CSE 434S

Lab: Debugging

Overview

The goal of this lab is to practice debugging. Debugging assembly is a skill that requires continuous practice; the more you practice, the more you become familiar with assembly blocks, and with different debugging techniques.
You are free to choose whatever debugging tool you like. You can try a couple of different tools to see which one works best for you.

Before you start looking at the details, I want to remind you some of the rules of reverse engineering by revisiting some of the things we discussed in the first lecture:

- Don't get caught in details!
- You don't need to understand 100% of the code
- Focus on key features

Please analyze the first two samples in the lab's archive, which you can download from Canvas, and answer the following questions regarding the samples. You may want to use other tools in addition to a debugger, such as our basic static and dynamic analysis tools and IDA, to analyze the samples, and that is encouraged.

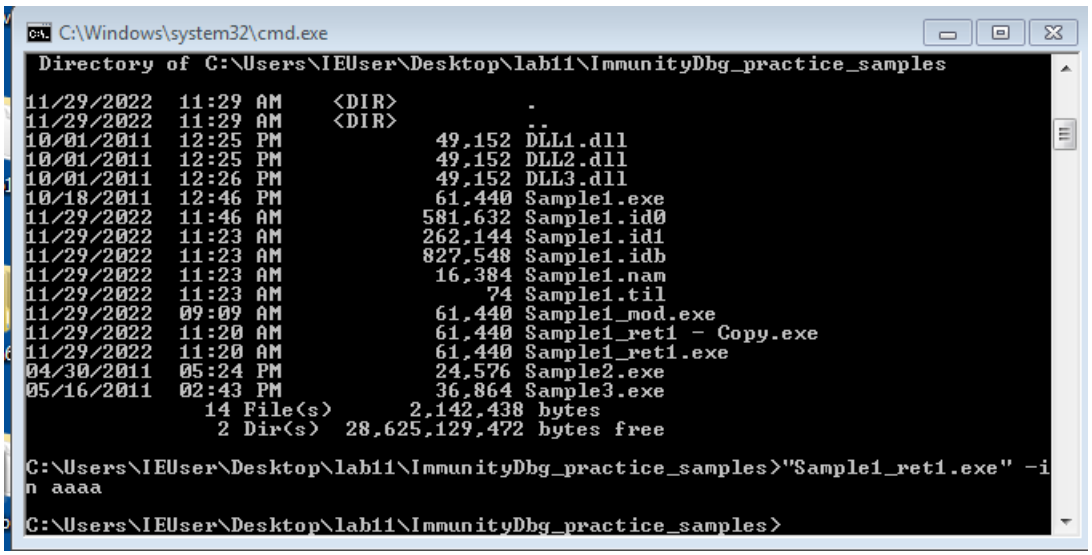
***** Remember to snapshot your VM and double-check that it is on an isolated network (e.g. an "internal network" in VirtualBox) before loading any sample in a debugger! *****

CSE 434S

Sample 1:

Q1-1. How can you get this malware to install itself?

Just run the program without arguments, and then the file had gone. I think it is deleted itself.
I tried again with arguments like '-in aaaa' after disabling password check function. But it seems like nothing happens and I am not sure it's installed or not.



```
C:\Windows\system32\cmd.exe
 Directory of C:\Users\IEUser\Desktop\lab11\ImmunityDbg_practice_samples

11/29/2022  11:29 AM    <DIR>          .
11/29/2022  11:29 AM    <DIR>          ..
10/01/2011  12:25 PM            49,152 DLL1.dll
10/01/2011  12:25 PM            49,152 DLL2.dll
10/01/2011  12:26 PM            49,152 DLL3.dll
10/18/2011  12:46 PM            61,440 Sample1.exe
11/29/2022  11:46 AM           581,632 Sample1.id0
11/29/2022  11:23 AM           262,144 Sample1.id1
11/29/2022  11:23 AM           827,548 Sample1.idb
11/29/2022  11:23 AM            16,384 Sample1.nam
11/29/2022  11:23 AM                74 Sample1.til
11/29/2022  09:09 AM            61,440 Sample1_mod.exe
11/29/2022  11:20 AM            61,440 Sample1_ret1 - Copy.exe
11/29/2022  11:20 AM            61,440 Sample1_ret1.exe
04/30/2011  05:24 PM            24,576 Sample2.exe
05/16/2011  02:43 PM            36,864 Sample3.exe
              14 File(s)      2,142,438 bytes
               2 Dir(s)  28,625,129,472 bytes free

C:\Users\IEUser\Desktop\lab11\ImmunityDbg_practice_samples>"Sample1_ret1.exe" -i
n aaaa

C:\Users\IEUser\Desktop\lab11\ImmunityDbg_practice_samples>
```

Q1-2. What are the command-line options for this program? What is the password requirement?

