Lab 13: Hacking Minesweeper with Ollydbg

Course Name: Malware Analysis and Reverse Engineering (IAM302)
Student Name: Nguyễn Trần Vinh – SE160258
Instructor Name: Mai Hoàng Đỉnh
Lab Due Date: 29/2/2023

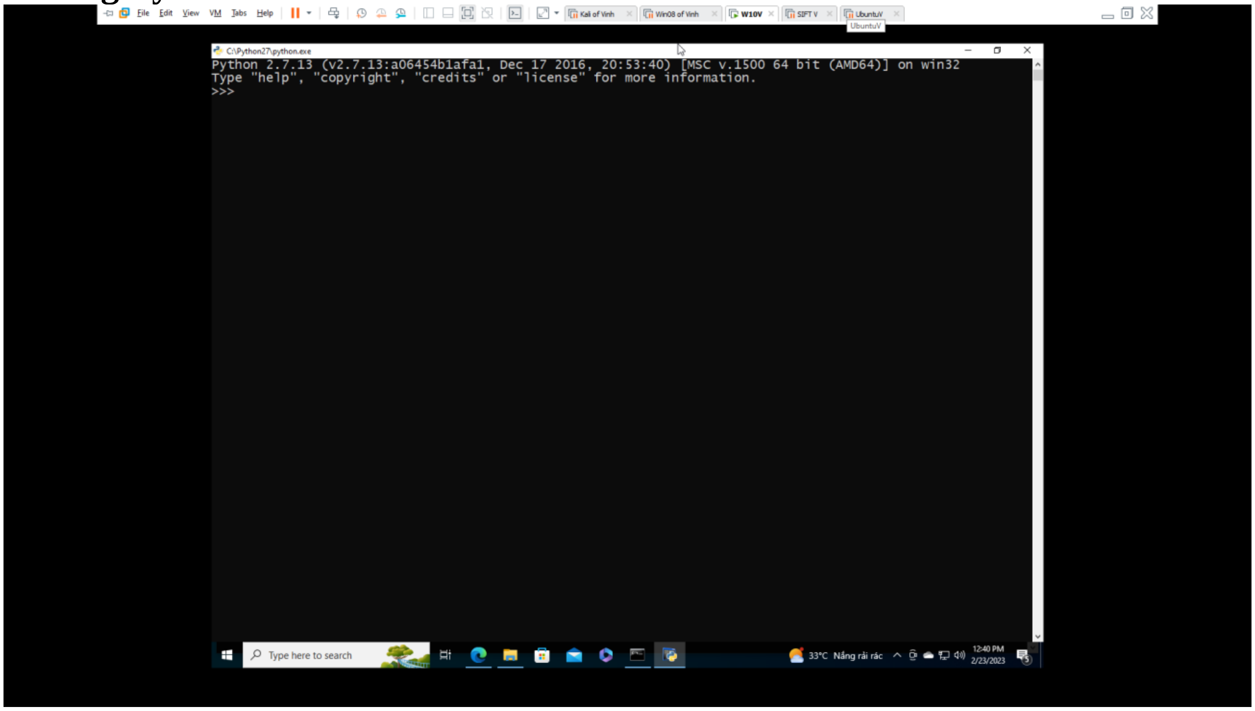
Purpose

To hack MineSweeper at the binary level. This gives you practice using the Ollydbg debugger, Procdump, and Python

What You Need

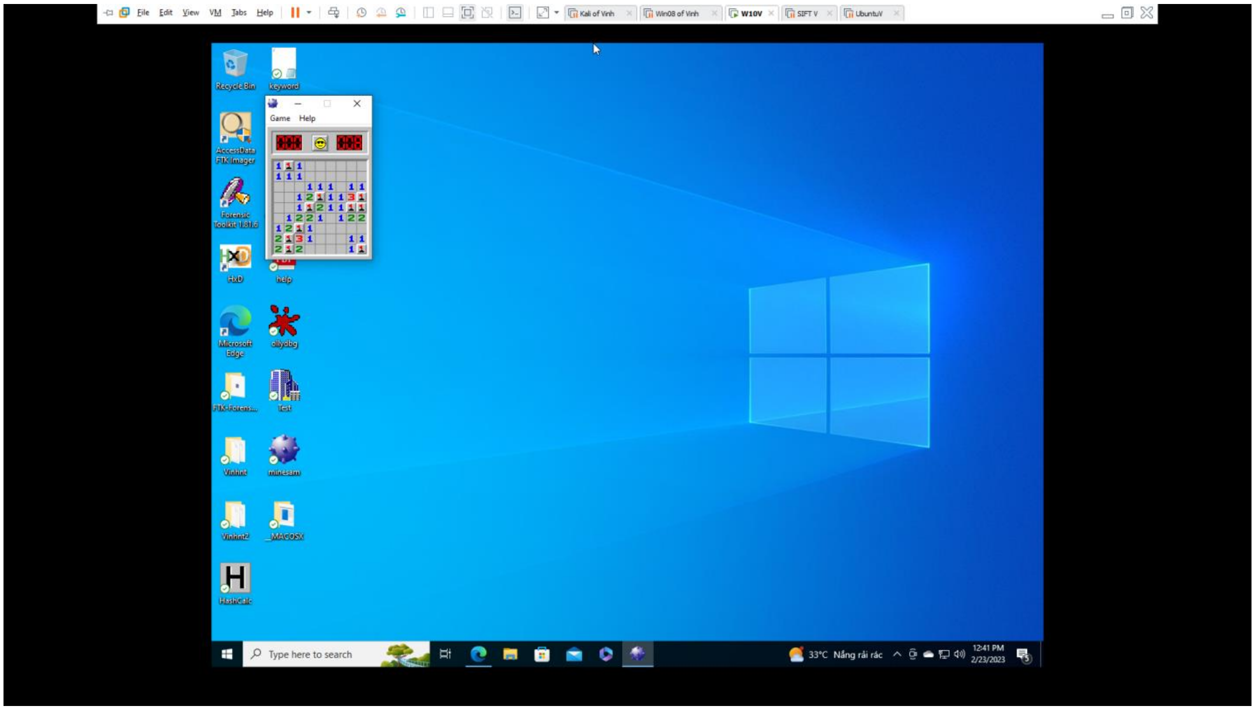
A Windows machine, real or virtual. I used a Windows Server 2008 virtual machine

Getting Python

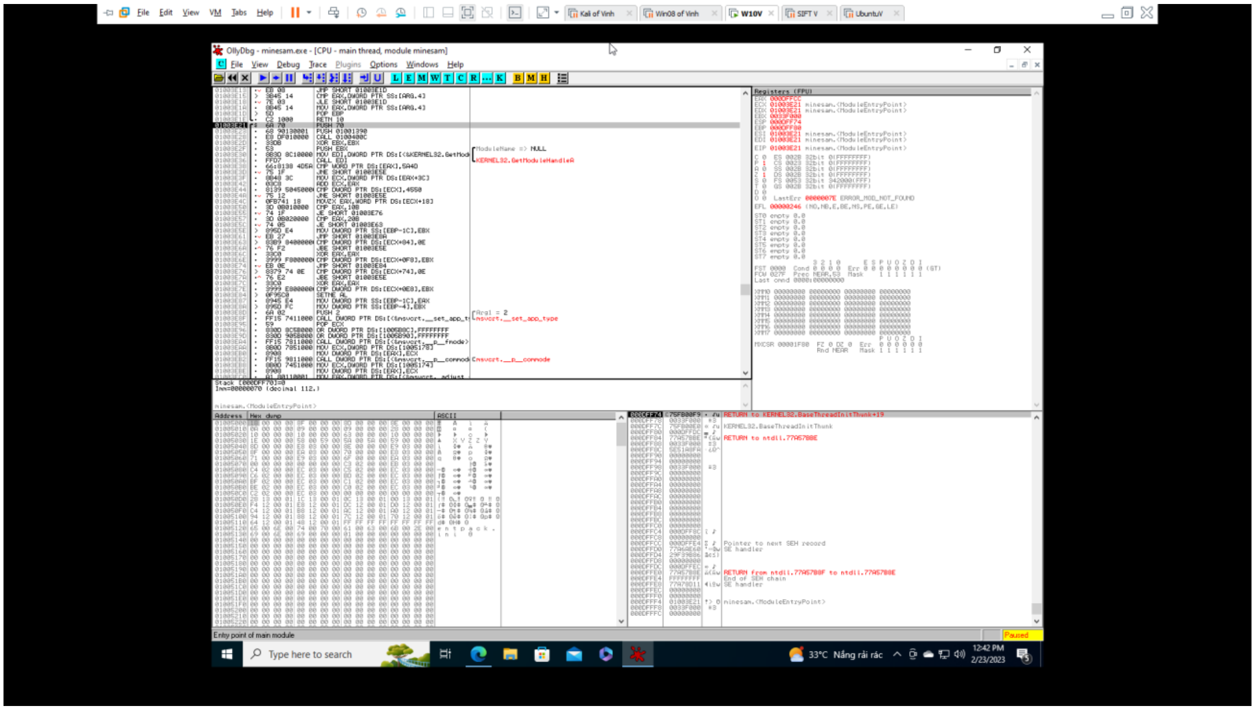


```
Python 2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 20:53:40) [MSC v.1500 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Getting Minesweeper



Viewing the Game in OllyDbg


File Edit View VM Tabs Help
Kali of Vinh
Win08 of Vinh
W10V
SIFT V
UbuntuV
OllyDbg - minesam.exe - [CPU - main thread, module minesam]
C File View Debug Trace Plugins Options Windows Help
L E M W T C R ... K B M H

```
01003E13  ∙v EB 08          JMP SHORT 01003E1D
01003E15  >  3B45 14        CMP EAX,DWORD PTR SS:[ARG.4]
01003E18  ∙v 7E 03          JLE SHORT 01003E1D
01003E1A  ∙  8B45 14        MOV EAX,DWORD PTR SS:[ARG.4]
01003E1D  >  5D             POP EBP
01003E1E  ∙  C2 1000        RETN 10
01003E21  $  6A 70          PUSH 70
01003E23  ∙  68 90130001    PUSH 01001390
01003E28  ∙  E8 DF010000    CALL 0100400C
01003E2D  ∙  33DB           XOR EBX,EBX
01003E2F  ∙  53             PUSH EBX                                  ┌ModuleName => NULL
01003E30  ∙  8B3D 8C10000   MOV EDI,DWORD PTR DS:[<&KERNEL32.GetMod   │
01003E36  ∙  FFD7           CALL EDI                                  └KERNEL32.GetModuleHandleA
01003E38  ∙  66:8138 4D5A   CMP WORD PTR DS:[EAX],5A4D
01003E3D  ∙v 75 1F          JNE SHORT 01003E5E
01003E3F  ∙  8B48 3C        MOV ECX,DWORD PTR DS:[EAX+3C]
01003E42  ∙  03C8           ADD ECX,EAX
01003E44  ∙  8139 5045000   CMP DWORD PTR DS:[ECX],4550
01003E4A  ∙v 75 12          JNE SHORT 01003E5E
01003E4C  ∙  0FB741 18      MOVZX EAX,WORD PTR DS:[ECX+18]
01003E50  ∙  3D 0B010000    CMP EAX,10B
01003E55  ∙v 74 1F          JE SHORT 01003E76
01003E57  ∙  3D 0B020000    CMP EAX,20B
01003E5C  ∙v 74 05          JE SHORT 01003E63
01003E5E  >  895D E4        MOV DWORD PTR SS:[EBP-1C],EBX
01003E61  ∙v EB 27          JMP SHORT 01003E8A
01003E63  >  83B9 8400000   CMP DWORD PTR DS:[ECX+84],0E
01003E6A  ∙^ 76 F2          JBE SHORT 01003E5E
01003E6C  ∙  33C0           XOR EAX,EAX
01003E6E  ∙  3999 F800000   CMP DWORD PTR DS:[ECX+0F8],EBX
01003E74  ∙v EB 0E          JMP SHORT 01003E84
01003E76  >  8379 74 0E     CMP DWORD PTR DS:[ECX+74],0E
01003E7A  ∙^ 76 E2          JBE SHORT 01003E5E
01003E7C  ∙  33C0           XOR EAX,EAX
01003E7E  ∙  3999 E800000   CMP DWORD PTR DS:[ECX+0E8],EBX
01003E84  >  0F95C0         SETNE AL
01003E87  ∙  8945 E4        MOV DWORD PTR SS:[EBP-1C],EAX
01003E8A  >  895D FC        MOV DWORD PTR SS:[EBP-4],EBX
01003E8D  ∙  6A 02          PUSH 2                                    ┌Arg1 = 2
01003E8F  ∙  FF15 7411000   CALL DWORD PTR DS:[<&msvcrt.__set_app_t   └msvcrt.__set_app_type
01003E95  ∙  59             POP ECX
01003E96  ∙  830D 8C5B000   OR DWORD PTR DS:[1005B8C],FFFFFFFF
01003E9D  ∙  830D 905B000   OR DWORD PTR DS:[1005B90],FFFFFFFF
01003EA4  ∙  FF15 7811000   CALL DWORD PTR DS:[<&msvcrt.__p__fmode>
01003EAA  ∙  8B0D 7851000   MOV ECX,DWORD PTR DS:[1005178]
01003EB0  ∙  8908           MOV DWORD PTR DS:[EAX],ECX
01003EB2  ∙  FF15 9811000   CALL DWORD PTR DS:[<&msvcrt.__p__commod   Cmsvcrt.__p__commode
01003EB8  ∙  8B0D 7451000   MOV ECX,DWORD PTR DS:[1005174]
01003EBE  ∙  8908           MOV DWORD PTR DS:[EAX],ECX
01003EC0  ∙  A1 80110001    MOV EAX,DWORD PTR DS:[<&msvcrt._adjust

