Computer Science 491/691
Malware Analysis
Final Exam

Assigned: May 13, 2019

Due: 11:59pm May 20, 2019

Name: Sai Sasaank Srivatsa Pallerla

ID: HG13015

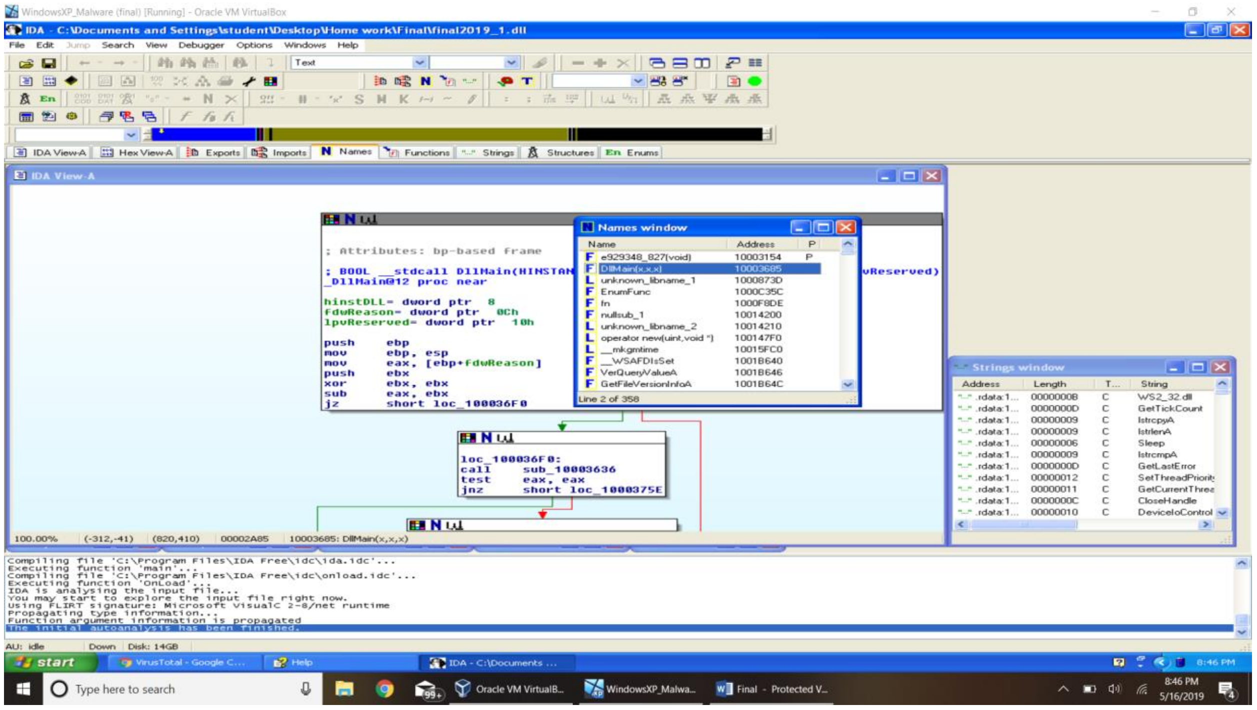
Download and extract final2019.7z on a Windows XP virtual machine. The password is infected. Make sure to disconnect your VM from the network!

There are 110 points available, but scores will be computed on a base of 100. So 100/110 is a "perfect" score. This is a form of built-in curve…

Part 1: final2019_01.dll (Total 55 pts)

1) **What is the address of the DLLMain function? (2 pts)**

   Address of DLLMain function is 10003685



```
; Attributes: bp-based frame

; BOOL __stdcall DllMain(HINSTAN
_DllMain@12 proc near

hinstDLL= dword ptr  8
fdwReason= dword ptr  0Ch
lpvReserved= dword ptr  10h

push    ebp
mov     ebp, esp
mov     eax, [ebp+fdwReason]
push    ebx
xor     ebx, ebx
sub     eax, ebx
jz      short loc_100036F0

loc_100036F0:
call    sub_10003636
test    eax, eax
jnz     short loc_1000375E
```

Names window

Name	Address	P
e929348_827(void)	10003154	P
DllMain(x,x,x)	10003685	
unknown_libname_1	1000873D	
EnumFunc	1000C35C	
fn	1000F8DE	
nullsub_1	10014200	
unknown_libname_2	10014210	
operator new(uint,void *)	100147F0	
__mkgmtime	10015FC0	
__WSAFDIsSet	1001B640	
VerQueryValueA	1001B646	
GetFileVersionInfoA	1001B64C	

Line 2 of 358

Strings window

Address	Length	T...	String
.rdata:1...	0000000B	C	WS2_32.dll
.rdata:1...	0000000D	C	GetTickCount
.rdata:1...	00000009	C	lstrcpyA
.rdata:1...	00000009	C	lstrlenA
.rdata:1...	00000006	C	Sleep
.rdata:1...	00000009	C	lstrcmpA
.rdata:1...	0000000D	C	GetLastError
.rdata:1...	00000012	C	SetThreadPriorit
.rdata:1...	00000011	C	GetCurrentThrea
.rdata:1...	0000000C	C	CloseHandle
.rdata:1...	00000010	C	DeviceIoControl

```
100.00%   (-312,-41)   (820,410)   00002A85   10003685: DllMain(x,x,x)
```

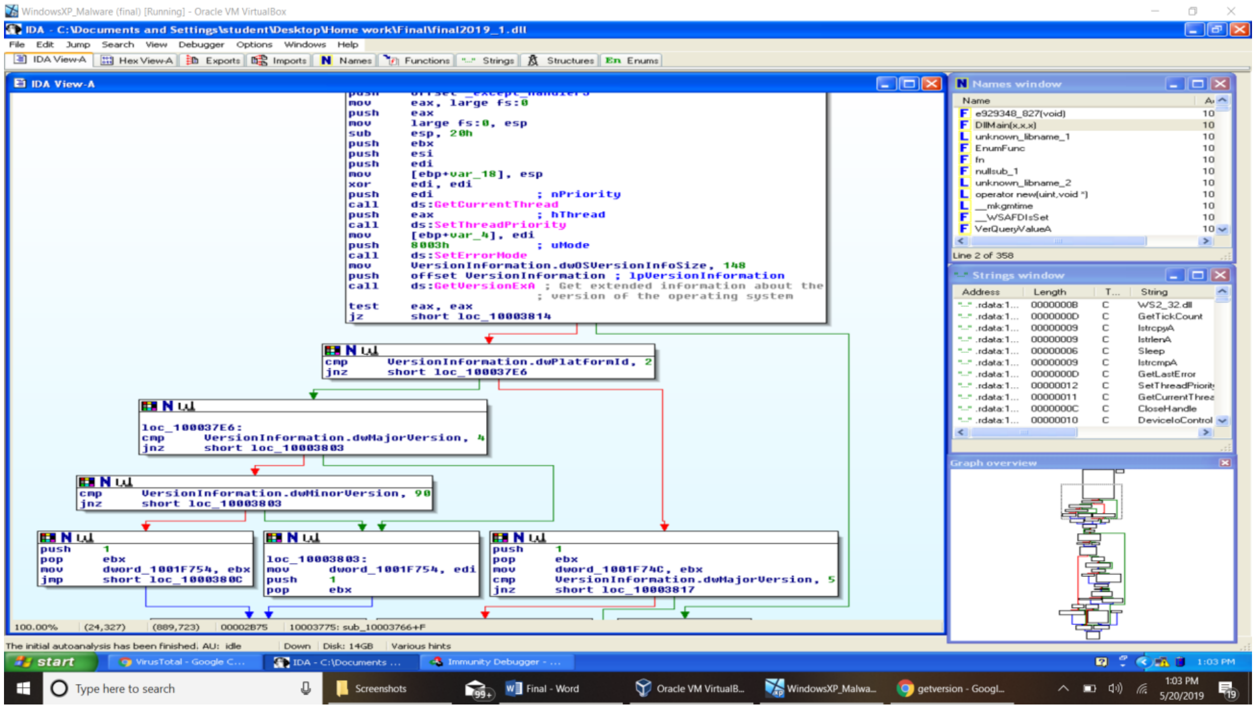
```
Compiling file 'C:\Program Files\IDA Free\idc\ida.idc'...
Executing function 'main'...
Compiling file 'C:\Program Files\IDA Free\idc\onload.idc'...
Executing function 'OnLoad'...
IDA is analysing the input file...
You may start to explore the input file right now.
Using FLIRT signature: Microsoft VisualC 2-8/net runtime
Propagating type information...
Function argument information is propagated
The initial autoanalysis has been finished.
```