I can notice 4 arguments of this program. 00402AF0 is the address of a main function and if you go to the address, we can see the arguments operations.
-in
-re
-c
-cc

CSE 434S

```
Immunity Debugger - Sample1.exe - [CPU - main thread, module Sample1]
File View Debug Plugins ImmLib Options Window Help Jobs
l e m t w h c P k b z r ... s ?
02B3F  > 8B4D 0C        MOV ECX,DWORD PTR SS:[EBP+C]
02B42  . 8B51 04        MOV EDX,DWORD PTR DS:[ECX+4]
02B45  . 8995 E0E7FFFF  MOV DWORD PTR SS:[EBP-1820],EDX
02B4B  . 68 70C14000    PUSH Sample1.0040C170                    ASCII "-in"
02B50  . 8B85 E0E7FFFF  MOV EAX,DWORD PTR SS:[EBP-1820]
02B56  . 50             PUSH EAX
02B57  . E8 B30C0000    CALL Sample1.0040380F
02B5C  . 83C4 08        ADD ESP,8
02B5F  . 85C0           TEST EAX,EAX
02B61  . 75 64          JNZ SHORT Sample1.00402BC7
02B63  . 837D 08 03     CMP DWORD PTR SS:[EBP+8],3
02B67  . 75 31          JNZ SHORT Sample1.00402B9A
02B69  . 68 00040000    PUSH 400                                 Arg2 = 00000400
02B6E  . 8D8D FCFBFFFF  LEA ECX,DWORD PTR SS:[EBP-404]           Arg1
02B74  . 51             PUSH ECX                                 Sample1.004025B0
02B75  . E8 36FAFFFF    CALL Sample1.004025B0
02B7A  . 83C4 08        ADD ESP,8
02B7D  . 85C0           TEST EAX,EAX
02B7F  . 74 08          JE SHORT Sample1.00402B89
02B81  . 83C8 FF        OR EAX,FFFFFFFF
02B84  . E9 EF010000    JMP Sample1.00402D78
02B89  > 8D95 FCFBFFFF  LEA EDX,DWORD PTR SS:[EBP-404]
02B8F  . 52             PUSH EDX                                 Arg1
02B90  . E8 6BFAFFFF    CALL Sample1.00402600                    Sample1.00402600
02B95  . 83C4 04        ADD ESP,4
02B98  . EB 28          JMP SHORT Sample1.00402BC2
02B9A  > 837D 08 04     CMP DWORD PTR SS:[EBP+8],4
02B9E  . 75 1D          JNZ SHORT Sample1.00402BBD
02BA0  . 8B45 0C        MOV EAX,DWORD PTR SS:[EBP+C]
02BA3  . 8B48 08        MOV ECX,DWORD PTR DS:[EAX+8]
02BA6  . 898D F8FBFFFF  MOV DWORD PTR SS:[EBP-408],ECX
02BAC  . 8B95 F8FBFFFF  MOV EDX,DWORD PTR SS:[EBP-408]
02BB2  . 52             PUSH EDX                                 Arg1
02BB3  . E8 48FAFFFF    CALL Sample1.00402600                    Sample1.00402600
02BB8  . 83C4 04        ADD ESP,4
02BBB  . EB 05          JMP SHORT Sample1.00402BC2
02BBD  > E8 4EF8FFFF    CALL Sample1.00402410
02BC2  > E9 AF010000    JMP Sample1.00402D76
02BC7  > 8B45 0C        MOV EAX,DWORD PTR SS:[EBP+C]
02BCA  . 8B48 04        MOV ECX,DWORD PTR DS:[EAX+4]
02BCD  . 898D DCE7FFFF  MOV DWORD PTR SS:[EBP-1824],ECX
02BD3  . 68 6CC14000    PUSH Sample1.0040C16C                    ASCII "-re"
02BD8  . 8B95 DCE7FFFF  MOV EDX,DWORD PTR SS:[EBP-1824]
02BDE  . 52             PUSH EDX
02BDF  . E8 2B0C0000    CALL Sample1.0040380F
02BE4  . 83C4 08        ADD ESP,8
02BE7  . 85C0           TEST EAX,EAX
02BE9  . 75 64          JNZ SHORT Sample1.00402C4F
02BEB  . 837D 08 03     CMP DWORD PTR SS:[EBP+8],3
02BEF  . 75 31          JNZ SHORT Sample1.00402C22
02BF1  . 68 00040000    PUSH 400                                 Arg2 = 00000400
02BF6  . 8D85 F8F7FFFF  LEA EAX,DWORD PTR SS:[EBP-808]           Arg1
02BFC  . 50             PUSH EAX                                 Sample1.004025B0
02BFD  . E8 AEF9FFFF    CALL Sample1.004025B0
02C02  . 83C4 08        ADD ESP,8
```