Stack [000DFF70]=0
Imm=00000070 (decimal 112.)

minesam.<ModuleEntryPoint>
```

```
Registers (FPU)
EAX 000DFFCC
ECX 01003E21 minesam.<ModuleEntryPoint>
EDX 01003E21 minesam.<ModuleEntryPoint>
EBX 0033F000
ESP 000DFF74
EBP 000DFF80
ESI 01003E21 minesam.<ModuleEntryPoint>
EDI 01003E21 minesam.<ModuleEntryPoint>

EIP 01003E21 minesam.<ModuleEntryPoint>

C 0  ES 002B 32bit 0(FFFFFFFF)
P 1  CS 0023 32bit 0(FFFFFFFF)
A 0  SS 002B 32bit 0(FFFFFFFF)
Z 1  DS 002B 32bit 0(FFFFFFFF)
S 0  FS 0053 32bit 342000(FFF)
T 0  GS 002B 32bit 0(FFFFFFFF)
D 0
O 0  LastErr 0000007E ERROR_MOD_NOT_FOUND

EFL 00000246 (NO,NB,E,BE,NS,PE,GE,LE)

ST0 empty 0.0
ST1 empty 0.0
ST2 empty 0.0
ST3 empty 0.0
ST4 empty 0.0
ST5 empty 0.0
ST6 empty 0.0
ST7 empty 0.0
               3 2 1 0      E S P U O Z D I
FST 0000  Cond 0 0 0 0  Err 0 0 0 0 0 0 0 0 (GT)
FCW 027F  Prec NEAR,53  Mask    1 1 1 1 1 1
Last cmnd 0000:00000000

XMM0 00000000 00000000 00000000 00000000
XMM1 00000000 00000000 00000000 00000000
XMM2 00000000 00000000 00000000 00000000
XMM3 00000000 00000000 00000000 00000000
XMM4 00000000 00000000 00000000 00000000
XMM5 00000000 00000000 00000000 00000000
XMM6 00000000 00000000 00000000 00000000
XMM7 00000000 00000000 00000000 00000000
                                P U O Z D I
MXCSR 00001F80  FZ 0 DZ 0  Err  0 0 0 0 0 0
                Rnd NEAR   Mask 1 1 1 1 1 1
```

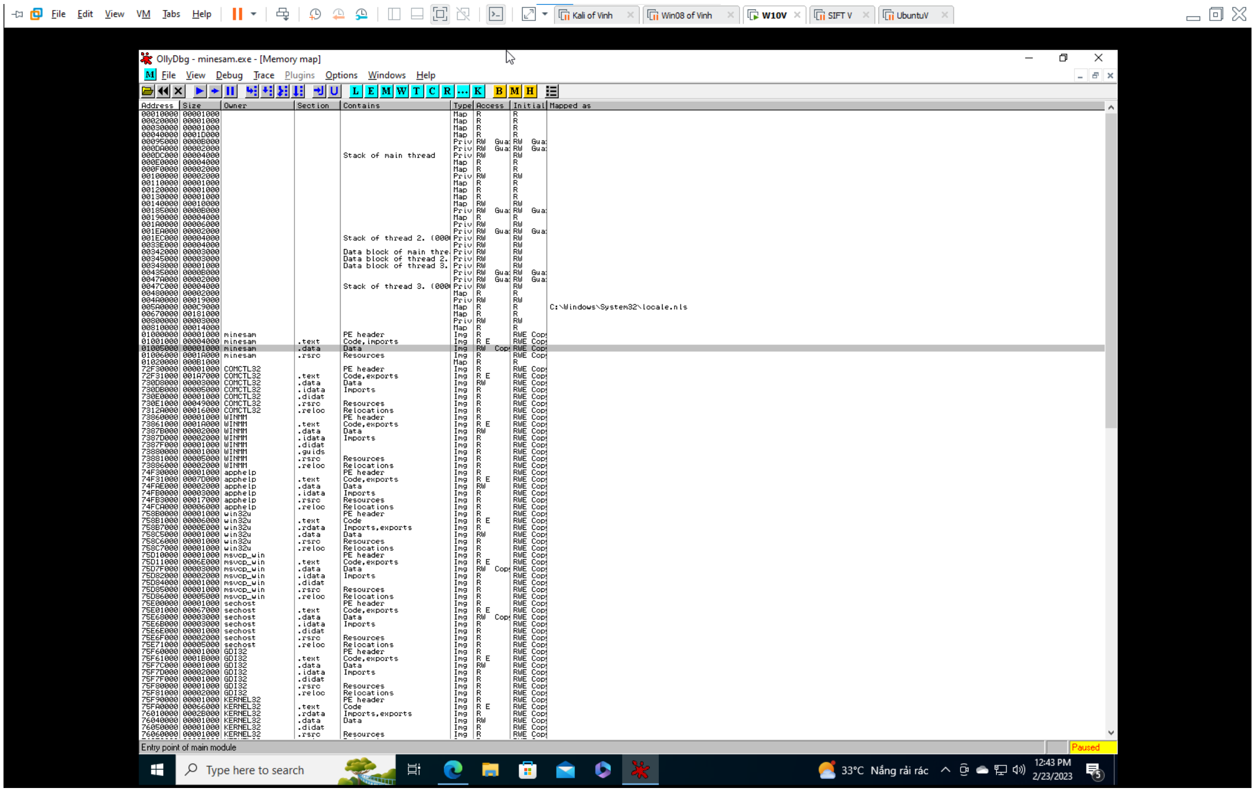
```
Address   Hex dump
01005000  1B 00 00 00 8F 00 00 00 8D 00 00 00 8E 00 00 00
01005010  0A 00 00 00 09 00 00 00 09 00 00 00 28 00 00 00
01005020  10 00 00 00 10 00 00 00 63 00 00 00 10 00 00 00
01005030  1E 00 00 00 58 00 59 00 5A 00 5A 00 59 00 00 00
01005040  8D 00 00 00 E8 03 00 00 8E 00 00 00 E9 03 00 00
01005050  8F 00 00 00 EA 03 00 00 70 00 00 00 E8 03 00 00
01005060  71 00 00 00 E9 03 00 00 6F 00 00 00 EA 03 00 00
01005070  00 00 00 00 00 00 00 00 C3 02 00 00 EB 03 00 00
01005080  C4 02 00 00 EC 03 00 00 C5 02 00 00 EC 03 00 00
01005090  C6 02 00 00 EC 03 00 00 BD 02 00 00 EC 03 00 00
010050A0  BF 02 00 00 EC 03 00 00 C1 02 00 00 EC 03 00 00
010050B0  BE 02 00 00 EC 03 00 00 C0 02 00 00 EC 03 00 00
010050C0  C2 02 00 00 EC 03 00 00 00 00 00 00 00 00 00 00
010050D0  28 13 00 01 1C 13 00 01 0C 13 00 01 00 13 00 01
010050E0  F4 12 00 01 E8 12 00 01 DC 12 00 01 D0 12 00 01
010050F0  C4 12 00 01 B8 12 00 01 AC 12 00 01 A0 12 00 01
01005100  94 12 00 01 88 12 00 01 7C 12 00 01 70 12 00 01
01005110  64 12 00 01 48 12 00 01 FF FF FF FF FF FF FF FF
01005120  65 00 6E 00 74 00 70 00 61 00 63 00 6B 00 2E 00   e n t p a c k .
01005130  69 00 6E 00 69 00 00 00 01 00 00 00 00 00 00 00   i n i
01005140  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005150  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005160  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005170  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005180  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005190  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
010051A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
010051B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
010051C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
010051D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
010051E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
010051F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005200  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005210  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005220  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
000DFF74  75FB00F9  RETURN to KERNEL32.BaseThreadInitThunk+19
000DFF78  0033F000
000DFF7C  75FB00E0  KERNEL32.BaseThreadInitThunk
000DFF80  000DFFDC
000DFF84  77A57BBE  RETURN to ntdll.77A57BBE
000DFF88  0033F000
000DFF8C  5E51A0FA
000DFF90  00000000
000DFF94  00000000
000DFF98  0033F000
000DFF9C  00000000
000DFFA0  00000000
000DFFA4  00000000
000DFFA8  00000000
000DFFAC  00000000
000DFFB0  00000000
000DFFB4  00000000
000DFFB8  00000000
000DFFBC  00000000
000DFFC0  00000000
000DFFC4  000DFF8C
000DFFC8  00000000
000DFFCC  000DFFE4  Pointer to next SEH record
000DFFD0  77A6AE60  SE handler
000DFFD4  29F39B86
000DFFD8  00000000
000DFFDC  000DFFEC
000DFFE0  77A57B8E  RETURN from ntdll.77A57B8F to ntdll.77A57B8E
000DFFE4  FFFFFFFF  End of SEH chain
000DFFE8  77A78D11  SE handler
000DFFEC  00000000
000DFFF0  00000000
000DFFF4  01003E21  minesam.<ModuleEntryPoint>
000DFFF8  0033F000
000DFFFC  00000000
```

Entry point of main module

Paused

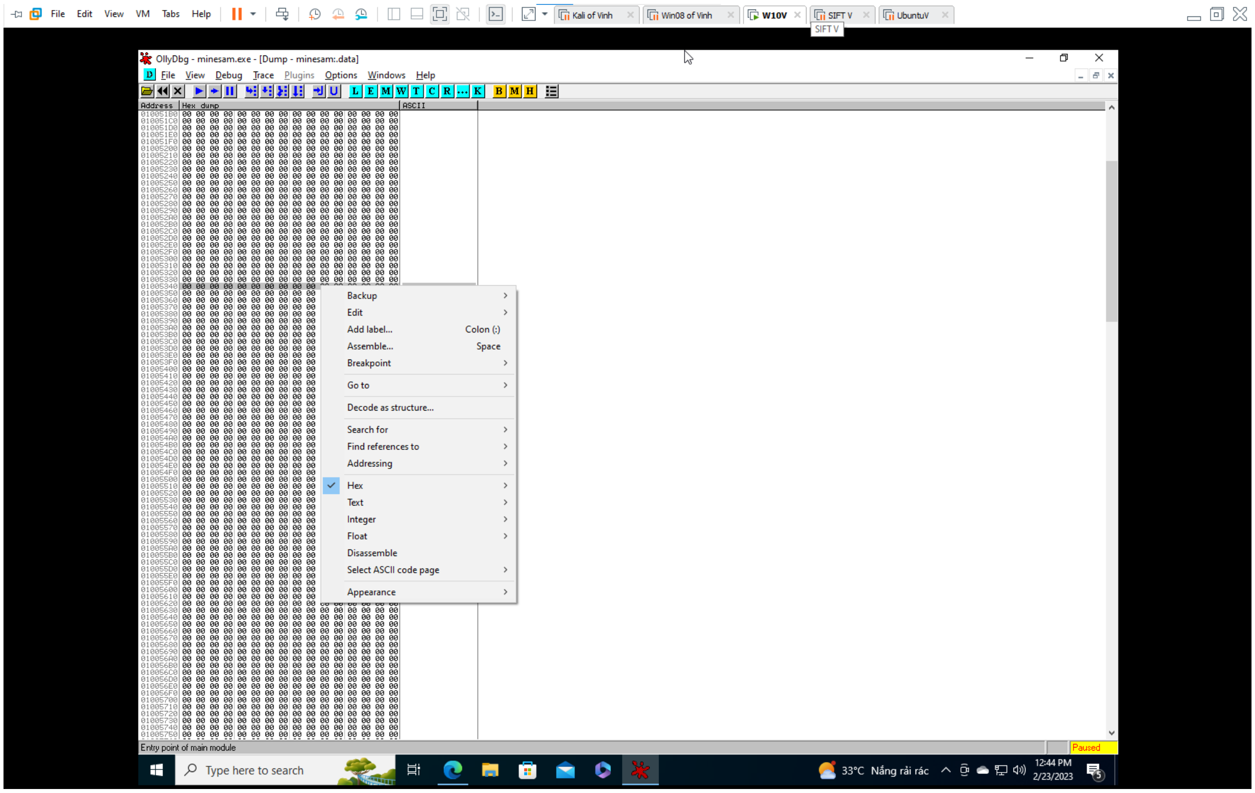
Type here to search
33°C Nắng rải rác
12:42 PM
2/23/2023


File Edit View VM Tabs Help
Kali of Vinh
Win08 of Vinh
W10V
SIFT V
UbuntuV
OllyDbg - minesam.exe - [Memory map]
M File View Debug Trace Plugins Options Windows Help
L E M W T C R ... K B M H