WindowsXP_Malware (final) [Running] - Oracle VM VirtualBox
IDA - C:\Documents and Settings\student\Desktop\Home work\Final\final2019_1.dll
File Edit Jump Search View Debugger Options Windows Help
IDA View-A Hex View-A Exports Imports Names Functions Strings Structures Enums
AU: idle Down Disk: 14GB
start VirusTotal - Google C... Help IDA - C:\Documents ...
Type here to search
Oracle VM VirtualB... WindowsXP_Malwa... Final - Protected V...
8:46 PM 5/16/2019

2) **The malware checks the version of Windows that it is running on and exits prematurely if it is not running on the right version. What is the full name of the <u>minimum</u> Windows operating system version that the malware will run on? Justify your answer. (10 pts)**

The function that is responsible for checking the version of the host operating system is sub_10003766. By inspecting the function we can see that the malware uses GetVersion to identify the operation system info. This Windows API returns major – operating system platform and minor – operating system version.

We can see that the function has two cpm statements which compare the return value of GetVersion against majorversioin 5 and platformed 2, here 2 signifies if it's a 32 bit platforom or not. So we can say that the minimum system requirement is Windows XP.



```
mov     eax, large fs:0
push    eax
mov     large fs:0, esp
sub     esp, 20h
push    ebx
push    esi
push    edi
mov     [ebp+var_18], esp
xor     edi, edi
push    edi             ; nPriority
call    ds:GetCurrentThread
push    eax             ; hThread
call    ds:SetThreadPriority
mov     [ebp+var_4], edi
push    8003h           ; uMode
call    ds:SetErrorMode
mov     VersionInformation.dwOSVersionInfoSize, 148
push    offset VersionInformation ; lpVersionInformation
call    ds:GetVersionExA ; Get extended information about the
                        ; version of the operating system
test    eax, eax
jz      short loc_10003814

cmp     VersionInformation.dwPlatformId, 2
jnz     short loc_100037E6

loc_100037E6:
cmp     VersionInformation.dwMajorVersion, 4
jnz     short loc_10003803

cmp     VersionInformation.dwMinorVersion, 90
jnz     short loc_10003803

push    1
pop     ebx
mov     dword_1001F754, ebx
jmp     short loc_1000380C

loc_10003803:
mov     dword_1001F754, edi
push    1
pop     ebx

push    1
pop     ebx
mov     dword_1001F74C, ebx
cmp     VersionInformation.dwMajorVersion, 5
jnz     short loc_10003817
```

```
Name                              A
F  e929348_827(void)              10
F  DllMain(x,x,x)                 10
L  unknown_libname_1              10
F  EnumFunc                       10
F  fn                             10
F  nullsub_1                      10
L  unknown_libname_2              10
L  operator new(uint,void *)      10
L  __mkgmtime                     10
F  __WSAFDIsSet                   10
F  VerQueryValueA                 10
Line 2 of 358
```

```
Address      Length    T...  String
.rdata:1...  0000000B  C     WS2_32.dll
.rdata:1...  0000000D  C     GetTickCount
.rdata:1...  00000009  C     lstrcpyA
.rdata:1...  00000009  C     lstrlenA
.rdata:1...  00000006  C     Sleep
.rdata:1...  00000009  C     lstrcmpA
.rdata:1...  0000000D  C     GetLastError
.rdata:1...  00000012  C     SetThreadPriority
.rdata:1...  00000011  C     GetCurrentThrea
.rdata:1...  0000000C  C     CloseHandle
.rdata:1...  00000010  C     DeviceIoControl
```

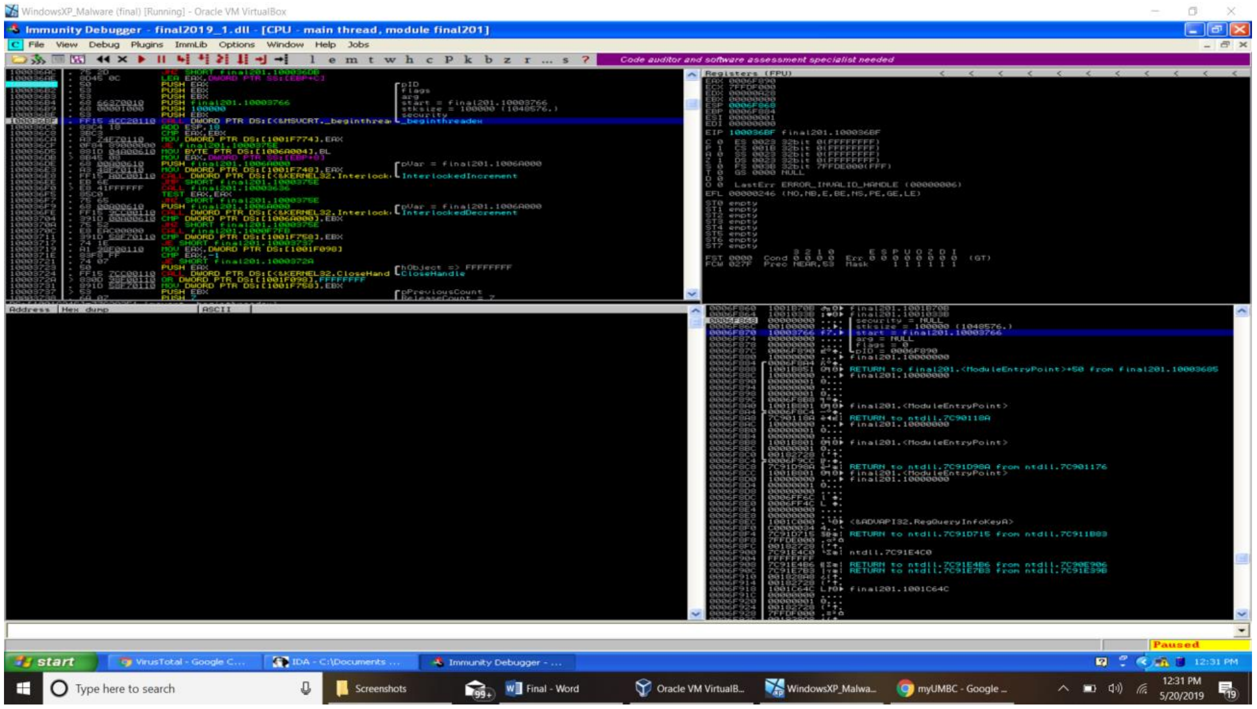
```
100.00%   (24,327)   (889,723)   00002B75   10003775: sub_10003766+F
The initial autoanalysis has been finished. AU:  idle   Down   Disk: 14GB   Various hints
```

Value	**Meaning**
VER_PLATFORM_WIN32_NT 2	The operating system is Windows 7, Windows Server 2008, Windows Vista, Windows Server 2003, Windows XP, or Windows 2000.