And the password is 'abcd' and it's a bit hard to figure out the meaning of the codes.

First, it is comparing argument size with 4. If it's not, it will jump to other address. If it's 4, and then proceed.

```
.text:00402515                 mov     edi, [ebp+arg_0]
.text:00402518                 or      ecx, 0FFFFFFFFh
.text:0040251B                 xor     eax, eax
.text:0040251D                 repne scasb
.text:0040251F                 not     ecx
.text:00402521                 add     ecx, 0FFFFFFFFh
.text:00402524                 cmp     ecx, 4
.text:00402527                 jz      short loc_40252D
.text:00402529                 xor     eax, eax
.text:0040252B                 jmp     short loc_4025A0
.text:0040252D ; ---------------------------------------------
.text:0040252D
.text:0040252D loc_40252D:                                 ; CODE
.text:0040252D                 mov     eax, [ebp+arg_0]
.text:00402530                 mov     cl, [eax]
.text:00402532                 mov     [ebp+var_4], cl
.text:00402535                 movsx   edx, [ebp+var_4]
.text:00402539                 cmp     edx, 61h
```

This image is where code is comparing first 2 arguments. The first check is straightforward. It is comparing 61h ASCII code. The Second one is a little bit tricky, it is subtracting second one to first one and then it compares the value to 1. Since 'a' and 'b' is apart from each other by 1, so we can know it is trying to find 'b'. The third and fourth character comparing is similar.