```
Address  Size     Owner       Section  Contains                 Type Access    Initial   Mapped as
00010000 00001000                                               Map  R         R
00020000 00001000                                               Map  R         R
00030000 00001000                                               Map  R         R
00040000 0001D000                                               Map  R         R
00095000 0000B000                                               Priv RW   Gua: RW   Gua:
000DA000 00002000                                               Priv RW   Gua: RW   Gua:
000DC000 00004000                      Stack of main thread     Priv RW        RW
000E0000 00004000                                               Map  R         R
000F0000 00002000                                               Map  R         R
00100000 00002000                                               Priv RW        RW
00110000 00001000                                               Map  R         R
00120000 00001000                                               Map  R         R
00130000 00001000                                               Map  R         R
00140000 00010000                                               Map  RW        RW
00185000 0000B000                                               Priv RW   Gua: RW   Gua:
00190000 00004000                                               Map  R         R
001A0000 00006000                                               Priv RW        RW
001EA000 00002000                                               Priv RW   Gua: RW   Gua:
001EC000 00004000                      Stack of thread 2. (000( Priv RW        RW
0033E000 00004000                                               Priv RW        RW
00342000 00003000                      Data block of main thre  Priv RW        RW
00345000 00003000                      Data block of thread 2.  Priv RW        RW
00348000 00001000                      Data block of thread 3.  Priv RW        RW
00435000 0000B000                                               Priv RW   Gua: RW   Gua:
0047A000 00002000                                               Priv RW   Gua: RW   Gua:
0047C000 00004000                      Stack of thread 3. (000( Priv RW        RW
00480000 00002000                                               Map  R         R
004A0000 00019000                                               Priv RW        RW
005A0000 000C9000                                               Map  R         R         C:\Windows\System32\locale.nls
00670000 00181000                                               Map  R         R
00800000 00003000                                               Priv RW        RW
00810000 00014000                                               Map  R         R
01000000 00001000 minesam              PE header                Img  R         RWE  Cop:
01001000 00004000 minesam     .text    Code,imports             Img  R E       RWE  Cop:
01005000 00001000 minesam     .data    Data                     Img  RW   Cop: RWE  Cop:
01006000 0001A000 minesam     .rsrc    Resources                Img  R         RWE  Cop:
01020000 000B1000                                               Map  R         R
72F30000 00001000 COMCTL32             PE header                Img  R         RWE  Cop:
72F31000 001A7000 COMCTL32    .text    Code,exports             Img  R E       RWE  Cop:
730D8000 00003000 COMCTL32    .data    Data                     Img  RW        RWE  Cop:
730DB000 00005000 COMCTL32    .idata   Imports                  Img  R         RWE  Cop:
730E0000 00001000 COMCTL32    .didat                            Img  R         RWE  Cop:
730E1000 00049000 COMCTL32    .rsrc    Resources                Img  R         RWE  Cop:
7312A000 00016000 COMCTL32    .reloc   Relocations              Img  R         RWE  Cop:
73860000 00001000 WINMM                PE header                Img  R         RWE  Cop:
73861000 0001A000 WINMM       .text    Code,exports             Img  R E       RWE  Cop:
7387B000 00002000 WINMM       .data    Data                     Img  RW        RWE  Cop:
7387D000 00002000 WINMM       .idata   Imports                  Img  R         RWE  Cop:
7387F000 00001000 WINMM       .didat                            Img  R         RWE  Cop:
73880000 00001000 WINMM       .guids                            Img  R         RWE  Cop:
73881000 00005000 WINMM       .rsrc    Resources                Img  R         RWE  Cop:
73886000 00002000 WINMM       .reloc   Relocations              Img  R         RWE  Cop:
74F30000 00001000 apphelp              PE header                Img  R         RWE  Cop:
74F31000 0007D000 apphelp     .text    Code,exports             Img  R E       RWE  Cop:
74FAE000 00002000 apphelp     .data    Data                     Img  RW        RWE  Cop:
74FB0000 00003000 apphelp     .idata   Imports                  Img  R         RWE  Cop:
74FB3000 00017000 apphelp     .rsrc    Resources                Img  R         RWE  Cop:
74FCA000 00006000 apphelp     .reloc   Relocations              Img  R         RWE  Cop:
758B0000 00001000 win32u               PE header                Img  R         RWE  Cop:
758B1000 00006000 win32u      .text    Code                     Img  R E       RWE  Cop:
758B7000 0000E000 win32u      .rdata   Imports,exports          Img  R         RWE  Cop:
758C5000 00001000 win32u      .data    Data                     Img  RW        RWE  Cop:
758C6000 00001000 win32u      .rsrc    Resources                Img  R         RWE  Cop:
758C7000 00001000 win32u      .reloc   Relocations              Img  R         RWE  Cop:
75D10000 00001000 msvcp_win            PE header                Img  R         RWE  Cop:
75D11000 0006E000 msvcp_win   .text    Code,exports             Img  R E       RWE  Cop:
75D7F000 00003000 msvcp_win   .data    Data                     Img  RW   Cop: RWE  Cop:
75D82000 00002000 msvcp_win   .idata   Imports                  Img  R         RWE  Cop:
75D84000 00001000 msvcp_win   .didat                            Img  R         RWE  Cop:
75D85000 00001000 msvcp_win   .rsrc    Resources                Img  R         RWE  Cop:
75D86000 00005000 msvcp_win   .reloc   Relocations              Img  R         RWE  Cop:
75E00000 00001000 sechost              PE header                Img  R         RWE  Cop:
75E01000 00067000 sechost     .text    Code,exports             Img  R E       RWE  Cop:
75E68000 00003000 sechost     .data    Data                     Img  RW   Cop: RWE  Cop:
75E6B000 00003000 sechost     .idata   Imports                  Img  R         RWE  Cop:
75E6E000 00001000 sechost     .didat                            Img  R         RWE  Cop:
75E6F000 00002000 sechost     .rsrc    Resources                Img  R         RWE  Cop:
75E71000 00005000 sechost     .reloc   Relocations              Img  R         RWE  Cop:
75F60000 00001000 GDI32                PE header                Img  R         RWE  Cop:
75F61000 0001B000 GDI32       .text    Code,exports             Img  R E       RWE  Cop:
75F7C000 00001000 GDI32       .data    Data                     Img  RW        RWE  Cop:
75F7D000 00002000 GDI32       .idata   Imports                  Img  R         RWE  Cop:
75F7F000 00001000 GDI32       .didat                            Img  R         RWE  Cop:
75F80000 00001000 GDI32       .rsrc    Resources                Img  R         RWE  Cop:
75F81000 00002000 GDI32       .reloc   Relocations              Img  R         RWE  Cop:
75F90000 00001000 KERNEL32             PE header                Img  R         RWE  Cop:
75FA0000 00066000 KERNEL32    .text    Code                     Img  R E       RWE  Cop:
76010000 0002B000 KERNEL32    .rdata   Imports,exports          Img  R         RWE  Cop:
76040000 00001000 KERNEL32    .data    Data                     Img  RW        RWE  Cop:
76050000 00001000 KERNEL32    .didat                            Img  R         RWE  Cop:
76060000 00001000 KERNEL32    .rsrc    Resources                Img  R         RWE  Cop:
```

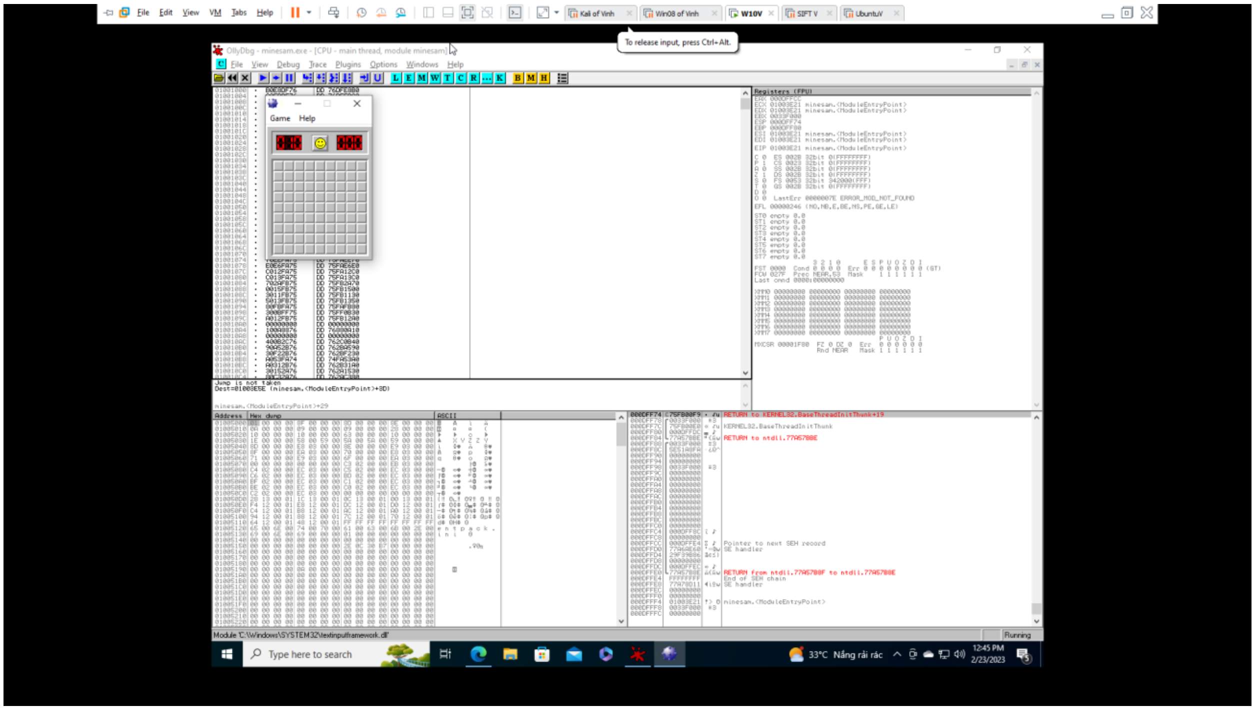
Entry point of main module

Paused



```
Address  Hex dump                                         ASCII
010051B0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
010051C0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
010051D0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
010051E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
010051F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005200 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005210 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005220 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005230 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005240 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005250 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005260 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005270 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005280 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005290 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
010052A0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
010052B0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
010052C0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
010052D0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
010052E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
010052F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005300 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005310 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005320 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005330 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005340 00 00 00 00 00 00 00 00 00 00
01005350 00 00 00 00 00 00 00 00 00 00
01005360 00 00 00 00 00 00 00 00 00 00
01005370 00 00 00 00 00 00 00 00 00 00
01005380 00 00 00 00 00 00 00 00 00 00
01005390 00 00 00 00 00 00 00 00 00 00
010053A0 00 00 00 00 00 00 00 00 00 00
010053B0 00 00 00 00 00 00 00 00 00 00
010053C0 00 00 00 00 00 00 00 00 00 00
010053D0 00 00 00 00 00 00 00 00 00 00
010053E0 00 00 00 00 00 00 00 00 00 00
010053F0 00 00 00 00 00 00 00 00 00 00
01005400 00 00 00 00 00 00 00 00 00 00
01005410 00 00 00 00 00 00 00 00 00 00
01005420 00 00 00 00 00 00 00 00 00 00
01005430 00 00 00 00 00 00 00 00 00 00
01005440 00 00 00 00 00 00 00 00 00 00
01005450 00 00 00 00 00 00 00 00 00 00
01005460 00 00 00 00 00 00 00 00 00 00
01005470 00 00 00 00 00 00 00 00 00 00
01005480 00 00 00 00 00 00 00 00 00 00
01005490 00 00 00 00 00 00 00 00 00 00
010054A0 00 00 00 00 00 00 00 00 00 00
010054B0 00 00 00 00 00 00 00 00 00 00
010054C0 00 00 00 00 00 00 00 00 00 00
010054D0 00 00 00 00 00 00 00 00 00 00
010054E0 00 00 00 00 00 00 00 00 00 00
010054F0 00 00 00 00 00 00 00 00 00 00
01005500 00 00 00 00 00 00 00 00 00 00
01005510 00 00 00 00 00 00 00 00 00 00
01005520 00 00 00 00 00 00 00 00 00 00
01005530 00 00 00 00 00 00 00 00 00 00
01005540 00 00 00 00 00 00 00 00 00 00
01005550 00 00 00 00 00 00 00 00 00 00
01005560 00 00 00 00 00 00 00 00 00 00
01005570 00 00 00 00 00 00 00 00 00 00
01005580 00 00 00 00 00 00 00 00 00 00
01005590 00 00 00 00 00 00 00 00 00 00
010055A0 00 00 00 00 00 00 00 00 00 00
010055B0 00 00 00 00 00 00 00 00 00 00
010055C0 00 00 00 00 00 00 00 00 00 00
010055D0 00 00 00 00 00 00 00 00 00 00
010055E0 00 00 00 00 00 00 00 00 00 00
010055F0 00 00 00 00 00 00 00 00 00 00
01005600 00 00 00 00 00 00 00 00 00 00
01005610 00 00 00 00 00 00 00 00 00 00
01005620 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005630 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005640 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005650 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005660 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005670 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005680 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005690 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
010056A0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
010056B0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
010056C0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
010056D0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
010056E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
010056F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005700 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005710 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005720 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005730 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005740 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01005750 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Entry point of main module



```
0100107C  .  C012FA75     DD 75FA12C0
01001080  .  C013FA75     DD 75FA13C0
01001084  .  702AFB75     DD 75FB2A70
01001088  .  0015FB75     DD 75FB1500
0100108C  .  3011FB75     DD 75FB1130
01001090  .  5013FB75     DD 75FB1350
01001094  .  00F0FA75     DD 75FAF000
01001098  .  3008FF75     DD 75FF0830
0100109C  .  A012FB75     DD 75FB12A0
010010A0  .  00000000     DD 00000000
010010A4  .  100A8876     DD 76880A10
010010A8  .  00000000     DD 00000000
010010AC  .  400B2C76     DD 762C0B40
010010B0  .  90A52B76     DD 762BA590
010010B4  .  30F22B76     DD 762BF230
010010B8  .  A053FA74     DD 74FA53A0
010010BC  .  A0012B76     DD 762B01A0
010010C0  .  30152A76     DD 762A1530
```