Windows Server 2008	6.0
Windows XP 64-Bit Edition	5.2
Windows Server 2003 R2	5.2
Windows Server 2003	5.2
Windows XP	5.1

3) **What is the address of the function that is executed by the call to _beginthreadex in DLLMain? (3 pts)**

The function _beginthreadex takes 6 parameters out of which the third parameter is an offset value to a function whose address is 10003766. So we can say that the address of the function that is executed by the call to _beginthreadex is 10003766


WindowsXP_Malware (final) [Running] - Oracle VM VirtualBox

Immunity Debugger - final2019_1.dll - [CPU - main thread, module final201]

```
100036AC  . 75 20           JNZ SHORT final201.100036CE
100036AE  . 8D45 0C         LEA EAX,DWORD PTR SS:[EBP+C]
100036B1  . 50              PUSH EAX                                   [pID
100036B2  . 53              PUSH EBX                                   |flags
100036B3  . 53              PUSH EBX                                   |arg
100036B4  . 68 66370010     PUSH final201.10003766                     |start = final201.10003766
100036B9  . 68 00001000     PUSH 100000                                |stksize = 100000 (1048576.)
100036BE  . 53              PUSH EBX                                   |security
100036BF  . FF15 4CC20110   CALL DWORD PTR DS:[<&MSVCRT._beginthrea   L_beginthreadex
100036C5  . 83C4 18         ADD ESP,18
100036C8  . 3BC3            CMP EAX,EBX
100036CA  . A3 74F70110     MOV DWORD PTR DS:[1001F774],EAX
100036CF  . 0F84 89000000   JE final201.1000375E
100036D5  . 881D 04A00610   MOV BYTE PTR DS:[1006A004],BL
100036DB  > 8B45 08         MOV EAX,DWORD PTR SS:[EBP+8]
100036DE  . 68 00A00610     PUSH final201.1006A000                     [pVar = final201.1006A000
100036E3  . A3 48F70110     MOV DWORD PTR DS:[1001F748],EAX            |
100036E8  . FF15 A0C00110   CALL DWORD PTR DS:[<&KERNEL32.Interlock   LInterlockedIncrement
100036EE  . EB 6E           JMP SHORT final201.1000375E
100036F0  > E8 41FFFFFF     CALL final201.10003636
100036F5  . 85C0            TEST EAX,EAX
100036F7  . 75 65           JNZ SHORT final201.1000375E
100036F9  . 68 00A00610     PUSH final201.1006A000                     [pVar = final201.1006A000
100036FE  . FF15 9CC00110   CALL DWORD PTR DS:[<&KERNEL32.Interlock   LInterlockedDecrement
10003704  . 391D 00A00610   CMP DWORD PTR DS:[1006A000],EBX
1000370A  . 75 52           JNZ SHORT final201.1000375E
1000370C  . E8 EAC00000     CALL final201.1000F7FB
10003711  . 391D 58F70110   CMP DWORD PTR DS:[1001F758],EBX
10003717  . 74 1E           JE SHORT final201.10003737
10003719  . A1 98F00110     MOV EAX,DWORD PTR DS:[1001F098]
1000371E  . 83F8 FF         CMP EAX,-1
10003721  . 74 07           JE SHORT final201.1000372A
10003723  . 50              PUSH EAX                                   [hObject => FFFFFFFF
10003724  . FF15 7CC00110   CALL DWORD PTR DS:[<&KERNEL32.CloseHand   LCloseHandle
1000372A  > 830D 98F00110   OR DWORD PTR DS:[1001F098],FFFFFFFF
10003731  . 891D 58F70110   MOV DWORD PTR DS:[1001F758],EBX
10003737  > 53              PUSH EBX                                   [pPreviousCount
10003738  . 6A 07           PUSH 7                                     |ReleaseCount = 7
```

```
Registers (FPU)
EAX 0006F890
ECX 7FFDF000
EDX 00000A28
EBX 00000000
ESP 0006F868
EBP 0006F884
ESI 00000001
EDI 00000000

EIP 100036BF final201.100036BF

C 0  ES 0023 32bit 0(FFFFFFFF)
P 1  CS 001B 32bit 0(FFFFFFFF)
A 0  SS 0023 32bit 0(FFFFFFFF)
Z 1  DS 0023 32bit 0(FFFFFFFF)
S 0  FS 003B 32bit 7FFDE000(FFF)
T 0  GS 0000 NULL
D 0
O 0  LastErr ERROR_INVALID_HANDLE (00000006)

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

ST0 empty
ST1 empty
ST2 empty
ST3 empty
ST4 empty
ST5 empty
ST6 empty
ST7 empty
               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
```

```
0006F860   1001B70B   final201.1001B70B
0006F864   10010338   final201.10010338
0006F868   00000000   [security = NULL
0006F86C   00100000   |stksize = 100000 (1048576.)
0006F870   10003766   |start = final201.10003766
0006F874   00000000   |arg = NULL
0006F878   00000000   |flags = 0
0006F87C   0006F890   LpID = 0006F890
0006F880   10000000   final201.10000000
0006F884   0006F8A4
0006F888   1001B851   RETURN to final201.<ModuleEntryPoint>+50 from final201.10003685
0006F88C   10000000   final201.10000000
0006F890   00000001
0006F894   00000000
0006F898   00000001
0006F89C   0006F8B8
0006F8A0   1001B801   final201.<ModuleEntryPoint>
0006F8A4   0006F8C4
0006F8A8   7C90118A   RETURN to ntdll.7C90118A
0006F8AC   10000000   final201.10000000
0006F8B0   00000001
0006F8B4   00000000
0006F8B8   1001B801   final201.<ModuleEntryPoint>
0006F8BC   00000001
0006F8C0   00182728
0006F8C4   0006F9CC
0006F8C8   7C91D98A   RETURN to ntdll.7C91D98A from ntdll.7C901176
0006F8CC   1001B801   final201.<ModuleEntryPoint>
0006F8D0   10000000   final201.10000000
0006F8D4   00000001
0006F8D8   00000000
0006F8DC   0006FF6C
0006F8E0   0006FF4C
0006F8E4   00000000
0006F8E8   00000000
0006F8EC   1001C000   <&ADVAPI32.RegQueryInfoKeyA>
0006F8F0   C0000034
0006F8F4   7C91D715   RETURN to ntdll.7C91D715 from ntdll.7C911B83
0006F8F8   7FFDE000
0006F8FC   00182728
0006F900   7C91E4C0   ntdll.7C91E4C0
0006F904   FFFFFFFF
0006F908   7C91E4B6   RETURN to ntdll.7C91E4B6 from ntdll.7C90E906
0006F90C   7C91E7B3   RETURN to ntdll.7C91E7B3 from ntdll.7C91E39B
0006F910   001828A8
0006F914   00182728
0006F918   1001C64C   final201.1001C64C
0006F91C   00000000
0006F920   00000001
0006F924   00182728
0006F928   7FFDF000
```

Paused

4) **List 5 imported functions that you believe are suspicious. For each import, summarize what it does and why you chose it. (5 pts)**

a) SetWindowsHook: Sets a hook function to be called whenever a certain event is called. Commonly used with keyloggers and spyware, this function also provides an easy way to load a DLL into all GUI processes on the system. This function is sometimes added by the compiler. Could be used to perform keylogging. (with combination of CallNextHook, UnHookWindowsHook)

b) GetTickCount: Retrieves the number of milliseconds that have elapsed since the system was started Could be used to perform anti-debugging.

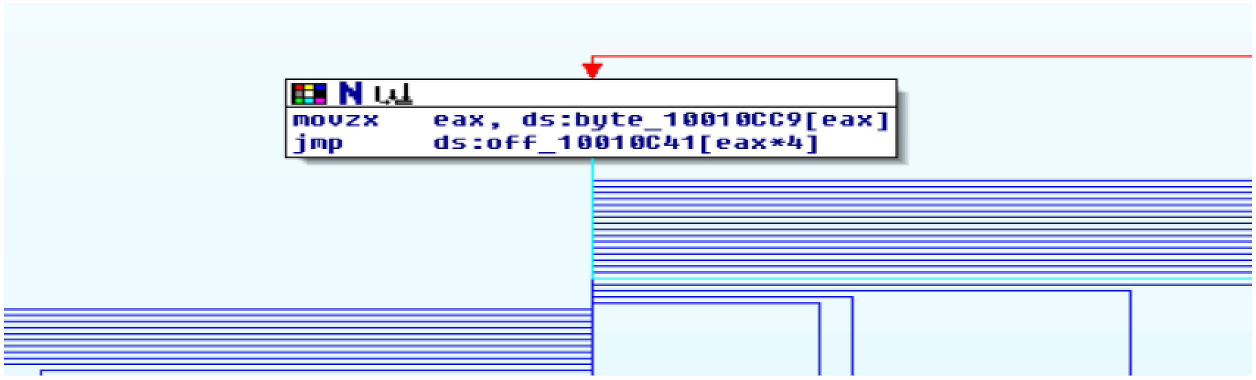
c) GetHostByName: Used to perform a DNS lookup on a hostname prior to making an IP connection to a remote host. Could be used to conect/ check network connectivity, sometimes WSAStartup also cound be used to check/ to initialize low-level network functionality probably to connect to something.