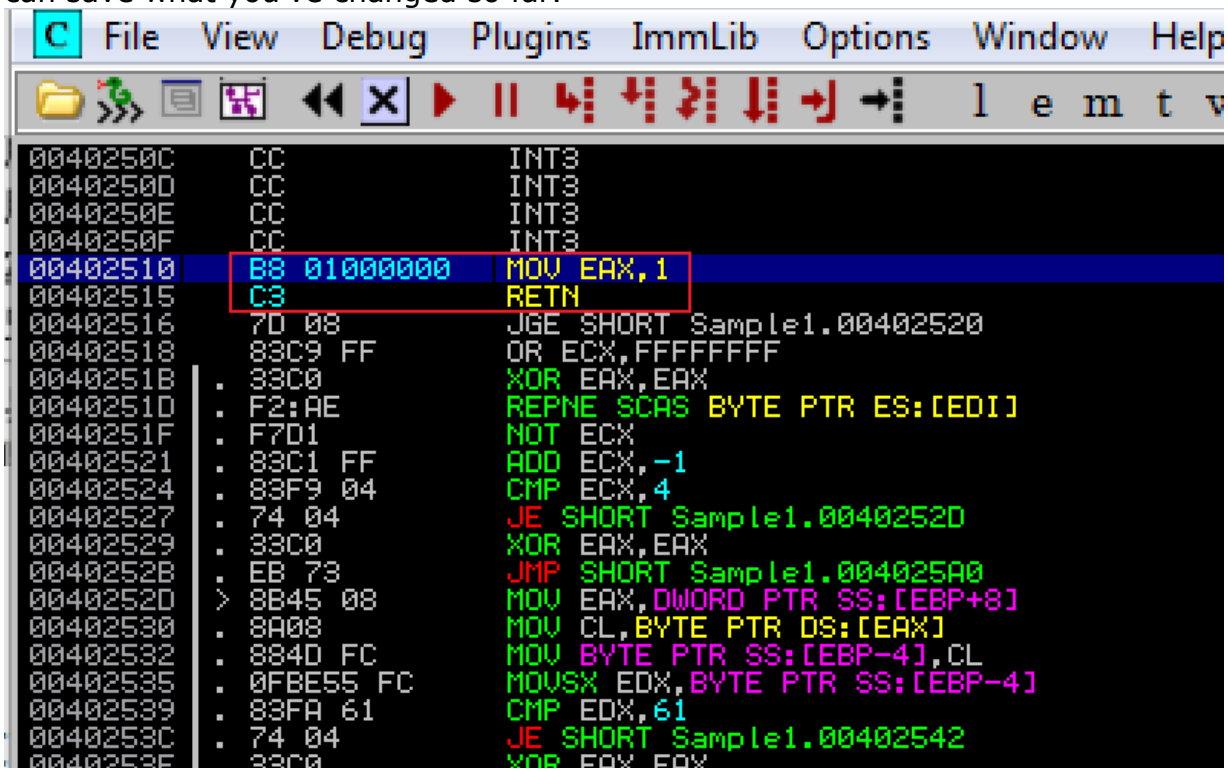
```
.text:0040252D ; ---------------------------------------
.text:0040252D
.text:0040252D loc_40252D:                             ;
.text:0040252D                 mov     eax, [ebp+arg_0]
.text:00402530                 mov     cl, [eax]
.text:00402532                 mov     [ebp+var_4], cl
.text:00402535                 movsx   edx, [ebp+var_4]
.text:00402539                 cmp     edx, 61h
.text:0040253C                 jz      short loc_402542
.text:0040253E                 xor     eax, eax
.text:00402540                 jmp     short loc_4025A0
.text:00402542 ; ---------------------------------------
.text:00402542
.text:00402542 loc_402542:                             ;
.text:00402542                 mov     eax, [ebp+arg_0]
.text:00402545                 mov     cl, [eax+1]
.text:00402548                 mov     [ebp+var_4], cl
.text:0040254B                 mov     edx, [ebp+arg_0]
.text:0040254E                 mov     al, [ebp+var_4]
.text:00402551                 sub     al, [edx]
.text:00402553                 mov     [ebp+var_4], al
.text:00402556                 movsx   ecx, [ebp+var_4]
.text:0040255A                 cmp     ecx, 1
rogram control flow            jz      short loc_402563
.text:0040255F                 xor     eax, eax
.text:00402561                 jmp     short loc_4025A0
```

```
:00402563
:00402563 loc_402563:                          ; CODE XREF: sub_402510+4D↑j
:00402563                 mov     al, [ebp+var_4]
:00402566                 mov     dl, 63h
:00402568                 imul    dl
:0040256A                 mov     [ebp+var_4], al
:0040256D                 movsx   eax, [ebp+var_4]
:00402571                 mov     ecx, [ebp+arg_0]
:00402574                 movsx   edx, byte ptr [ecx+2]
:00402578                 cmp     eax, edx
:0040257A                 jz      short loc_402580
:0040257C                 xor     eax, eax
:0040257E                 jmp     short loc_4025A0
:00402580 ; ---------------------------------------------------------------------
:00402580
:00402580 loc_402580:                          ; CODE XREF: sub_402510+6A↑j
:00402580                 mov     al, [ebp+var_4]
:00402583                 add     al, 1
:00402585                 mov     [ebp+var_4], al
:00402588                 movsx   ecx, [ebp+var_4]
:0040258C                 mov     edx, [ebp+arg_0]
:0040258F                 movsx   eax, byte ptr [edx+3]
:00402593                 cmp     ecx, eax
:00402595                 jz      short loc_40259B
:00402597                 xor     eax, eax
:00402599                 jmp     short loc_4025A0
```

Q1-3. How can you use OllyDbg to permanently patch this malware, so that it doesn't require the special command-line password?