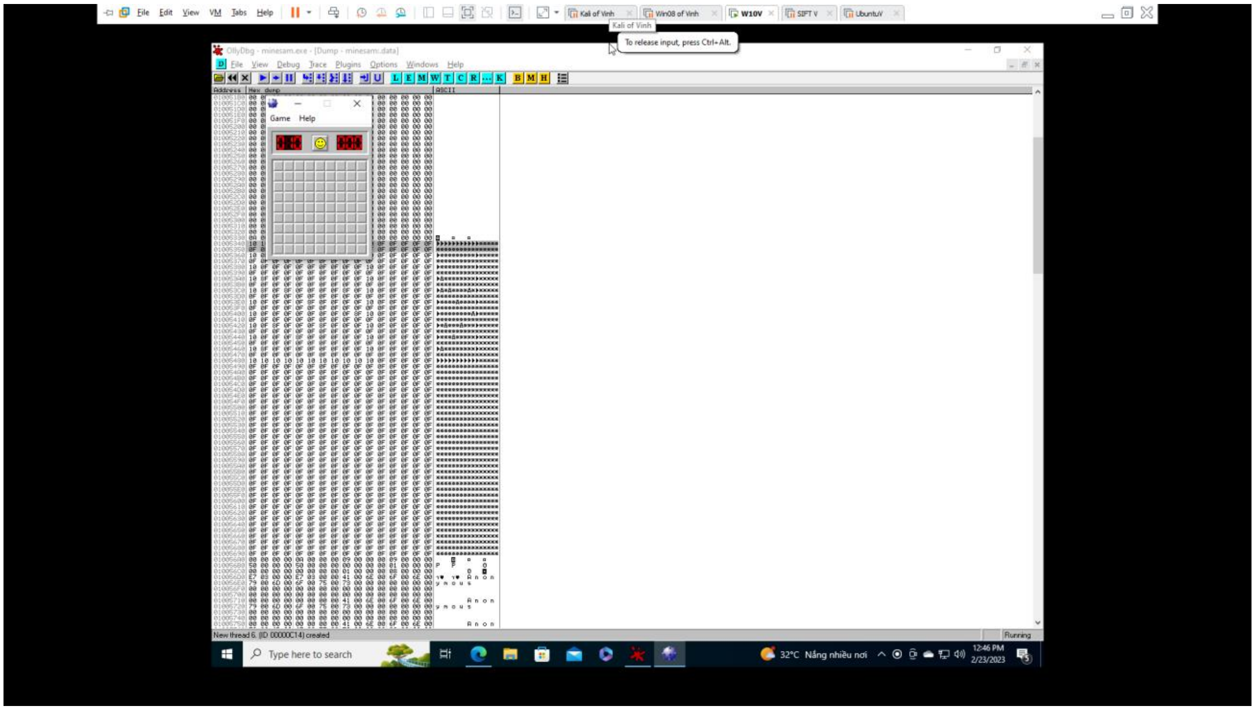
```
Jump is not taken
Dest=01003E5E (minesam.<ModuleEntryPoint>+8D)

minesam.<ModuleEntryPoint>+29
```

```
Address  Hex dump
01005000 01 00 00 00 8F 00 00 00 8D 00 00 00 8E 00 00 00
01005010 0A 00 00 00 09 00 00 00 09 00 00 00 28 00 00 00
01005020 10 00 00 00 10 00 00 00 63 00 00 00 10 00 00 00
01005030 1E 00 00 00 58 00 59 00 5A 00 5A 00 59 00 00 00
01005040 8D 00 00 00 E8 03 00 00 8E 00 00 00 E9 03 00 00
01005050 8F 00 00 00 EA 03 00 00 70 00 00 00 E8 03 00 00
01005060 71 00 00 00 E9 03 00 00 6F 00 00 00 EA 03 00 00
01005070 00 00 00 00 00 00 00 00 C3 02 00 00 EB 03 00 00
01005080 C4 02 00 00 EC 03 00 00 C5 02 00 00 EC 03 00 00
01005090 C6 02 00 00 EC 03 00 00 BD 02 00 00 EC 03 00 00
010050A0 BF 02 00 00 EC 03 00 00 C1 02 00 00 EC 03 00 00
010050B0 BE 02 00 00 EC 03 00 00 C0 02 00 00 EC 03 00 00
010050C0 C2 02 00 00 EC 03 00 00 00 00 00 00 00 00 00 00
010050D0 28 13 00 01 1C 13 00 01 0C 13 00 01 00 13 00 01
010050E0 F4 12 00 01 E8 12 00 01 DC 12 00 01 D0 12 00 01
010050F0 C4 12 00 01 B8 12 00 01 AC 12 00 01 A0 12 00 01
01005100 94 12 00 01 88 12 00 01 7C 12 00 01 70 12 00 01
01005110 64 12 00 01 48 12 00 01 FF FF FF FF FF FF FF FF
```

```
Registers (FPU)
EAX 000DFFCC
ECX 01003E21 minesam.<ModuleEntryPoint>
EDX 01003E21 minesam.<ModuleEntryPoint>
EBX 0033F000
ESP 000DFF74
EBP 000DFF80
ESI 01003E21 minesam.<ModuleEntryPoint>
EDI 01003E21 minesam.<ModuleEntryPoint>

EIP 01003E21 minesam.<ModuleEntryPoint>

C 0  ES 002B 32bit 0(FFFFFFFF)
P 1  CS 0023 32bit 0(FFFFFFFF)
A 0  SS 002B 32bit 0(FFFFFFFF)
Z 1  DS 002B 32bit 0(FFFFFFFF)
S 0  FS 0053 32bit 342000(FFF)
T 0  GS 002B 32bit 0(FFFFFFFF)
D 0
O 0  LastErr 0000007E ERROR_MOD_NOT_FOUND

EFL 00000246 (NO,NB,E,BE,NS,PE,GE,LE)

ST0 empty 0.0
ST1 empty 0.0
ST2 empty 0.0
ST3 empty 0.0
ST4 empty 0.0
ST5 empty 0.0
ST6 empty 0.0
ST7 empty 0.0
               3 2 1 0      E S P U O Z D I
FST 0000  Cond 0 0 0 0  Err 0 0 0 0 0 0 0 0 (GT)
FCW 027F  Prec NEAR,53  Mask    1 1 1 1 1 1
Last cmnd 0000:00000000

XMM0 00000000 00000000 00000000 00000000
XMM1 00000000 00000000 00000000 00000000
XMM2 00000000 00000000 00000000 00000000
XMM3 00000000 00000000 00000000 00000000
XMM4 00000000 00000000 00000000 00000000
XMM5 00000000 00000000 00000000 00000000
XMM6 00000000 00000000 00000000 00000000
XMM7 00000000 00000000 00000000 00000000
                                P U O Z D I
MXCSR 00001F80  FZ 0 DZ 0  Err  0 0 0 0 0 0
                Rnd NEAR   Mask 1 1 1 1 1 1
```

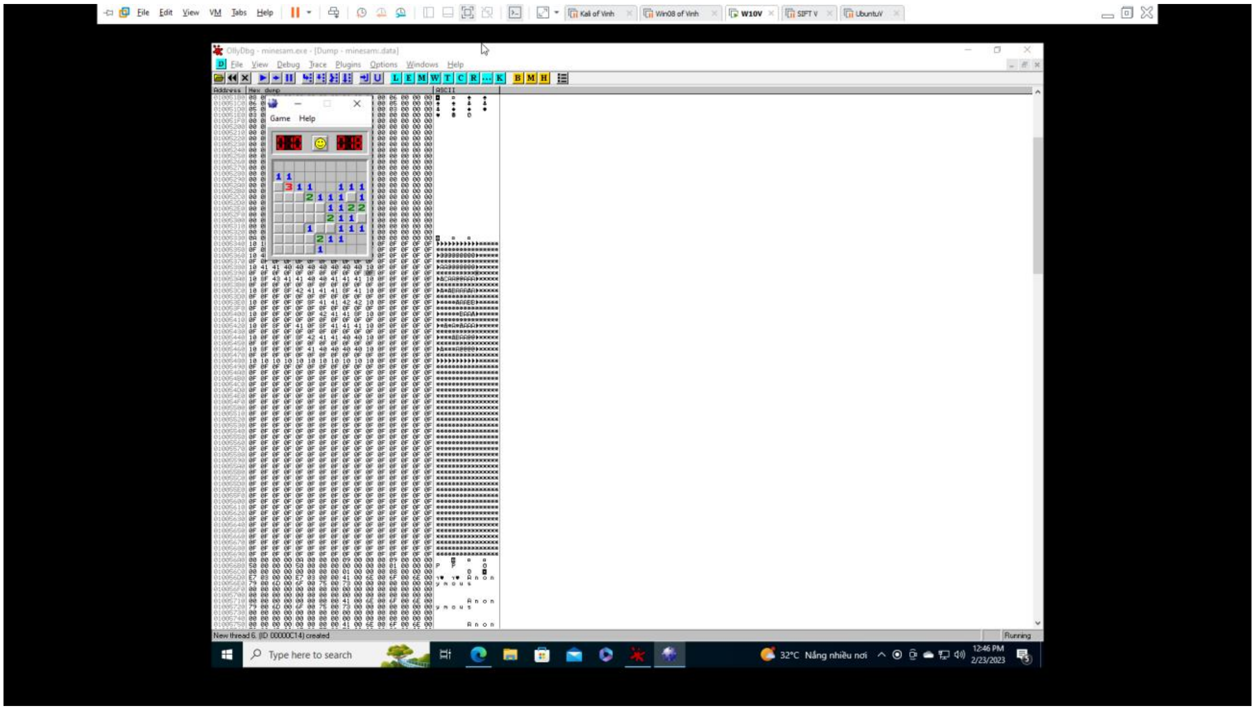
```
000DFF74  75FB00F9  RETURN to KERNEL32.BaseThreadInitThunk+19
000DFF78  0033F000
000DFF7C  75FB00E0  KERNEL32.BaseThreadInitThunk
000DFF80  000DFFDC
000DFF84  77A57B8E  RETURN to ntdll.77A57B8E
000DFF88  0033F000
000DFF8C  5E51A0FA
000DFF90  00000000
000DFF94  00000000
000DFF98  0033F000
000DFF9C  00000000
000DFFA0  00000000
000DFFA4  00000000
000DFFA8  00000000
000DFFAC  00000000
000DFFB0  00000000
000DFFB4  00000000
000DFFB8  00000000
000DFFBC  00000000
000DFFC0  00000000
000DFFC4  000DFF8C
000DFFC8  00000000
000DFFCC  000DFFE4  Pointer to next SEH record
000DFFD0  77A6AE60  SE handler
000DFFD4  29F39B86
000DFFD8  00000000
000DFFDC  000DFFEC
000DFFE0  77A57B8E  RETURN from ntdll.77A57B8F to ntdll.77A57B8E
000DFFE4  FFFFFFFF  End of SEH chain
000DFFE8  77A78D11  SE handler
000DFFEC  00000000
000DFFF0  00000000
000DFFF4  01003E21  minesam.<ModuleEntryPoint>
000DFFF8  0033F000
000DFFFC  00000000
```

Module C:\Windows\SYSTEM32\textinputframework.dll



To release input, press Ctrl+Alt.