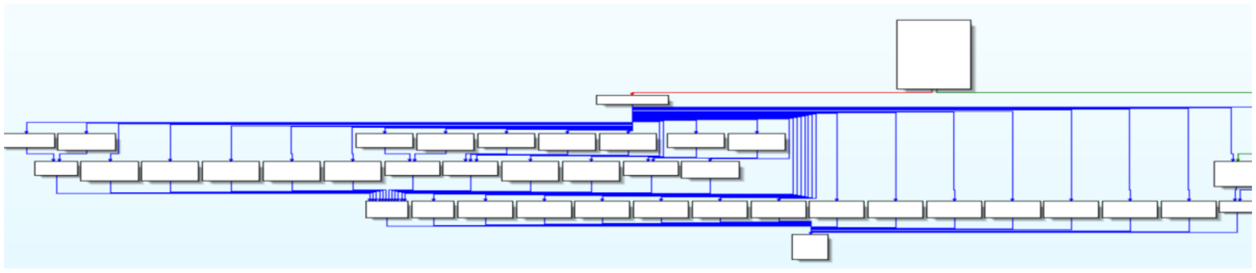
d) *GetProcAddress* – Gets the address of a function from a DLL in memory. Could be used to perform runtime linking. (with a combination of LoadLibrary).

e) Send/ Revc : Sends data on a connected socket. Receives data from a remote machine. Malware often uses this function to receive data from a remote command-and-control server.

5) **What is the address of the function in which both ImpersonateLoggedOnUser and OpenProcessToken are called? In a few sentences, describe what this function does. (8 pts)**
Address of function in which both ImpersonateLoggedOnUser and OpenProcessToken are called id sub_10011B0E, where ImpersonateLoggedOnUser is called at 10011B65 OpenProcessToken is called at 10011B52.
This function opens a local (existing) process and then gets the access token to that process. It uses the token handle that it got by using the API call OpenProcessToken and uses it to modify the security context of that process (ImpersonateLoggedOnUser) giving it all access pass. Later it checks for any errors that occurred while performing this operation and then closes the handle to the process.

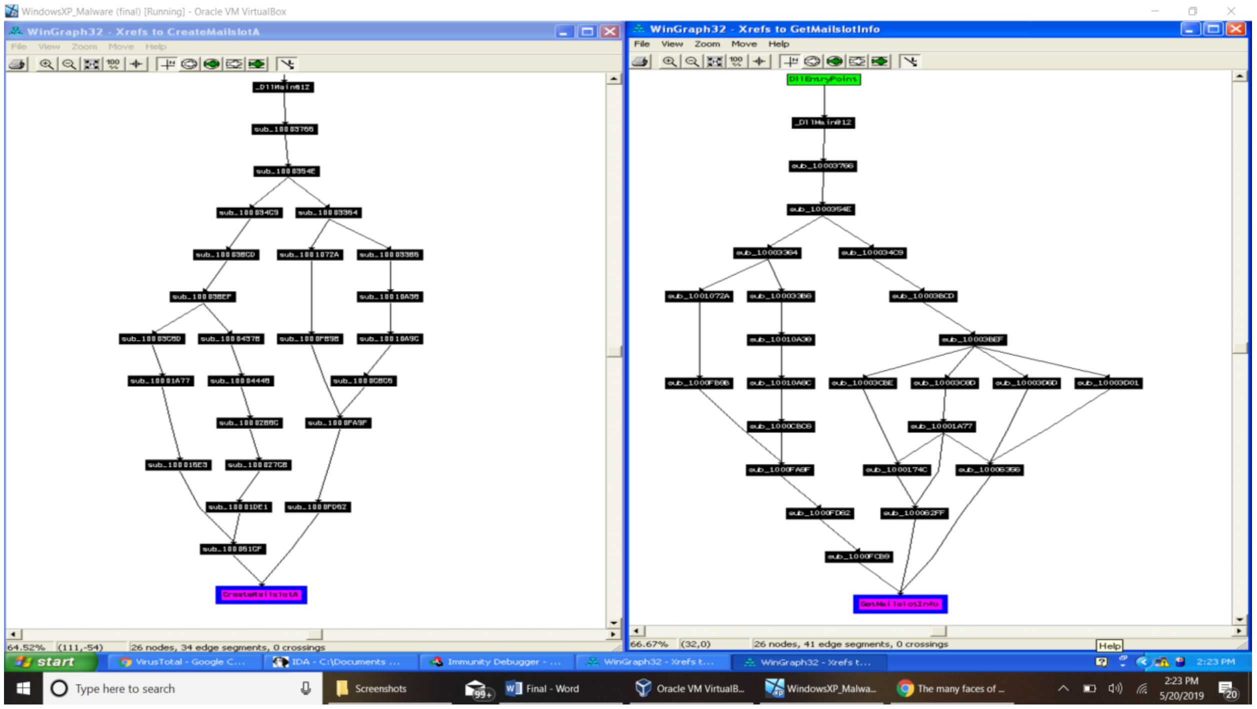
6) **What kind of uncommon C code construct is present in the function sub_10010A9C? What is the malware's purpose for this function? (7 pts)**
The code construct looks like a switch statement, where figure 1 shows the switch condition and all the branching out shows switch cases. Here eax register stores the condition for the switch statement and jmp statement specifies which switch case it should jump to. The malware might be doing this for two possible reasons. One, based on the arguments pushed at the calling function, the switch statement will decide what location to execute (Upon going to the calling function of sub_10010A9C we can see that it is in a loop, so we can conclude that it is trying to perform operations based on the result from other functions which changes for every iteration.) or Two, to throw off the analyst because looking at the add and compare statements it looks like the code in fig 1 is never executed. It goes to two other functions where ReleaseSemaphore is being called.


```
movzx   eax, ds:byte_10010CC9[eax]
jmp     ds:off_10010C41[eax*4]
```



7) All of the calls to CreateMailSlotA and GetMailSlotInfo can eventually be traced back to one common function. What is the address of this function? Why is it necessary for this malware to use mailslots rather than pipes? (8 pts)

Using the option Charts from Xref to Operands we can obtain the follows charts that helps us trace back function calls. From the chart both CreateMailSlotA and GetMailSlotInfo can be traced back to one unction which is 1000354E. Mailslots are primarily used for broadcast messages and retrieve information stored buy the author, in a situation where one is trying to connect to all the systems in a range Mailslots are useful, this is more like UDP – one way connection. Whereas Pipes are like TCP, it is connection oriented and it waits for response from client.



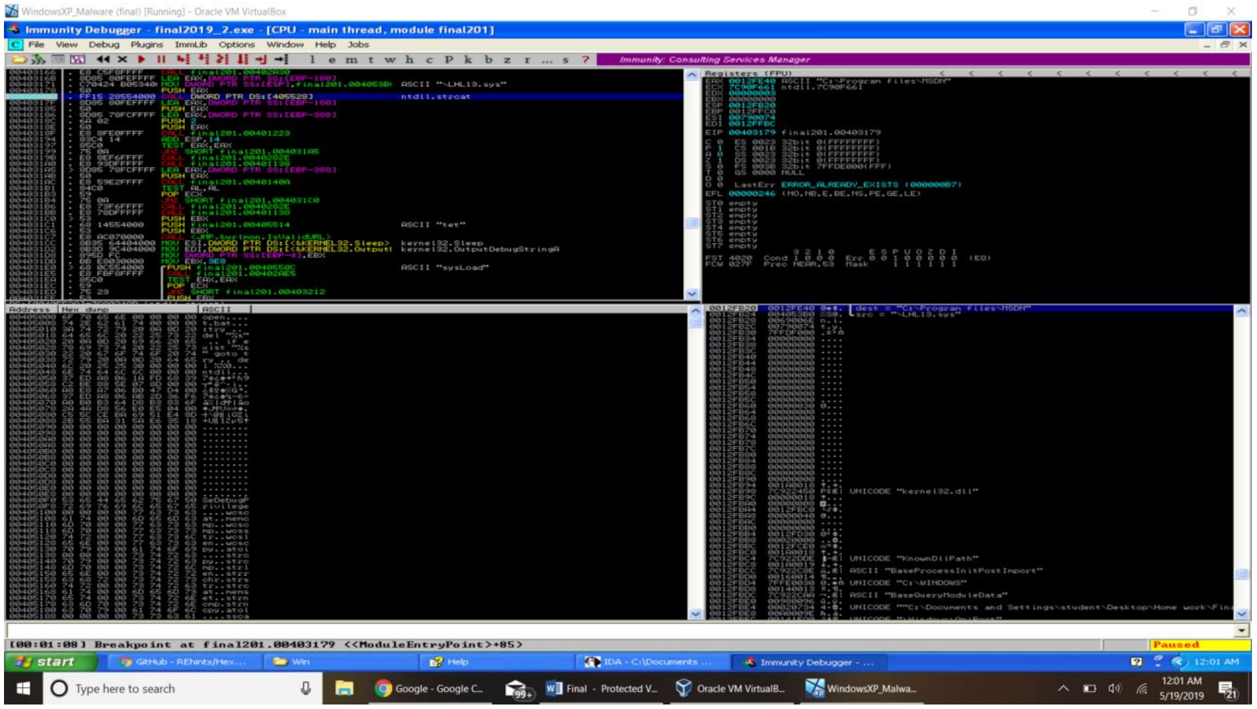
8) What category of malware do you believe final2019_1.dll belongs to? In a paragraph, justify your answer using specific examples from IDA Pro. (12 pts)