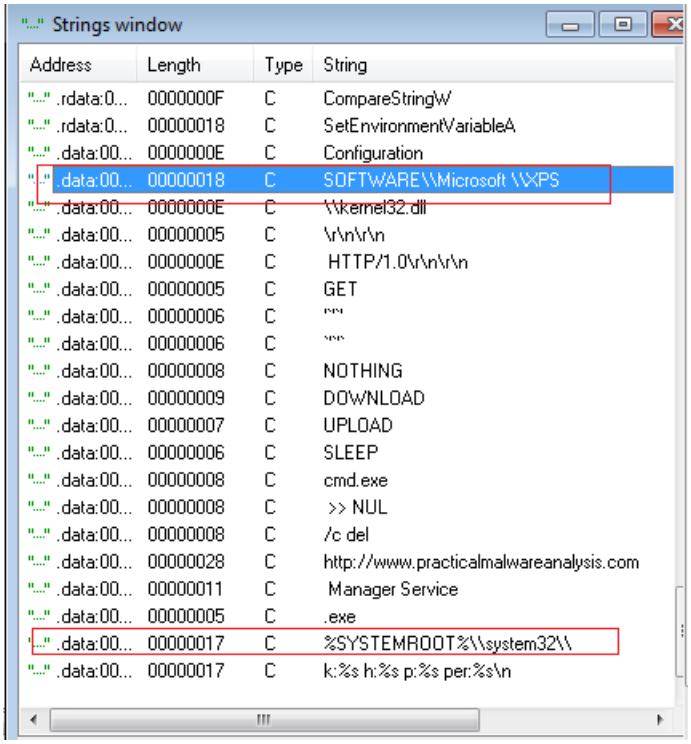
CSE 434S

At address 402510(which is a password check function), I changed assembly code to 'MOV EAX,1 RET' to return always 1.
Click right mouse button and choose 'Binary -> Edit'. And you can edit binary value and put this binary value. 'B8 01 00 00 00 C3' which means that 'MOV EAX,1 RET'
After editing, click right button again and choose 'Copy to executable' and you can save what you've changed so far.



Q1-4. What are the host-based indicators of this malware?

I thinks there are 2 indicators which show that it is a host-based malware.
When I look into the 'Strings window' of IDA, I can see 'SOFTWARE/Microsoft/XPS' and '%SYSTEMROOT%/system32'.
I think it is trying to do something on the victim computer.(Copying a file or making a registry key or something)



Strings window

Address	Length	Type	String
".rdata:0...	0000000F	C	CompareStringW
".rdata:0...	00000018	C	SetEnvironmentVariableA
".data:00...	0000000E	C	Configuration
".data:00...	00000018	C	SOFTWARE\\Microsoft \\XPS
".data:00...	0000000E	C	\\kernel32.dll
".data:00...	00000005	C	\r\n\r\n
".data:00...	0000000E	C	HTTP/1.0\r\n\r\n
".data:00...	00000005	C	GET
".data:00...	00000006	C	[illegible]
".data:00...	00000006	C	[illegible]
".data:00...	00000008	C	NOTHING
".data:00...	00000009	C	DOWNLOAD
".data:00...	00000007	C	UPLOAD
".data:00...	00000006	C	SLEEP
".data:00...	00000008	C	cmd.exe
".data:00...	00000008	C	>> NUL
".data:00...	00000008	C	/c del
".data:00...	00000028	C	http://www.practicalmalwareanalysis.com
".data:00...	00000011	C	Manager Service
".data:00...	00000005	C	.exe
".data:00...	00000017	C	%SYSTEMROOT%\\system32\\
".data:00...	00000017	C	k:%s h:%s p:%s per:%s\n

Q1-5. What are the different actions this malware can be instructed to take via the network?

I found several commands this malware could execute at address 402020.