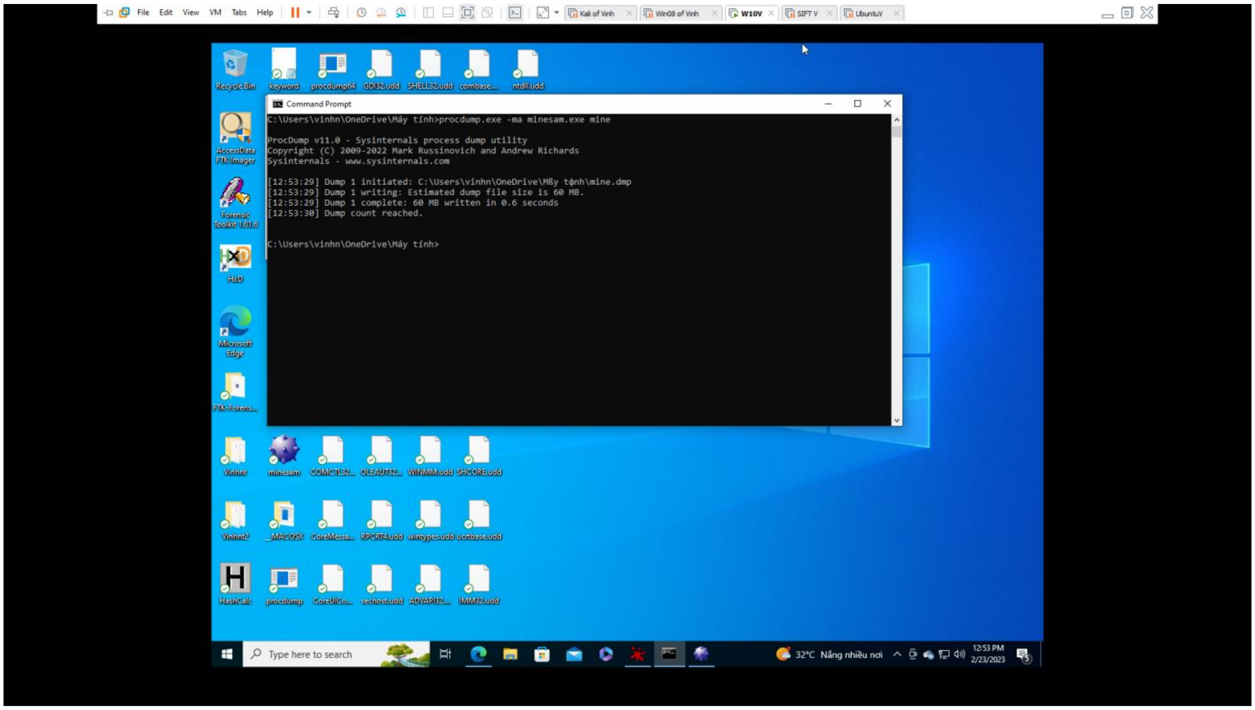
New thread 6. (ID 00000C14) created



New thread 6. (ID 00000C14) created

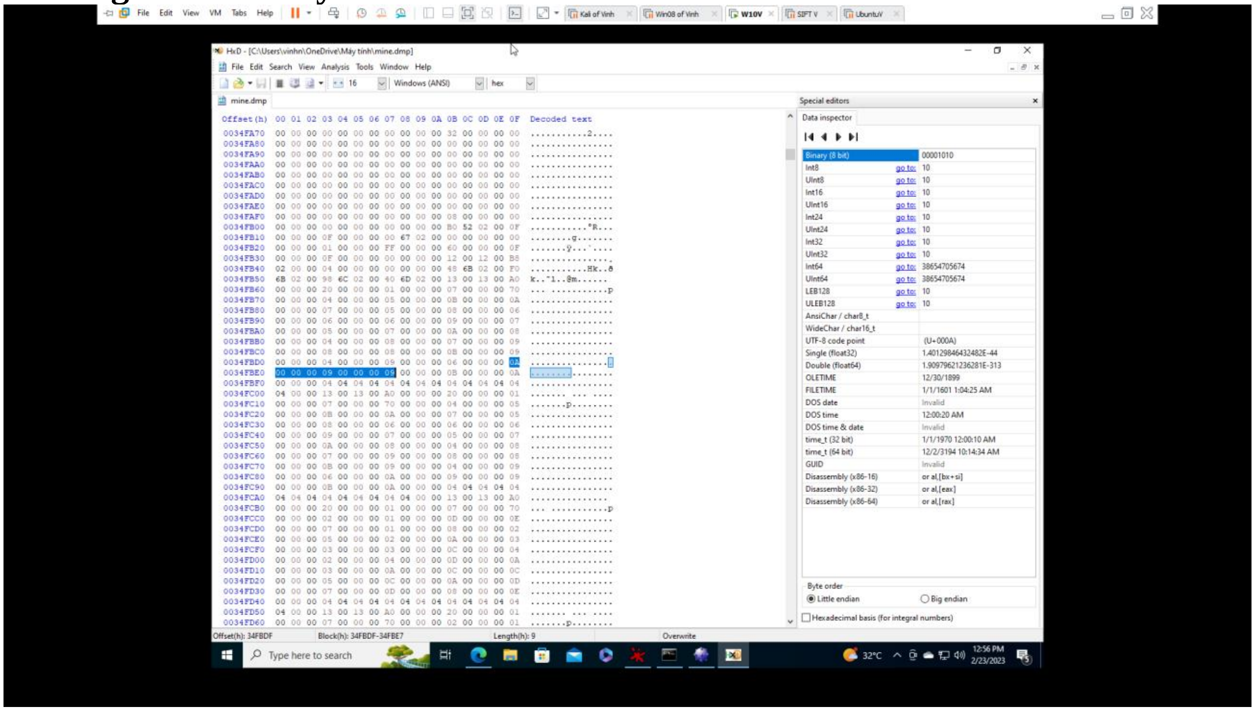
Getting Procdump

Capturing Process Memory



```
C:\Users\vinhn\OneDrive\Máy tính>procdump.exe -ma minesam.exe mine

ProcDump v11.0 - Sysinternals process dump utility
Copyright (C) 2009-2022 Mark Russinovich and Andrew Richards
Sysinternals - www.sysinternals.com

[12:53:29] Dump 1 initiated: C:\Users\vinhn\OneDrive\Mßy tọnh\mine.dmp
[12:53:29] Dump 1 writing: Estimated dump file size is 60 MB.
[12:53:29] Dump 1 complete: 60 MB written in 0.6 seconds
[12:53:30] Dump count reached.


C:\Users\vinhn\OneDrive\Máy tính>
```

Viewing the Memory with HxD



```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
0034FA70  00 00 00 00 00 00 00 00 00 00 00 32 00 00 00 00  ...........2....
0034FA80  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0034FA90  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0034FAA0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0034FAB0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0034FAC0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0034FAD0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0034FAE0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0034FAF0  00 00 00 00 00 00 00 00 00 00 00 08 00 00 00 00  ................
0034FB00  00 00 00 00 00 00 00 00 00 00 00 B0 52 02 00 0F  ...........°R...
0034FB10  00 00 00 0F 00 00 00 00 67 02 00 00 00 00 00 00  ........g.......
0034FB20  00 00 00 01 00 00 00 FF 00 00 00 60 00 00 00 0F  .......ÿ...`....
0034FB30  00 00 00 0F 00 00 00 00 00 00 00 12 00 12 00 B8  ...............¸
0034FB40  02 00 00 04 00 00 00 00 00 00 00 48 6B 02 00 F0  ...........Hk..ð
0034FB50  6B 02 00 98 6C 02 00 40 6D 02 00 13 00 13 00 A0  k..˜l..@m...... 
0034FB60  00 00 00 20 00 00 00 01 00 00 00 07 00 00 00 70  ... ...........p
0034FB70  00 00 00 04 00 00 00 05 00 00 00 0B 00 00 00 0A  ................
0034FB80  00 00 00 07 00 00 00 05 00 00 00 08 00 00 00 06  ................
0034FB90  00 00 00 06 00 00 00 06 00 00 00 09 00 00 00 07  ................
0034FBA0  00 00 00 05 00 00 00 07 00 00 00 0A 00 00 00 08  ................
0034FBB0  00 00 00 04 00 00 00 08 00 00 00 07 00 00 00 09  ................
0034FBC0  00 00 00 08 00 00 00 08 00 00 00 0B 00 00 00 09  ................
0034FBD0  00 00 00 04 00 00 00 09 00 00 00 06 00 00 00 0A  ................
0034FBE0  00 00 00 09 00 00 00 09 00 00 00 0B 00 00 00 0A  ................
0034FBF0  00 00 00 04 04 04 04 04 04 04 04 04 04 04 04 04  ................
0034FC00  04 00 00 13 00 13 00 A0 00 00 00 20 00 00 00 01  ....... ... ....
0034FC10  00 00 00 07 00 00 00 70 00 00 00 04 00 00 00 05  .......p........
0034FC20  00 00 00 0B 00 00 00 0A 00 00 00 07 00 00 00 05  ................
0034FC30  00 00 00 08 00 00 00 06 00 00 00 06 00 00 00 06  ................
0034FC40  00 00 00 09 00 00 00 07 00 00 00 05 00 00 00 07  ................
0034FC50  00 00 00 0A 00 00 00 08 00 00 00 04 00 00 00 08  ................
0034FC60  00 00 00 07 00 00 00 09 00 00 00 08 00 00 00 08  ................
0034FC70  00 00 00 0B 00 00 00 09 00 00 00 04 00 00 00 09  ................
0034FC80  00 00 00 06 00 00 00 0A 00 00 00 09 00 00 00 09  ................
0034FC90  00 00 00 0B 00 00 00 0A 00 00 00 04 04 04 04 04  ................
0034FCA0  04 04 04 04 04 04 04 04 04 00 00 13 00 13 00 A0  ............... 
0034FCB0  00 00 00 20 00 00 00 01 00 00 00 07 00 00 00 70  ... ...........p
0034FCC0  00 00 00 02 00 00 00 01 00 00 00 0D 00 00 00 0E  ................
0034FCD0  00 00 00 07 00 00 00 01 00 00 00 08 00 00 00 02  ................
0034FCE0  00 00 00 05 00 00 00 02 00 00 00 0A 00 00 00 03  ................
0034FCF0  00 00 00 03 00 00 00 03 00 00 00 0C 00 00 00 04  ................
0034FD00  00 00 00 02 00 00 00 04 00 00 00 0D 00 00 00 0A  ................
0034FD10  00 00 00 03 00 00 00 0A 00 00 00 0C 00 00 00 0C  ................
0034FD20  00 00 00 05 00 00 00 0C 00 00 00 0A 00 00 00 0D  ................
0034FD30  00 00 00 07 00 00 00 0D 00 00 00 08 00 00 00 0E  ................
0034FD40  00 00 00 04 04 04 04 04 04 04 04 04 04 04 04 04  ................
0034FD50  04 00 00 13 00 13 00 A0 00 00 00 20 00 00 00 01  ....... ... ....
0034FD60  00 00 00 07 00 00 00 70 00 00 00 02 00 00 00 01  .......p........

Offset(h): 34FBDF    Block(h): 34FBDF-34FBE7    Length(h): 9    Overwrite
```

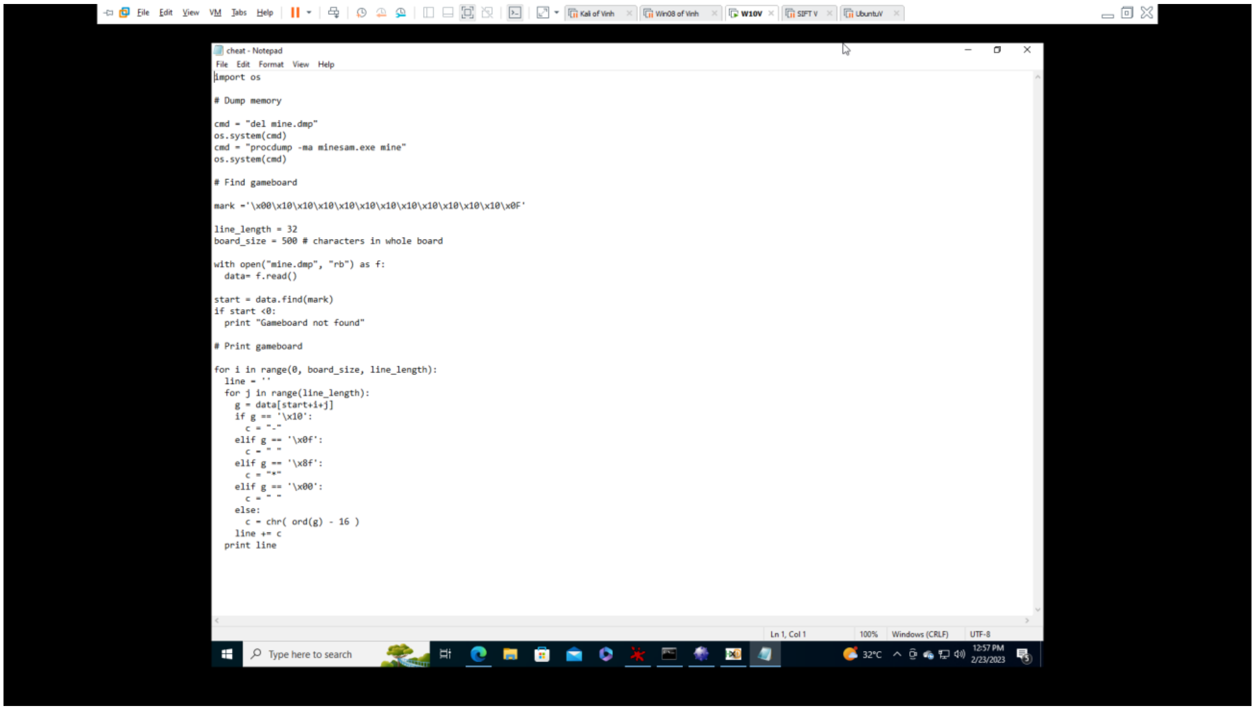
Binary (8 bit)		00001010
Int8	go to:	10
UInt8	go to:	10
Int16	go to:	10
UInt16	go to:	10
Int24	go to:	10
UInt24	go to:	10
Int32	go to:	10
UInt32	go to:	10
Int64	go to:	38654705674
UInt64	go to:	38654705674
LEB128	go to:	10
ULEB128	go to:	10
AnsiChar / char8_t		
WideChar / char16_t		
UTF-8 code point		(U+000A)
Single (float32)		1.40129846432482E-44
Double (float64)		1.90979621236281E-313
OLETIME		12/30/1899
FILETIME		1/1/1601 1:04:25 AM
DOS date		Invalid
DOS time		12:00:20 AM
DOS time & date		Invalid
time_t (32 bit)		1/1/1970 12:00:10 AM
time_t (64 bit)		12/2/3194 10:14:34 AM
GUID		Invalid
Disassembly (x86-16)		or al,[bx+si]
Disassembly (x86-32)		or al,[eax]
Disassembly (x86-64)		or al,[rax]