By examining the DLL it is clear that the malware is performing Process Injection, Anti-Debugging Timing Checks, Privilege Escalation, Access Token Manipulation, Stealing Credentials and performing Runtime Linking. Because it uses mailslots (all this can be analyzed based on the info from IDA Pro) we could also say that it a POC related virus. So final2019_1.dll belong is trojan that sets up a backdoor and steals credentials.

Part 2: final2019_2.exe (55 pts)

1) **In a few sentences, describe what sub_4019E8 does. Why is the malware doing this? (8 pts)**
The function sub_4019E8 uses various Windows API calls such as GetCurrentProcess, OpenProcessToken, LookupPrivilegeValue, AdjustTokenPrevilige and CloseHnadle. This function first gets a pseudo handle of the current running process and retrieves the access token of that process. Now by using the access token of that it retrieves LUID which is the unique identifier for the process. Using LUID it modifies the security description of the process giving it all access pass. The malware might be doing this for escalate privileges for itself or a specific process.

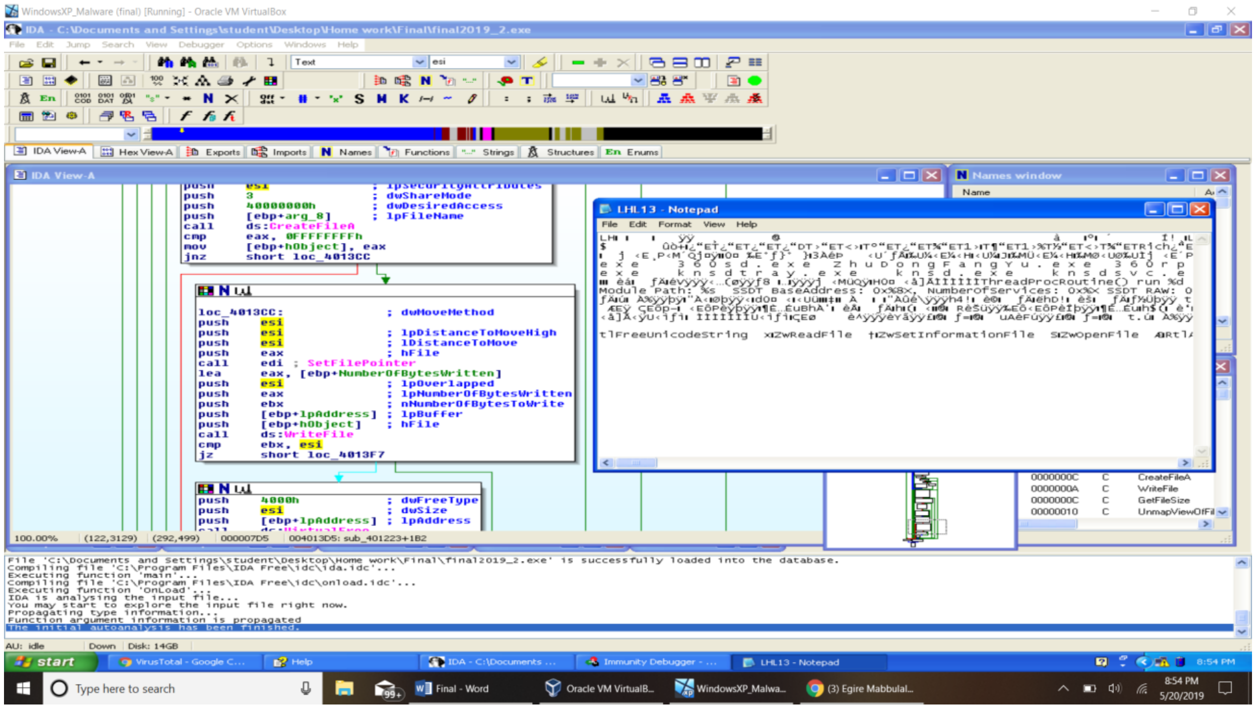
2) **What is the name of the function that dword_405528 points to after the instruction at 0x402790 is executed? When dword_405528 is called at 0x403179, it returns a pointer to a file path. What is the value of this file path? (8 pts)**
Name of the function that dword_405528 points to after the instruction at 402790 is executed is strcat. File path when dword_45528 is called at 403179 is C:\Program files\MSDN\LHL13.sys



```
WindowsXP_Malware (final) [Running] - Oracle VM VirtualBox
Immunity Debugger - final2019_2.exe - [CPU - main thread, module final201]

00403166  . E8 C5F8FFFF     CALL final201.00402A30
0040316B  . 8D85 80FEFFFF   LEA EAX,DWORD PTR SS:[EBP-180]
00403171  . C70424 B0534000 MOV DWORD PTR SS:[ESP],final201.004053B0  ASCII "\LHL13.sys"
00403178  . 50              PUSH EAX
00403179  . FF15 28554000   CALL DWORD PTR DS:[405528]                ntdll.strcat
0040317F  . 8D85 80FEFFFF   LEA EAX,DWORD PTR SS:[EBP-180]
00403185  . 50              PUSH EAX
00403186  . 8D85 78FCFFFF   LEA EAX,DWORD PTR SS:[EBP-388]
0040318C  . 6A 02           PUSH 2
0040318E  . 50              PUSH EAX
0040318F  . E8 8FE0FFFF     CALL final201.00401223
00403194  . 83C4 14         ADD ESP,14
00403197  . 85C0            TEST EAX,EAX
00403199  . 75 0A           JNZ SHORT final201.004031A5
0040319B  . E8 8EF6FFFF     CALL final201.0040282E
004031A0  . E8 93DFFFFF     CALL final201.00401138
004031A5  > 8D85 78FCFFFF   LEA EAX,DWORD PTR SS:[EBP-388]
004031AB  . 50              PUSH EAX
004031AC  . E8 59E2FFFF     CALL final201.0040140A
004031B1  . 84C0            TEST AL,AL
004031B3  . 59              POP ECX
004031B4  . 75 0A           JNZ SHORT final201.004031C0
004031B6  . E8 73F6FFFF     CALL final201.0040282E
004031BB  . E8 78DFFFFF     CALL final201.00401138
004031C0  > 53              PUSH EBX
004031C1  . 68 14554000     PUSH final201.00405514                    ASCII "tet"
004031C6  . 53              PUSH EBX
004031C7  . E8 AC070000     CALL <JMP.&urlmon.IsValidURL>
004031CC  . 8B35 64404000   MOV ESI,DWORD PTR DS:[<&KERNEL32.Sleep>]  kernel32.Sleep
004031D2  . 8B3D 9C404000   MOV EDI,DWORD PTR DS:[<&KERNEL32.Outputl  kernel32.OutputDebugStringA
004031D8  . 895D FC         MOV DWORD PTR SS:[EBP-4],EBX
004031DB  . BB E8030000     MOV EBX,3E8
004031E0  > 68 0C554000     PUSH final201.0040550C                    ASCII "sysLoad"
004031E5  . E8 FBF8FFFF     CALL final201.00402AE5
004031EA  . 85C0            TEST EAX,EAX
004031EC  . 59              POP ECX
004031ED  . 75 23           JNZ SHORT final201.00403212

Registers (FPU)
EAX 0012FE40 ASCII "C:\Program files\MSDN"
ECX 7C90F661 ntdll.7C90F661
EDX 00000003
EBX 00000000
ESP 0012FB20
EBP 0012FFC0
ESI 00790074
EDI 0012FFBC
EIP 00403179 final201.00403179
C 0  ES 0023 32bit 0(FFFFFFFF)
P 1  CS 001B 32bit 0(FFFFFFFF)
A 0  SS 0023 32bit 0(FFFFFFFF)
Z 1  DS 0023 32bit 0(FFFFFFFF)
S 0  FS 003B 32bit 7FFDE000(FFF)
T 0  GS 0000 NULL
D 0
O 0  LastErr ERROR_ALREADY_EXISTS (000000B7)
EFL 00000246 (NO,NB,E,BE,NS,PE,GE,LE)
ST0 empty
ST1 empty
ST2 empty
ST3 empty
ST4 empty
ST5 empty
ST6 empty
ST7 empty
               3 2 1 0      E S P U O Z D I
FST 4020  Cond 1 0 0 0  Err 0 0 1 0 0 0 0 0  (EQ)
FCW 027F  Prec NEAR,53  Mask    1 1 1 1 1 1

Address   Hex dump                  ASCII
00405000  6F 70 65 6E 00 00 00 00   open....
00405008  74 2E 62 61 74 00 00 00   t.bat...
00405010  3A 74 72 79 20 0A 0D 20   :try ..
00405018  64 65 6C 20 22 25 73 22   del "%s"
00405020  20 0A 0D 20 69 66 20 65    .. if e
00405028  78 69 73 74 20 22 25 73   xist "%s
00405030  22 20 67 6F 74 6F 20 74   " goto t
00405038  72 79 20 0A 0D 20 64 65   ry .. de
00405040  6C 20 25 25 30 00 00 00   l %%0...
00405048  6E 74 64 6C 6C 00 00 00   ntdll...

0012FB20  0012FE40  @■#.  [dest = "C:\Program files\MSDN"
0012FB24  004053B0  ■S@.  [src = "\LHL13.sys"
0012FB28  0069006E  n.i.
0012FB2C  00790074  t.y.
0012FB30  7FFDF000  .≡■⌂
0012FB34  00000000  ....
0012FB38  00000000  ....
0012FB3C  00000000  ....
0012FB40  00000000  ....
0012FB44  00000000  ....
0012FB48  00000000  ....
                          UNICODE "kernel32.dll"
                          UNICODE "KnownDllPath"
                          ASCII "BaseProcessInitPostImport"
                          UNICODE "C:\WINDOWS"
                          ASCII "BaseQueryModuleData"
                          UNICODE ""C:\Documents and Settings\student\Desktop\Home work\Fin

[00:01:08] Breakpoint at final201.00403179 <<ModuleEntryPoint>+85>   Paused
```

