC 8B 4D FC 8B 95 58 FF FF FF 8B  .À..Eü.Mü..Xÿÿÿ.
000200F0  04 8A 03 45 F0 89 85 78 FF FF FF C6 45 DF 01 C7  ...Eð..xÿÿÿÆEß.Ç
00020100  45 E4 00 00 00 00 EB 09 8B 4D E4 83 C1 01 89 4D  Eä....ë..Mä.Á..M
00020110  E4 83 7D E4 0E 73 32 8B 95 78 FF FF FF 03 55 E4  ä.}ä.s2..xÿÿÿ.Uä
00020120  0F BE 02 85 C0 74 1C 8B 8D 78 FF FF FF 03 4D E4  .¾..Àt...xÿÿÿ.Mä
00020130  0F BE 11 8B 85 00 FF FF FF 03 45 E4 0F BE 08 3B  .¾....ÿÿÿ.Eä.¾.;
00020140  D1 74 04 C6 45 DF 00 EB BF 0F B6 55 DF 85 D2 74  Ñt.ÆEß.ë¿.¶Uß.Òt
00020150  44 8B 45 FC 8B 8D 50 FF FF FF 66 8B 14 41 66 89  D.Eü..Pÿÿÿf..Af.
00020160  55 98 0F B7 45 98 0F B7 8D 20 FF FF FF 2B C1 83  U..·E..·. ÿÿÿ+Á.
00020170  C0 01 89 85 48 FF FF FF 8B 95 48 FF FF FF 8B 85  À...Hÿÿÿ..Hÿÿÿ..
00020180  08 FF FF FF 8B 0C 90 89 4D A0 8B 55 A0 03 55 F0  .ÿÿÿ....M .U .Uð
00020190  89 55 A0 EB 05 E9 43 FF FF FF 8B 45 A0 89 45 D8  .U ë.éCÿÿÿ.E .EØ
000201A0  50 E8 00 00 00 00 58 EB 11 47 65 74 4D 6F 64 75  Pè....Xë.GetModu
000201B0  6C 65 48 61 6E 64 6C 65 41 00 83 C0 03 89 85 40  leHandleA..À...@
000201C0  FF FF FF 58 8B 8D 40 FF FF FF 51 8B 55 F0 52 FF  ÿÿÿX..@ÿÿÿQ.UðRÿ
000201D0  55 D8 89 85 70 FF FF FF 50 E8 00 00 00 00 58 EB  UØ..pÿÿÿPè....Xë
000201E0  0D 4C 6F 61 64 4C 69 62 72 61 72 79 41 00 83 C0  .LoadLibraryA..À
000201F0  03 89 85 18 FF FF FF 58 8B 85 18 FF FF FF 50 8B  ....ÿÿÿX...ÿÿÿP.
```

```
0018FD60  0002055D  return to 0002055D from ???
0018FD64  00000000
0018FD68  0002DA50
0018FD6C  00001000
0018FD70  00000004
0018FD74  0040AEDE  hancitor.0040AEDE
0018FD78  00000C80
0018FD7C  00406019  hancitor.00406019
0018FD80  482A3F17
0018FD84  91AA42E1
0018FD88  9567096D
0018FD8C  8B9C5852
0018FD90  BFC92D40
0018FD94  00D79073
0018FD98  81371F2B
0018FD9C  A9FC9855
0018FDA0  882983B9
0018FDA4  5E637C60
0018FDA8  3D665316
0018FDAC  0B71C36E
0018FDB0  AB07B71B
0018FDB4  2C8413DA
0018FDB8  DE802024
0018FDBC  E7C2DCCB
0018FDC0  EC2FD2CC

Command: Commands are comma separated (like assembly instructions): mov eax, ebx
```

Figure(3):

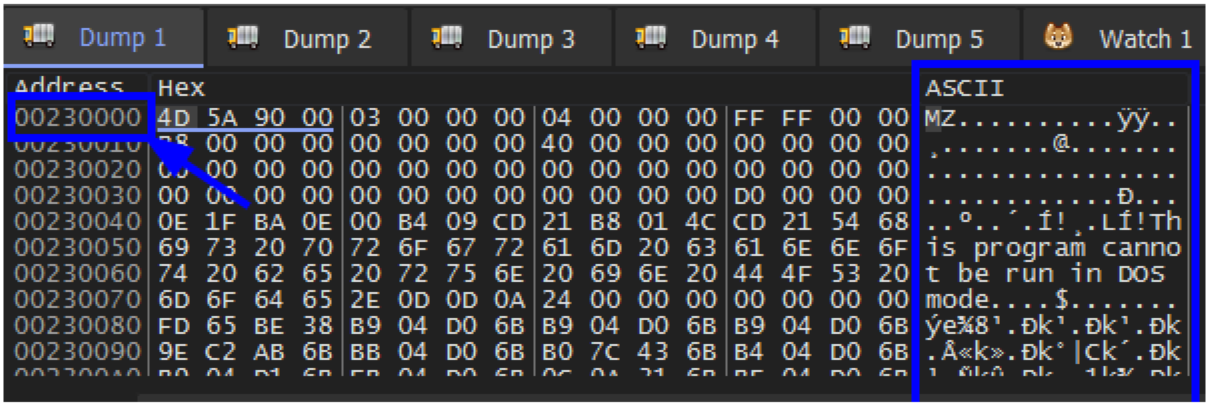
We keep pressing `run` to hit the breakpoint of `VirtualAlloc` **7 times**. After that we we hit `run` again.



Type	Addres	dule/Label/Exception	State	Disassembly	Hits	Summary
Software						
	760C1826	<kernel32.dll.VirtualAlloc>	Enabled	mov edi,edi	7	
	760C43BE	<kernel32.dll.VirtualProtect>	Enabled	mov edi,edi	2	

Figure(4):

We will see our unpacked `exe` in the dump. Then we `Follow in Memory map` and save to file. And `Image base` is `230000` .



```
Address  Hex                                              ASCII
00230000 4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 MZ..........ÿÿ..
00230010 B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 ¸.......@.......
00230020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00230030 00 00 00 00 00 00 00 00 00 00 00 00 D0 00 00 00 ............Ð...
00230040 0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68 ..º..´.Í!¸.LÍ!Th
00230050 69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F is program canno
00230060 74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20 t be run in DOS 
00230070 6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00 mode....$.......
00230080 FD 65 BE 38 B9 04 D0 6B B9 04 D0 6B B9 04 D0 6B ýe¾8¹.Ðk¹.Ðk¹.Ðk
00230090 9E C2 AB 6B BB 04 D0 6B B0 7C 43 6B B4 04 D0 6B .Â«k».Ðk°|Ck´.Ðk
```

Figure(5):

Unmap the unpacked file

To get `improts` of the unpacked file. We need to repair the section headers see my article Here. After unmapping you can see `Imports` and `libraries`.

Article quote

> وماذا يهم إذا لم تكن الحياة على ما يُرام؟

REF

1- https://unit42.paloaltonetworks.com/hancitor-infections-cobalt-strike/

2- VirtualAlloc

3- VirtualProtect