Byte order

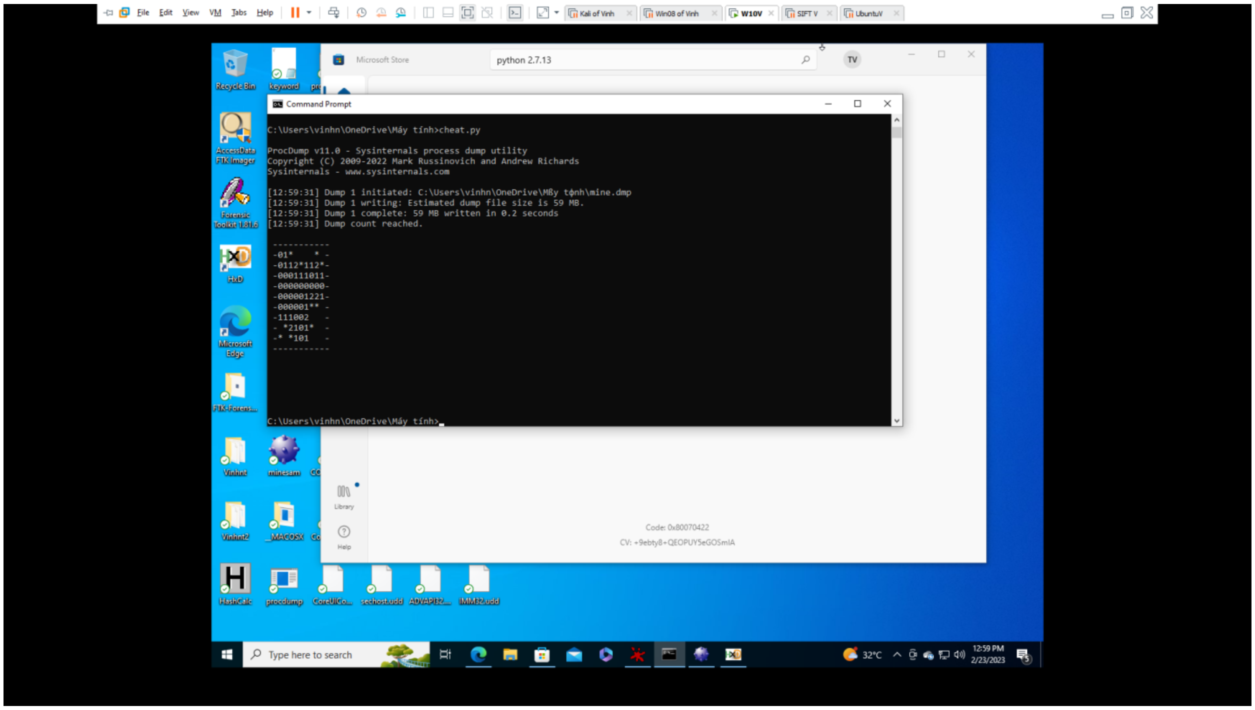
● Little endian ○ Big endian

□ Hexadecimal basis (for integral numbers)

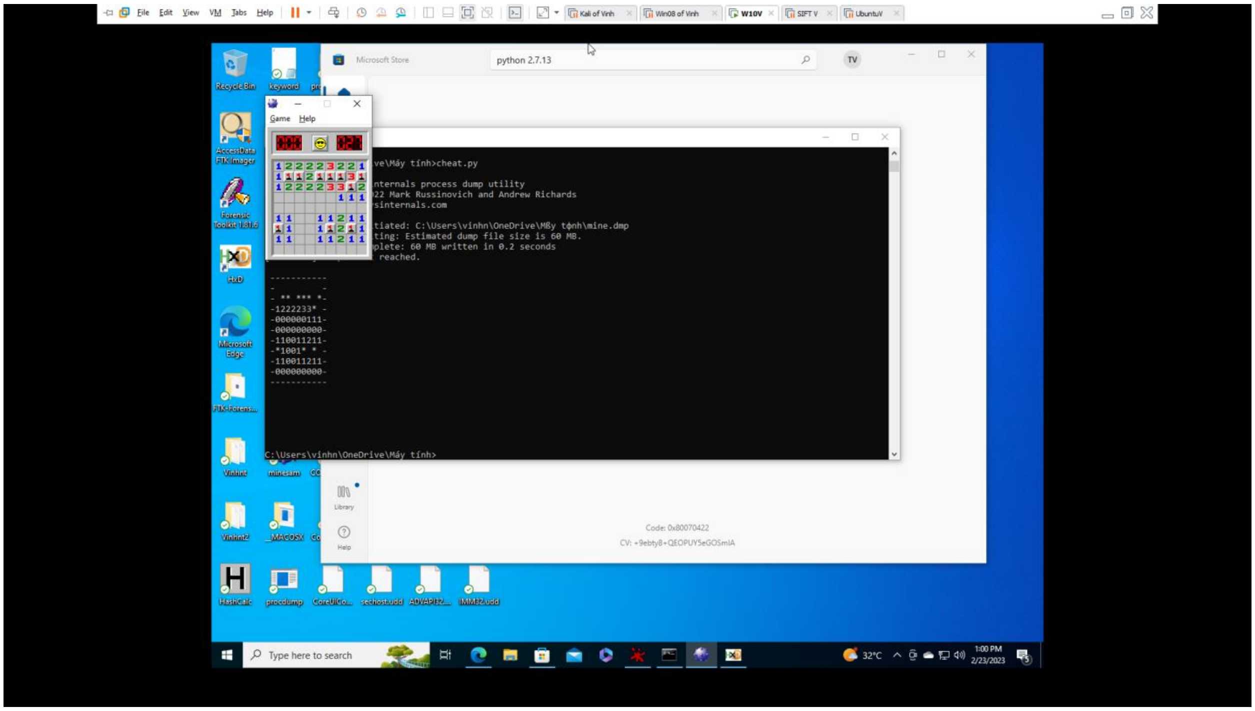
Creating a Python Script



```
import os

# Dump memory

cmd = "del mine.dmp"
os.system(cmd)
cmd = "procdump -ma minesam.exe mine"
os.system(cmd)

# Find gameboard

mark ='\x00\x10\x10\x10\x10\x10\x10\x10\x10\x10\x10\x10\x0F'

line_length = 32
board_size = 500 # characters in whole board

with open("mine.dmp", "rb") as f:
  data= f.read()

start = data.find(mark)
if start <0:
  print "Gameboard not found"

# Print gameboard

for i in range(0, board_size, line_length):
  line = ''
  for j in range(line_length):
    g = data[start+i+j]
    if g == '\x10':
      c = "-"
    elif g == '\x0f':
      c = " "
    elif g == '\x8f':
      c = "*"
    elif g == '\x00':
      c = " "
    else:
      c = chr( ord(g) - 16 )
    line += c
  print line
```

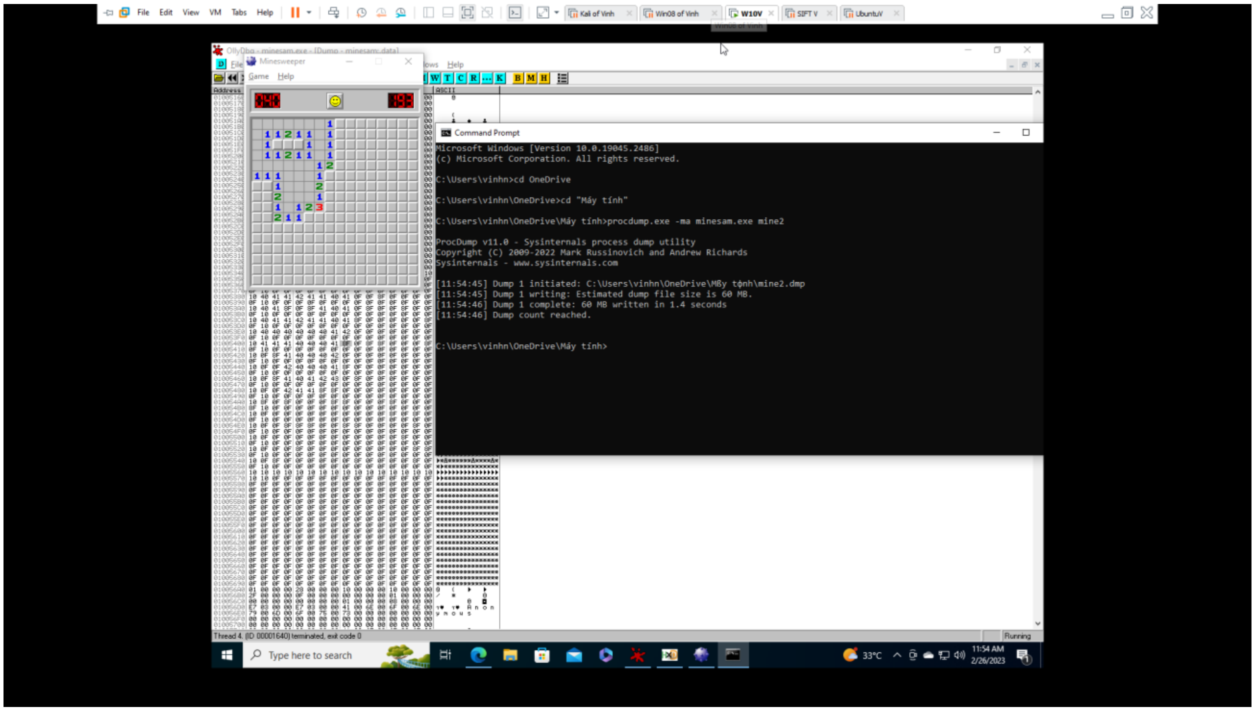


```
C:\Users\vinhn\OneDrive\Máy tính>cheat.py

ProcDump v11.0 - Sysinternals process dump utility
Copyright (C) 2009-2022 Mark Russinovich and Andrew Richards
Sysinternals - www.sysinternals.com

[12:59:31] Dump 1 initiated: C:\Users\vinhn\OneDrive\Mßy tộnh\mine.dmp
[12:59:31] Dump 1 writing: Estimated dump file size is 59 MB.
[12:59:31] Dump 1 complete: 59 MB written in 0.2 seconds
[12:59:31] Dump count reached.

 -----------
 -01*    * -
 -0112*112*-
 -000111011-
 -000000000-
 -000001221-
 -000001** -
 -111002   -
 -  *2101*  -
 -* *101   -
 -----------



C:\Users\vinhn\OneDrive\Máy tính>_
```