3) **When sub_401223 is called at 0x40318F, it writes the file from the previous question to disk. Investigate this file. What malicious action does it likely perform? Justify your answer. (8 pts)**

The file name when the function sub_401223 is *LHL13.sys*. Upon inspecting the function, we can see that there is a write statement which is writing into the file LHL13.sys. So, we can say that this function is writing data form final2019_2.exe to LHL13.sys. Upon going to the directory MSDN we can see that the file *LHL13.sys* is *a binary file* and it contains executable code.



WindowsXP_Malware (final) [Running] - Oracle VM VirtualBox

IDA - C:\Documents and Settings\student\Desktop\Home work\Final\final2019_2.exe

```
push    esi              ; lpSecurityAttributes
push    3                ; dwShareMode
push    40000000h        ; dwDesiredAccess
push    [ebp+arg_8]      ; lpFileName
call    ds:CreateFileA
cmp     eax, 0FFFFFFFFh
mov     [ebp+hObject], eax
jnz     short loc_4013CC

loc_4013CC:                      ; dwMoveMethod
push    esi
push    esi                      ; lpDistanceToMoveHigh
push    esi                      ; lDistanceToMove
push    eax                      ; hFile
call    edi ; SetFilePointer
lea     eax, [ebp+NumberOfBytesWritten]
push    esi                      ; lpOverlapped
push    eax                      ; lpNumberOfBytesWritten
push    ebx                      ; nNumberOfBytesToWrite
push    [ebp+lpAddress]          ; lpBuffer
push    [ebp+hObject]            ; hFile
call    ds:WriteFile
cmp     ebx, esi
jz      short loc_4013F7

push    4000h                    ; dwFreeType
push    esi                      ; dwSize
push    [ebp+lpAddress]          ; lpAddress
```

100.00% (122,3129) (292,499) 000007D5 004013D5: sub_401223+1B2

LHL13 - Notepad

```
LH    yÿ        @                                              à   ı°ı          Ï! ıL
$        ÛÒ+ı¿"ET¿"ET¿"ET¿"DT›"ET<›ıT°"ET¿"ET¾"ET1›ıT¶"ET1›%T½"ET<›T¾"ETRich¿"E
ı  j <E¸P<M´QjŒÿıÒŒ %E˙f}˙ }ı3ÀèÞ    <U´fÄı%U¼<E¼<Hı<U¼ıJı%MÛ<E¼<Hı%MØ<UØ%UÌj <E´P
e x e     3 6 0 s d . e x e   Z h u D o n g F a n g Y u . e x e   3 6 0 r p
e x e     k n s d t r a y . e x e     k n s d . e x e     k n s d s v c . e
ııı èáı  fÄıéVÿÿÿ<...(øÿÿf8 ı..ıÿÿÿj <MÜQÿıHOŒ <å]ÄÌÌÌÌÌÌThreadProcRoutine() run %d
Module Path: %s  SSDT BaseAddress: 0x%8X, NumberOfServices: 0x%X SSDT RAW: 0
fÄıúı À%ÿÿþÿı"À<ıØþÿÿ<ıdOŒ <ı<Uümı‡ı À  ı ı"ÀÛé\ÿÿÿh4!ı èOı  fÄıëhD!ı èšı  fÄıf½Übÿÿ t
 ÆEÿ ÇEÕp-ı <EÔPèÿþÿÿı¶É…ÉuBhÀ'ı èÄı  fÄıhı(ı <ı@ı RèŠüÿÿ%EÕ<EÔPèÌþÿÿı¶É…Éuıh$(ı è'ı
<å]À<ÿU<ìfìı ÌÌÌÌÌÌÌÜ<ìfìıÇEø    è^ÿÿÿèYãÿÿ£ı@ı f=ı@ı  uÀèFüÿÿ£ı@ı f=ı@ı  t.üı À%ÿÿ

tlFreeUnicodeString  xıZwReadFile  †ıZwSetInformationFile  SıZwOpenFile  ÆıRtlA
```