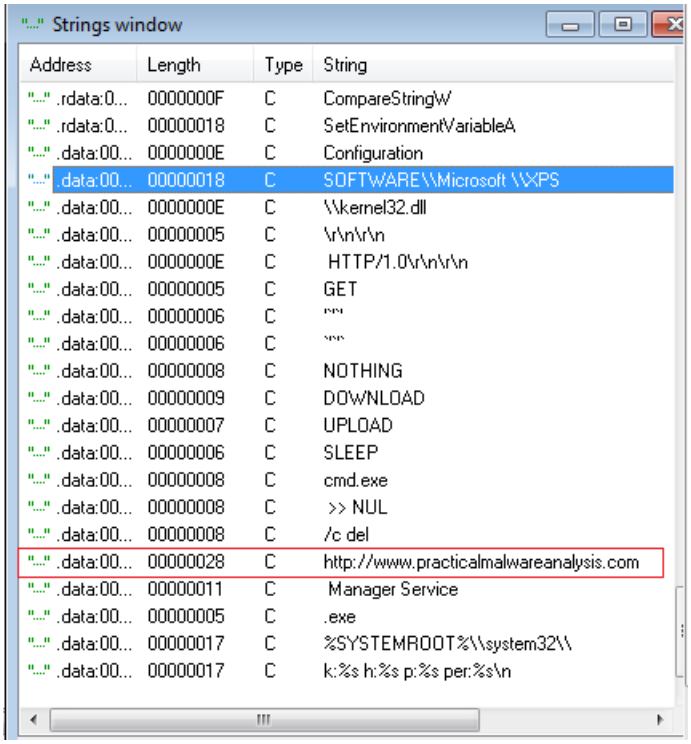
SLEEP
UPLOAD
DOWNLOAD
CMD
NOTHING

```
IDA View-A
.text:00402020 ; int __cdecl sub_402020(char *name)
.text:00402020 sub_402020      proc near               ; CODE XREF: sub_402360+74↓p
.text:00402020
.text:00402020 hostshort       = word ptr -424h
.text:00402020 var_420         = dword ptr -420h
.text:00402020 var_41C         = dword ptr -41Ch
.text:00402020 var_418         = dword ptr -418h
.text:00402020 lpFileName      = dword ptr -414h
.text:00402020 var_410         = dword ptr -410h
.text:00402020 var_40C         = dword ptr -40Ch
.text:00402020 var_408         = dword ptr -408h
.text:00402020 var_404         = dword ptr -404h
.text:00402020 var_400         = byte ptr -400h
.text:00402020 name            = dword ptr  8
.text:00402020
.text:00402020                 push    ebp
.text:00402021                 mov     ebp, esp
.text:00402023                 sub     esp, 424h
.text:00402029                 push    edi
.text:0040202A                 push    400h
.text:0040202F                 lea     eax, [ebp+var_400]
.text:00402035                 push    eax
.text:00402036                 call    sub_401E60
.text:0040203B                 add     esp, 8
.text:0040203E                 test    eax, eax
.text:00402040                 jz      short loc_40204C
.text:00402042                 mov     eax, 1
.text:00402047                 jmp     loc_402358
.text:0040204C ; ---------------------------------------------------------------------------
.text:0040204C
.text:0040204C loc_40204C:                             ; CODE XREF: sub_402020+20↑j
.text:0040204C                 mov     edi, offset aSleep ; "SLEEP"
.text:00402051                 or      ecx, 0FFFFFFFFh
.text:00402054                 xor     eax, eax
```

Q1-6. Are there any useful network-based signatures for this malware?

'http://www.practicalmalwareanalysis.com'

CSE 434S



Strings window

Address	Length	Type	String
"..." .rdata:0...	0000000F	C	CompareStringW
"..." .rdata:0...	00000018	C	SetEnvironmentVariableA
"..." .data:00...	0000000E	C	Configuration
"..." .data:00...	00000018	C	SOFTWARE\\Microsoft \\XPS
"..." .data:00...	0000000E	C	\\kernel32.dll
"..." .data:00...	00000005	C	\r\n\r\n
"..." .data:00...	0000000E	C	HTTP/1.0\r\n\r\n
"..." .data:00...	00000005	C	GET
"..." .data:00...	00000006	C	[illegible]
"..." .data:00...	00000006	C	[illegible]
"..." .data:00...	00000008	C	NOTHING
"..." .data:00...	00000009	C	DOWNLOAD
"..." .data:00...	00000007	C	UPLOAD
"..." .data:00...	00000006	C	SLEEP
"..." .data:00...	00000008	C	cmd.exe
"..." .data:00...	00000008	C	>> NUL
"..." .data:00...	00000008	C	/c del
"..." .data:00...	00000028	C	http://www.practicalmalwareanalysis.com
"..." .data:00...	00000011	C	Manager Service
"..." .data:00...	00000005	C	.exe
"..." .data:00...	00000017	C	%SYSTEMROOT%\\system32\\
"..." .data:00...	00000017	C	k:%s h:%s p:%s per:%s\n