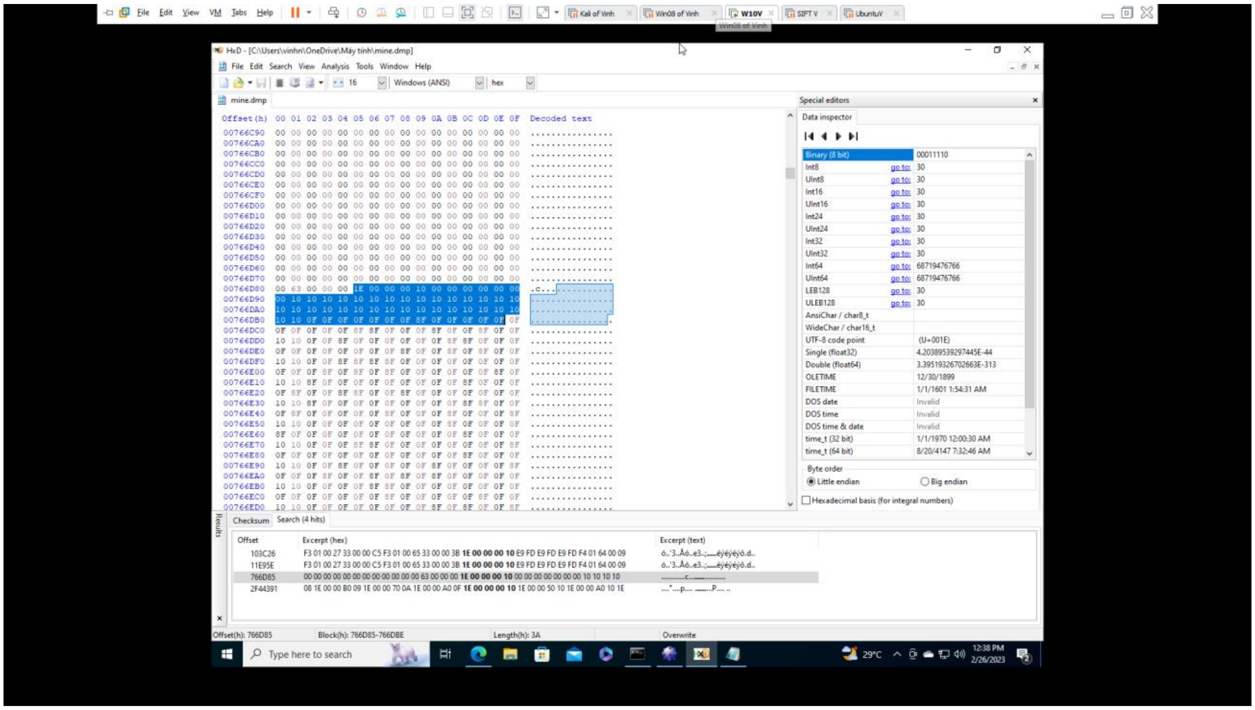


```
ve\Máy tính>cheat.py

nternals process dump utility
22 Mark Russinovich and Andrew Richards
sinternals.com

tiated: C:\Users\vinhn\OneDrive\Mßy tộnh\mine.dmp
ting: Estimated dump file size is 60 MB.
plete: 60 MB written in 0.2 seconds
 reached.

-----------
-         -
_ ** *** *_
-1222233* -
-000000111-
-000000000-
-110011211-
-*1001* * -
-110011211-
-000000000-
-----------


C:\Users\vinhn\OneDrive\Máy tính>
```

Intermediate Level



```
Microsoft Windows [Version 10.0.19045.2486]
(c) Microsoft Corporation. All rights reserved.

C:\Users\vinhn>cd OneDrive

C:\Users\vinhn\OneDrive>cd "Máy tính"

C:\Users\vinhn\OneDrive\Máy tính>procdump.exe -ma minesam.exe mine2

ProcDump v11.0 - Sysinternals process dump utility
Copyright (C) 2009-2022 Mark Russinovich and Andrew Richards
Sysinternals - www.sysinternals.com

[11:54:45] Dump 1 initiated: C:\Users\vinhn\OneDrive\Mßy tộnh\mine2.dmp
[11:54:45] Dump 1 writing: Estimated dump file size is 60 MB.
[11:54:46] Dump 1 complete: 60 MB written in 1.4 seconds
[11:54:46] Dump count reached.


C:\Users\vinhn\OneDrive\Máy tính>
```



HxD - [C:\Users\vinhn\OneDrive\Máy tính\mine.dmp]

```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00766C90  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00766CA0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00766CB0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00766CC0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00766CD0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00766CE0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00766CF0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00766D00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00766D10  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00766D20  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00766D30  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00766D40  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00766D50  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00766D60  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00766D70  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00766D80  00 63 00 00 00 1E 00 00 00 10 00 00 00 00 00 00  .c..............
00766D90  00 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10  ................
00766DA0  10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10  ................
00766DB0  10 10 0F 0F 0F 0F 0F 0F 0F 8F 0F 0F 0F 0F 0F 0F  ................
00766DC0  0F 0F 0F 0F 0F 8F 8F 0F 0F 0F 8F 0F 0F 8F 0F 0F  ................
00766DD0  10 10 0F 0F 8F 0F 0F 0F 0F 0F 0F 8F 8F 0F 0F 0F  ................
00766DE0  0F 0F 0F 0F 0F 0F 0F 0F 8F 0F 0F 8F 0F 8F 0F 0F  ................
00766DF0  10 10 0F 0F 8F 8F 8F 8F 0F 0F 0F 0F 0F 0F 0F 0F  ................
00766E00  0F 0F 0F 8F 0F 8F 0F 8F 0F 0F 0F 0F 0F 0F 8F 0F  ................
00766E10  10 10 8F 0F 0F 0F 0F 0F 0F 0F 0F 0F 8F 0F 0F 0F  ................
00766E20  0F 8F 0F 0F 8F 8F 0F 0F 8F 0F 0F 0F 0F 0F 0F 0F  ................
00766E30  10 10 8F 0F 0F 0F 0F 0F 0F 0F 0F 0F 8F 8F 0F 0F  ................
00766E40  0F 8F 0F 0F 0F 0F 0F 8F 0F 0F 0F 8F 0F 0F 0F 8F  ................
00766E50  10 10 0F 0F 0F 0F 0F 0F 0F 0F 0F 8F 0F 0F 0F 0F  ................
00766E60  8F 0F 0F 0F 0F 0F 0F 0F 0F 0F 0F 0F 8F 0F 0F 0F  ................
00766E70  10 10 0F 0F 0F 8F 8F 0F 0F 0F 0F 0F 0F 0F 0F 8F  ................
00766E80  0F 0F 0F 0F 0F 0F 0F 0F 0F 0F 0F 0F 8F 0F 0F 0F  ................
00766E90  10 10 0F 0F 8F 0F 0F 0F 0F 0F 8F 0F 0F 0F 0F 8F  ................
00766EA0  0F 0F 0F 8F 0F 0F 8F 0F 8F 0F 8F 0F 0F 0F 0F 0F  ................
00766EB0  10 10 0F 0F 0F 0F 8F 8F 0F 0F 0F 0F 0F 0F 0F 0F  ................
00766EC0  0F 0F 0F 0F 0F 0F 0F 8F 0F 0F 0F 0F 0F 8F 0F 0F  ................
00766ED0  10 10 0F 0F 0F 0F 0F 0F 0F 0F 8F 0F 8F 0F 0F 8F  ................
```

Special editors

Data inspector

Binary (8 bit)		00011110
Int8	go to:	30
UInt8	go to:	30
Int16	go to:	30
UInt16	go to:	30
Int24	go to:	30
UInt24	go to:	30
Int32	go to:	30
UInt32	go to:	30
Int64	go to:	68719476766
UInt64	go to:	68719476766
LEB128	go to:	30
ULEB128	go to:	30
AnsiChar / char8_t		
WideChar / char16_t		
UTF-8 code point		(U+001E)
Single (float32)		4.20389539297445E-44
Double (float64)		3.39519326702663E-313
OLETIME		12/30/1899
FILETIME		1/1/1601 1:54:31 AM
DOS date		Invalid
DOS time		Invalid
DOS time & date		Invalid
time_t (32 bit)		1/1/1970 12:00:30 AM
time_t (64 bit)		8/20/4147 7:32:46 AM

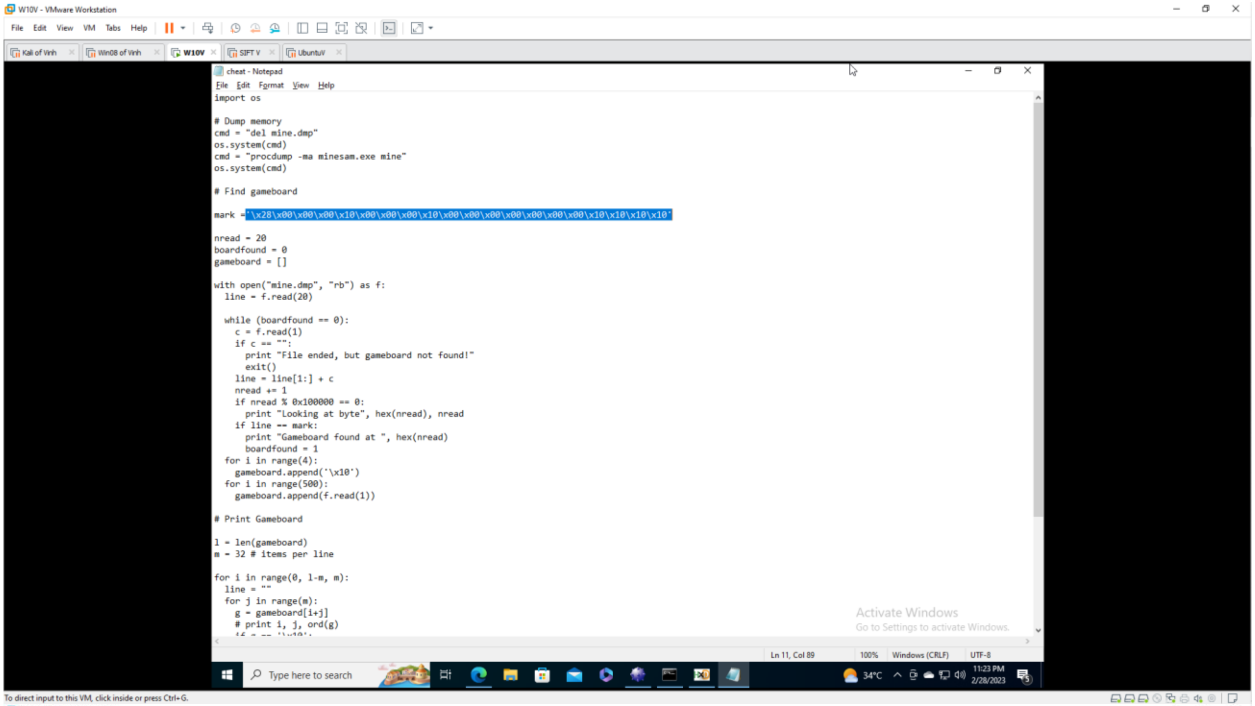
Byte order

● Little endian ○ Big endian

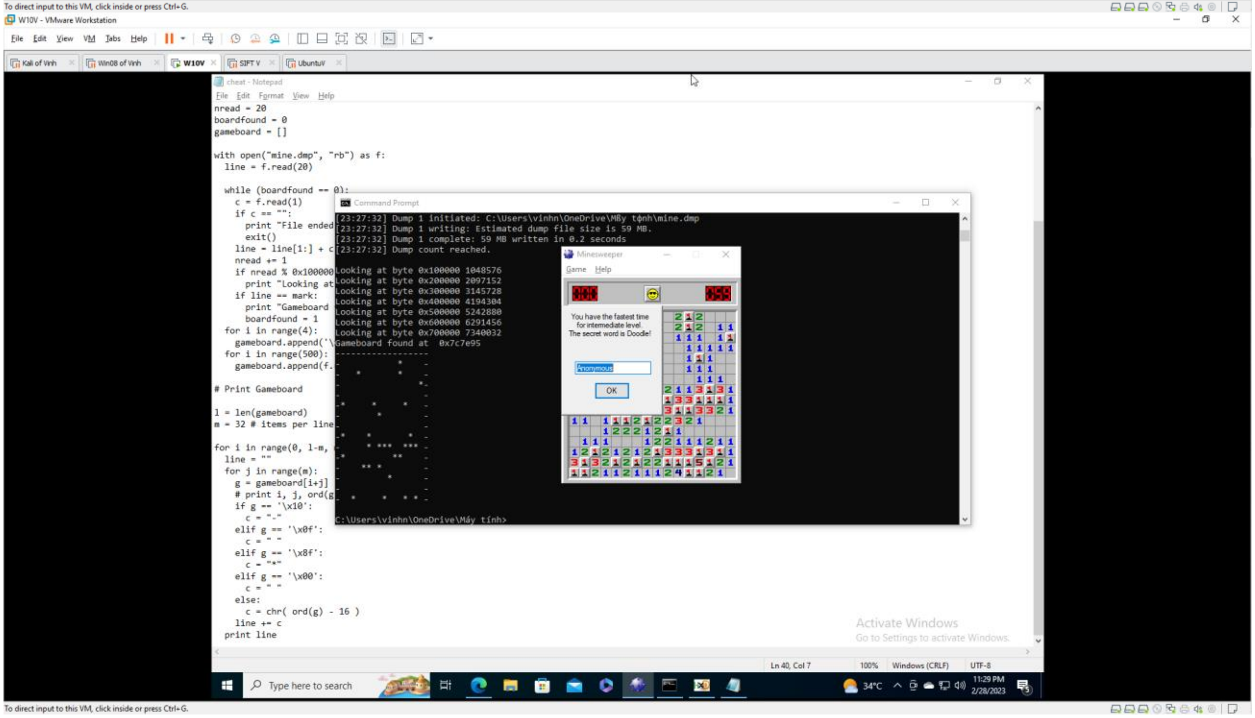
□ Hexadecimal basis (for integral numbers)

Offset	Excerpt (hex)	Excerpt (text)
103C26	F3 01 00 27 33 00 00 C5 F3 01 00 65 33 00 00 3B **1E 00 00 00 10** E9 FD E9 FD E9 FD F4 01 64 00 09	ó..'3..Åó..e3..;.....éýéýéýô.d..
11E95E	F3 01 00 27 33 00 00 C5 F3 01 00 65 33 00 00 3B **1E 00 00 00 10** E9 FD E9 FD E9 FD F4 01 64 00 09	ó..'3..Åó..e3..;.....éýéýéýô.d..
766D85	00 00 00 00 00 00 00 00 00 00 00 00 63 00 00 00 **1E 00 00 00 10** 00 00 00 00 00 00 00 10 10 10 10	............c...................
2F44391	08 1E 00 00 B0 09 1E 00 00 70 0A 1E 00 00 A0 0F **1E 00 00 00 10** 1E 00 00 50 10 1E 00 00 A0 10 1E	....°....p.... ....P.... ..