Names window

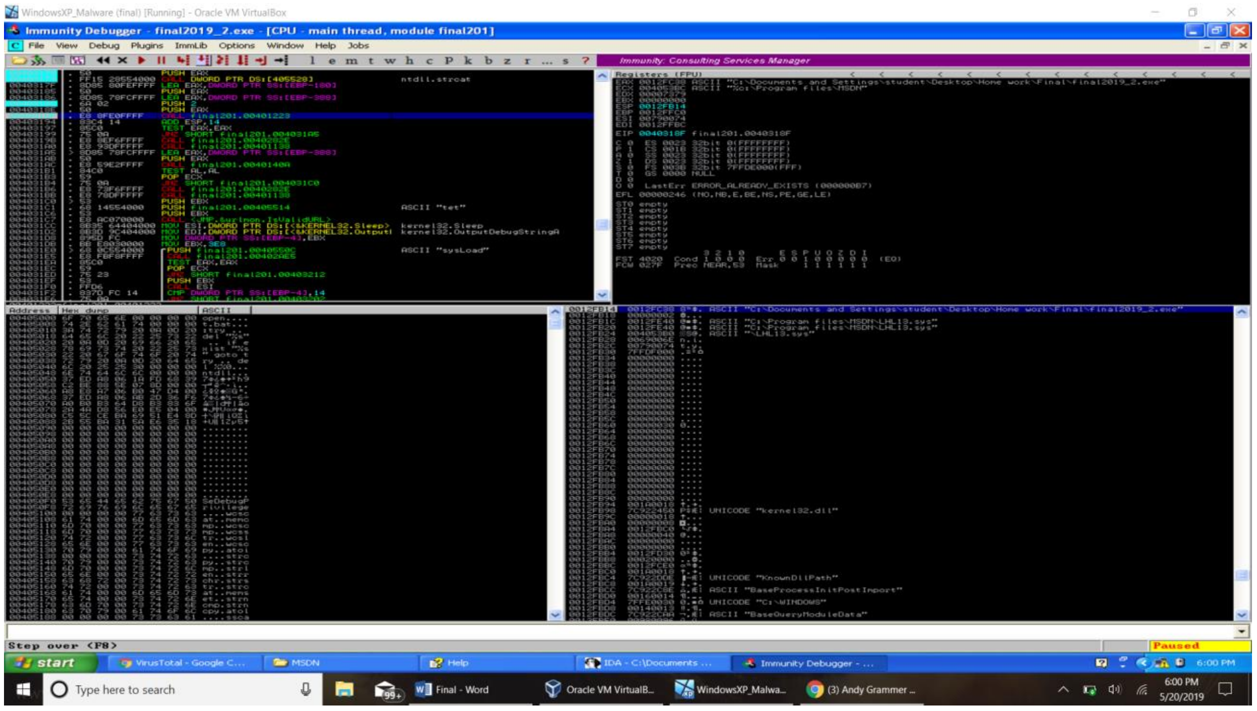
Name	
0000000C	C	CreateFileA
0000000A	C	WriteFile
0000000C	C	GetFileSize
00000010	C	UnmapViewOfFil

```
File 'C:\Documents and Settings\student\Desktop\Home work\Final\final2019_2.exe' is successfully loaded into the database.
Compiling file 'C:\Program Files\IDA Free\idc\ida.idc'...
Executing function 'main'...
Compiling file 'C:\Program Files\IDA Free\idc\onload.idc'...
Executing function 'OnLoad'...
IDA is analysing the input file...
You may start to explore the input file right now.
Propagating type information...
Function argument information is propagated
The initial autoanalysis has been finished.
```

AU: idle   Down   Disk: 14GB

start   VirusTotal - Google C...   Help   IDA - C:\Documents ...   Immunity Debugger - ...   LHL13 - Notepad   8:54 PM

Type here to search   Final - Word   Oracle VM VirtualB...   WindowsXP_Malwa...   (3) Egire Mabbulal...   8:54 PM 5/20/2019



WindowsXP_Malware (final) [Running] - Oracle VM VirtualBox

Immunity Debugger - final2019_2.exe - [CPU - main thread, module final201]

```
        . 50             PUSH EAX
0040317F. FF15 28554000  CALL DWORD PTR DS:[405528]            ntdll.strcat
00403185. 8D85 80FEFFFF  LEA EAX,DWORD PTR SS:[EBP-180]
00403186. 50             PUSH EAX
        . 8D85 78FCFFFF  LEA EAX,DWORD PTR SS:[EBP-388]
        . 6A 02          PUSH 2
        . 50             PUSH EAX
0040318F. E8 8FE0FFFF    CALL final201.00401223
00403194. 83C4 14        ADD ESP,14
00403197. 85C0           TEST EAX,EAX
00403199. 75 0A          JNZ SHORT final201.004031A5
0040319B. E8 8EF6FFFF    CALL final201.0040282E
004031A0. E8 93DFFFFF    CALL final201.00401138
004031A5> 8D85 78FCFFFF  LEA EAX,DWORD PTR SS:[EBP-388]
004031AB. 50             PUSH EAX
004031AC. E8 59E2FFFF    CALL final201.0040140A
004031B1. 84C0           TEST AL,AL
004031B3. 59             POP ECX
004031B4. 75 0A          JNZ SHORT final201.004031C0
004031B6. E8 73F6FFFF    CALL final201.0040282E
004031BB. E8 78DFFFFF    CALL final201.00401138
004031C0> 53             PUSH EBX
004031C1. 68 14554000    PUSH final201.00405514                ASCII "tet"
004031C6. 53             PUSH EBX
004031C7. E8 AC070000    CALL <JMP.&urlmon.IsValidURL>
004031CC. 8B35 64404000  MOV ESI,DWORD PTR DS:[<&KERNEL32.Sleep>   kernel32.Sleep
004031D2. 8B3D 9C404000  MOV EDI,DWORD PTR DS:[<&KERNEL32.OutputD  kernel32.OutputDebugStringA
004031D8. 895D FC        MOV DWORD PTR SS:[EBP-4],EBX
004031DB. BB E8030000    MOV EBX,3E8
004031E0> 68 0C554000    PUSH final201.0040550C                ASCII "sysLoad"
004031E5. E8 FBF8FFFF    CALL final201.00402AE5
004031EA. 85C0           TEST EAX,EAX
004031EC. 59             POP ECX
004031ED. 75 23          JNZ SHORT final201.00403212
004031EF. 53             PUSH EBX
004031F0. FFD6           CALL ESI
004031F2. 837D FC 14     CMP DWORD PTR SS:[EBP-4],14
```

```
Registers (FPU)
EAX 0012FC38 ASCII "C:\Documents and Settings\student\Desktop\Home work\Final\final2019_2.exe"
ECX 004053BC ASCII "%c:\Program files\MSDN"
EDX 00007379
EBX 00000000
ESP 0012FB14
EBP 0012FFC0
ESI 00790074
EDI 0012FFBC

EIP 0040318F final201.0040318F

C 0  ES 0023 32bit 0(FFFFFFFF)
P 1  CS 001B 32bit 0(FFFFFFFF)
A 0  SS 0023 32bit 0(FFFFFFFF)
Z 1  DS 0023 32bit 0(FFFFFFFF)
S 0  FS 003B 32bit 7FFDE000(FFF)
T 0  GS 0000 NULL
D 0
O 0  LastErr ERROR_ALREADY_EXISTS (000000B7)
EFL 00000246 (NO,NB,E,BE,NS,PE,GE,LE)
ST0 empty
ST1 empty
ST2 empty
ST3 empty
ST4 empty
ST5 empty
ST6 empty
ST7 empty
               3 2 1 0      E S P U O Z D I
FST 4020  Cond 1 0 0 0  Err 0 0 1 0 0 0 0 0  (EQ)
FCW 027F  Prec NEAR,53  Mask    1 1 1 1 1 1
```

```
Address   Hex dump                  ASCII
00405000  6F 70 65 6E 00 00 00 00   open....
00405008  74 2E 62 61 74 00 00 00   t.bat...
00405010  3A 74 72 79 20 0A 0D 20   :try ..
00405018  64 65 6C 20 22 25 73 22   del "%s"
00405020  20 0A 0D 20 69 66 20 65    .. if e
00405028  78 69 73 74 20 22 25 73   xist "%s
00405030  22 20 67 6F 74 6F 20 74   " goto t
00405038  72 79 20 0A 0D 20 64 65   ry .. de
00405040  6C 20 25 25 30 00 00 00   l %%0...
00405048  6E 74 64 6C 6C 00 00 00   ntdll...
```