Offset(h): 766D85 Block(h): 766D85-766DBE Length(h): 3A Overwrite


W10V - VMware Workstation
File Edit View VM Tabs Help
Kali of Vinh
Win08 of Vinh
W10V
SIFT V
UbuntuV
cheat - Notepad
File Edit Format View Help
Activate Windows
Go to Settings to activate Windows.
Ln 11, Col 89
100%
Windows (CRLF)
UTF-8
Type here to search
34°C
11:23 PM
2/28/2023
To direct input to this VM, click inside or press Ctrl+G.

```
import os

# Dump memory
cmd = "del mine.dmp"
os.system(cmd)
cmd = "procdump -ma minesam.exe mine"
os.system(cmd)

# Find gameboard

mark ='\x28\x00\x00\x00\x10\x00\x00\x00\x10\x00\x00\x00\x00\x00\x00\x00\x10\x10\x10\x10'

nread = 20
boardfound = 0
gameboard = []

with open("mine.dmp", "rb") as f:
  line = f.read(20)

  while (boardfound == 0):
    c = f.read(1)
    if c == "":
      print "File ended, but gameboard not found!"
      exit()
    line = line[1:] + c
    nread += 1
    if nread % 0x100000 == 0:
      print "Looking at byte", hex(nread), nread
    if line == mark:
      print "Gameboard found at ", hex(nread)
      boardfound = 1
  for i in range(4):
    gameboard.append('\x10')
  for i in range(500):
    gameboard.append(f.read(1))

# Print Gameboard

l = len(gameboard)
m = 32 # items per line

for i in range(0, l-m, m):
  line = ""
  for j in range(m):
    g = gameboard[i+j]
    # print i, j, ord(g)
```



```
    if g == '\x10':
      c = "-"
    elif g == '\x0f':
      c = " "
    elif g == '\x8f':
      c = "*"
    elif g == '\x00':
      c = " "
    else:
      c = chr( ord(g) - 16 )
    line += c
  print line
```

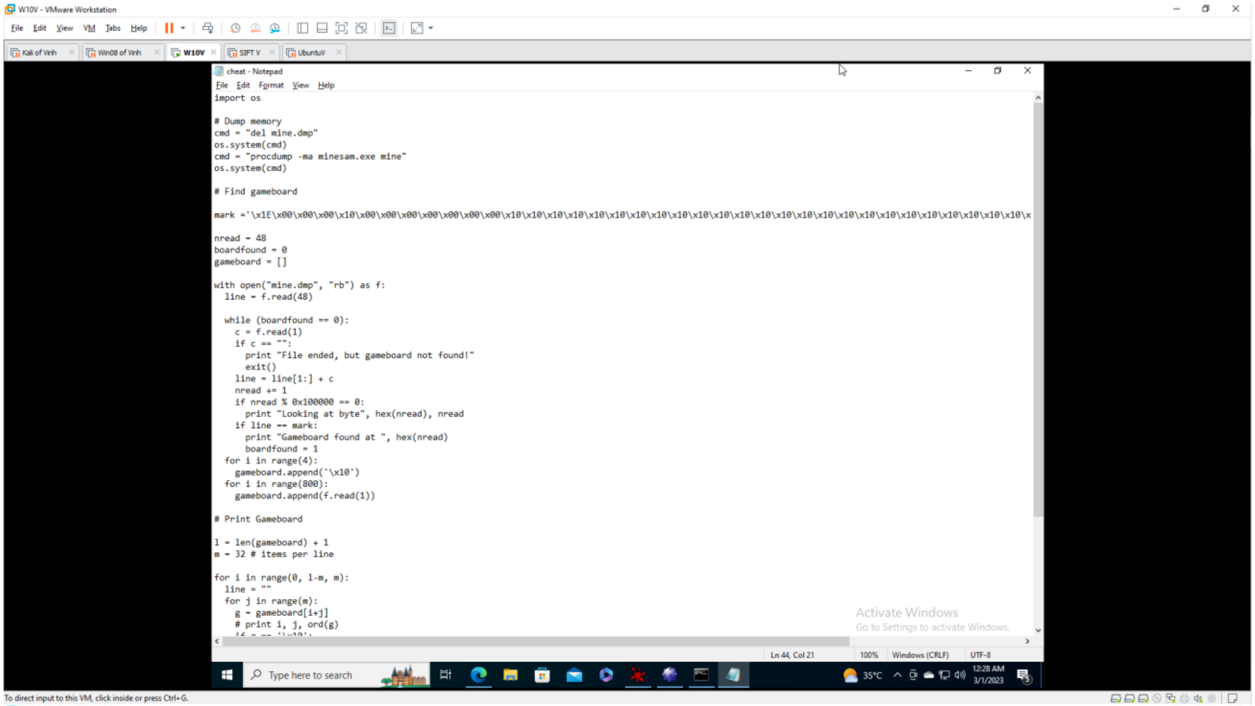
```
[23:27:32] Dump 1 initiated: C:\Users\vinhn\OneDrive\MBy tộnh\mine.dmp
[23:27:32] Dump 1 writing: Estimated dump file size is 59 MB.
[23:27:32] Dump 1 complete: 59 MB written in 0.2 seconds
[23:27:32] Dump count reached.

Looking at byte 0x100000 1048576
Looking at byte 0x200000 2097152
Looking at byte 0x300000 3145728
Looking at byte 0x400000 4194304
Looking at byte 0x500000 5242880
Looking at byte 0x600000 6291456
Looking at byte 0x700000 7340032
Gameboard found at  0x7c7e95
------------------
-         *     -
-   *     *     -
-              *-
-               -
-*    *     *   -
-      *        -
-               -
-*    *      *  -
-     * ***  ***-
-*       **     -
-    ** *       -
-       *       -
-               -
-  *     *   * *-

C:\Users\vinhn\OneDrive\Máy tính>
```

You have the fastest time for intermediate level. The secret word is Doodle!

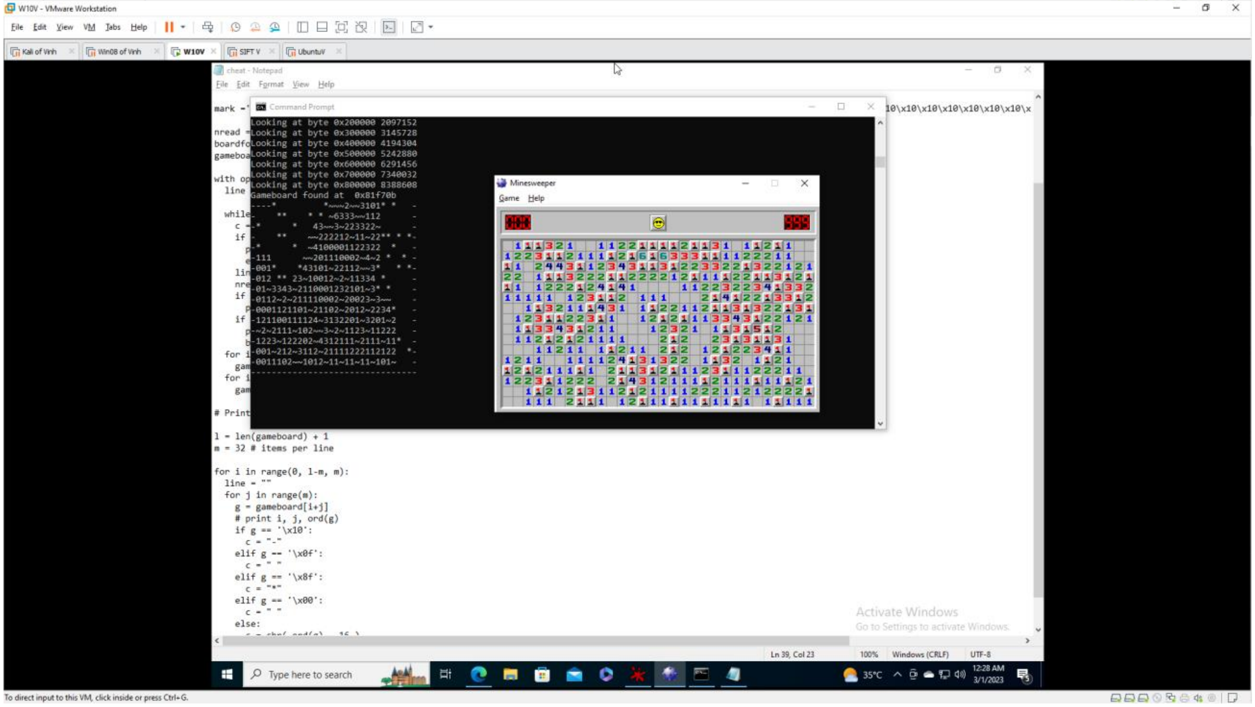
Anonymous

OK

Expert Level


W10V - VMware Workstation
File Edit View VM Tabs Help
Kali of Vinh
Win08 of Vinh
W10V
SIFT V
UbuntuV
cheat - Notepad
File Edit Format View Help
Activate Windows
Go to Settings to activate Windows.
Ln 44, Col 21
100%
Windows (CRLF)
UTF-8
Type here to search
35°C
12:28 AM
3/1/2023
To direct input to this VM, click inside or press Ctrl+G.

```
import os

# Dump memory
cmd = "del mine.dmp"
os.system(cmd)
cmd = "procdump -ma minesam.exe mine"
os.system(cmd)

# Find gameboard

mark ='\x1E\x00\x00\x00\x10\x00\x00\x00\x00\x00\x00\x00\x10\x10\x10\x10\x10\x10\x10\x10\x10\x10\x10\x10\x10\x10\x10\x10\x10\x10\x10\x10\x10\x10\x10\x10\x10\x

nread = 48
boardfound = 0
gameboard = []

with open("mine.dmp", "rb") as f:
  line = f.read(48)

  while (boardfound == 0):
    c = f.read(1)
    if c == "":
      print "File ended, but gameboard not found!"
      exit()
    line = line[1:] + c
    nread += 1
    if nread % 0x100000 == 0:
      print "Looking at byte", hex(nread), nread
    if line == mark:
      print "Gameboard found at ", hex(nread)
      boardfound = 1
  for i in range(4):
    gameboard.append('\x10')
  for i in range(800):
    gameboard.append(f.read(1))

# Print Gameboard

l = len(gameboard) + 1
m = 32 # items per line

for i in range(0, l-m, m):
  line = ""
  for j in range(m):
    g = gameboard[i+j]
    # print i, j, ord(g)
    if g == '\x10':
```



```
Looking at byte 0x200000 2097152
Looking at byte 0x300000 3145728
Looking at byte 0x400000 4194304
Looking at byte 0x500000 5242880
Looking at byte 0x600000 6291456
Looking at byte 0x700000 7340032
Looking at byte 0x800000 8388608
Gameboard found at  0x81f70b
----*         *~~~2~~3101* *   -
-    **    * * ~6333~~112      -
-*      *   43~~3~223322~      -
-    **    ~~222212~11~22** * *-
-*      *  ~4100001122322  *   -
-111      ~~201110002~4~2 *  * -
-001*    *43101~22112~~3*   * *-
-012 ** 23~10012~2~11334 *     -
-01~3343~2110001232101~3* *    -
-0112~2~211110002~20023~3~~    -
-0001121101~21102~2012~2234*   -
-121100111124~3132201~3201~2   -
-~2~2111~102~~3~2~1123~11222   -
-1223~122202~4312111~2111~11*  -
-001~212~3112~21111222112122  *-
-0011102~~1012~11~11~11~101~   -
--------------------------------
```

```
l = len(gameboard) + 1
m = 32 # items per line

for i in range(0, l-m, m):
  line = ""
  for j in range(m):
    g = gameboard[i+j]
    # print i, j, ord(g)
    if g == '\x10':
      c = "-"
    elif g == '\x0f':
      c = " "
    elif g == '\x8f':
      c = "*"
    elif g == '\x00':
      c = " "
    else:
      c = chr( ord(g) - 16 )
```