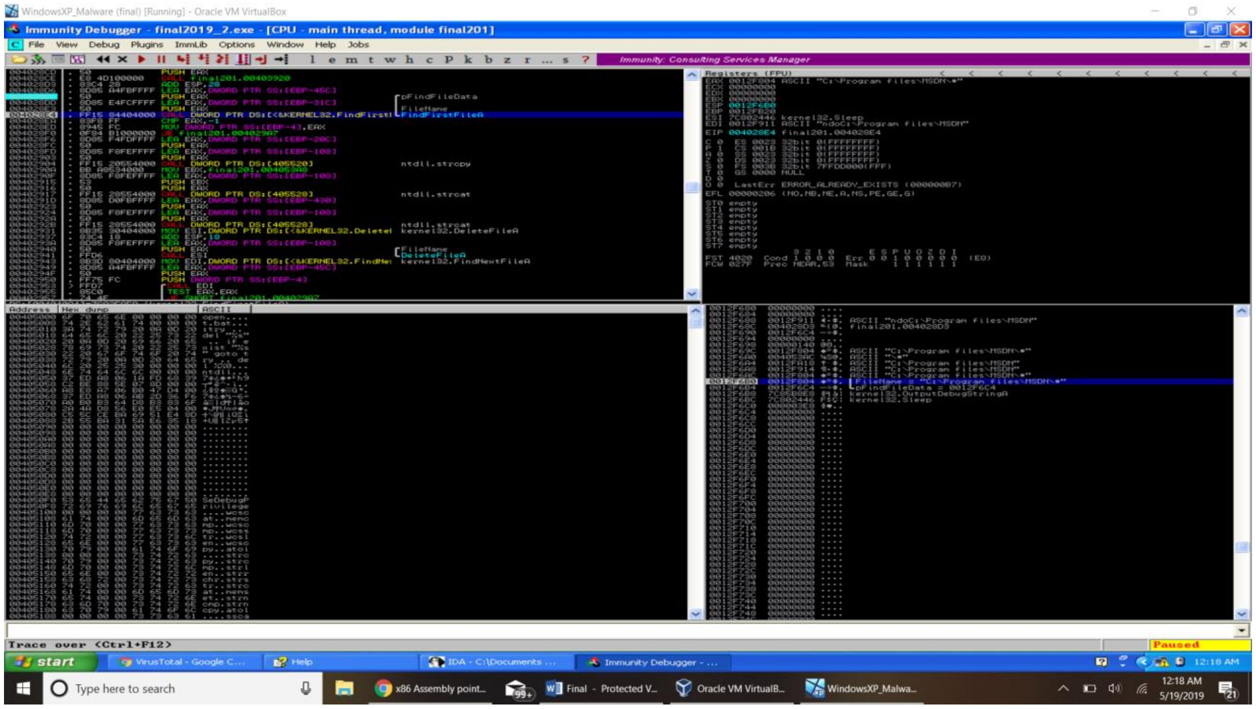
```
0012FB14  0012FC38  ASCII "C:\Documents and Settings\student\Desktop\Home work\Final\final2019_2.exe"
0012FB18  00000002
0012FB1C  0012FE40  ASCII "C:\Program files\MSDN\LHL13.sys"
0012FB20  0012FE40  ASCII "C:\Program files\MSDN\LHL13.sys"
0012FB24  004053B0  ASCII "\LHL13.sys"
0012FB28  0069006E  n.i.
0012FB2C  00790074  t.y.
0012FB30  7FFDF000
0012FB34  00000000  ....
0012FB38  00000000  ....
0012FB3C  00000000  ....
0012FB40  00000000  ....
0012FB44  00000000  ....
0012FB48  00000000  ....
0012FB4C  00000000  ....
0012FB50  00000000  ....
0012FB54  00000000  ....
0012FB58  00000000  ....
0012FB5C  00000000  ....
0012FB60  00000030  0...
0012FB64  00000000  ....
0012FB68  00000000  ....
0012FB6C  00000000  ....
0012FB70  00000000  ....
0012FB74  00000000  ....
0012FB78  00000000  ....
0012FB7C  00000000  ....
0012FB80  00000000  ....
0012FB84  00000000  ....
0012FB88  00000000  ....
0012FB8C  00000000  ....
0012FB90  00000000  ....
0012FB94  001A0018
0012FB98  7C922450  UNICODE "kernel32.dll"
0012FB9C  00000018
0012FBA0  00000005
0012FBA4  0012FBC0
0012FBA8  00000040  @...
0012FBAC  00000000  ....
0012FBB0  00000000  ....
0012FBB4  0012FD30
0012FBB8  00020000
0012FBBC  0012FCE0
0012FBC0  001A0018
0012FBC4  7C922DDE  UNICODE "KnownDllPath"
0012FBC8  001A0019
0012FBCC  7C922C8E  ASCII "BaseProcessInitPostImport"
0012FBD0  00160014
0012FBD4  7FFE0030  UNICODE "C:\WINDOWS"
0012FBD8  00140013
0012FBDC  7C922CAA  ASCII "BaseQueryModuleData"
```

Step over <F8>   Paused

start   VirusTotal - Google C...   MSDN   Help   IDA - C:\Documents ...   Immunity Debugger - ...   6:00 PM

Type here to search   Final - Word   Oracle VM VirtualB...   WindowsXP_Malwa...   (3) Andy Grammer ...   6:00 PM 5/20/2019

4) **What is the value of the first argument passed to the call to FindFirstFileA in sub_40282E? In a sentence or two, summarize what sub_40282E does. (8 pts)**
First Argument passed to the call to FindFirstFileA in sub_40282E is C:\Program files\MSDN\*
This function gets the first file in the folder MSDN using FindFirstFileA and then deletes it using DeleteFileA. Later it used FindNextFileA and deletes that file. It keeps doing that till there are no more files left in the folder MSDN.



```
004028CD  .  50              PUSH EAX
004028CE  .  E8 4D100000     CALL final201.00403920
004028D3  .  83C4 28         ADD ESP,28
004028D6  .  8D85 A4FBFFFF   LEA EAX,DWORD PTR SS:[EBP-45C]
004028DC  .  50              PUSH EAX                                  ┌pFindFileData
004028DD  .  8D85 E4FCFFFF   LEA EAX,DWORD PTR SS:[EBP-31C]            │
004028E3  .  50              PUSH EAX                                  │FileName
004028E4  .  FF15 84404000   CALL DWORD PTR DS:[<&KERNEL32.FindFirstI  └FindFirstFileA
004028EA  .  83F8 FF         CMP EAX,-1
004028ED  .  8945 FC         MOV DWORD PTR SS:[EBP-4],EAX
004028F0  .  0F84 B1000000   JE final201.004029A7
004028F6  .  8D85 F4FDFFFF   LEA EAX,DWORD PTR SS:[EBP-20C]
004028FC  .  50              PUSH EAX
004028FD  .  8D85 F8FEFFFF   LEA EAX,DWORD PTR SS:[EBP-108]
00402903  .  50              PUSH EAX
00402904  .  FF15 20554000   CALL DWORD PTR DS:[405520]                ntdll.strcpy
0040290A  .  BB A8534000     MOV EBX,final201.004053A8
0040290F  .  8D85 F8FEFFFF   LEA EAX,DWORD PTR SS:[EBP-108]
00402915  .  53              PUSH EBX
00402916  .  50              PUSH EAX
00402917  .  FF15 28554000   CALL DWORD PTR DS:[405528]                ntdll.strcat
0040291D  .  8D85 D0FBFFFF   LEA EAX,DWORD PTR SS:[EBP-430]
00402923  .  50              PUSH EAX
00402924  .  8D85 F8FEFFFF   LEA EAX,DWORD PTR SS:[EBP-108]
0040292A  .  50              PUSH EAX
0040292B  .  FF15 28554000   CALL DWORD PTR DS:[405528]                ntdll.strcat
00402931  .  8B35 30404000   MOV ESI,DWORD PTR DS:[<&KERNEL32.DeleteI  kernel32.DeleteFileA
00402937  .  83C4 18         ADD ESP,18
0040293A  .  8D85 F8FEFFFF   LEA EAX,DWORD PTR SS:[EBP-108]
00402940  .  50              PUSH EAX                                  ┌FileName
00402941  .  FFD6            CALL ESI                                  └DeleteFileA
00402943  .  8B3D 80404000   MOV EDI,DWORD PTR DS:[<&KERNEL32.FindNe:  kernel32.FindNextFileA
00402949  .  8D85 A4FBFFFF   LEA EAX,DWORD PTR SS:[EBP-45C]
0040294F  .  50              PUSH EAX
00402950  .  FF75 FC         PUSH DWORD PTR SS:[EBP-4]
00402953  >  FFD7            ┌CALL EDI
00402955  .  85C0            │TEST EAX,EAX
```

```
Address   Hex dump                 ASCII
00405000  6F 70 65 6E 00 00 00 00  open....
00405008  74 2E 62 61 74 00 00 00  t.bat...
00405010  3A 74 72 79 20 0A 0D 20  :try ..
00405018  64 65 6C 20 22 25 73 22  del "%s"
00405020  20 0A 0D 20 69 66 20 65   .. if e
00405028  78 69 73 74 20 22 25 73  xist "%s
00405030  22 20 67 6F 74 6F 20 74  " goto t
00405038  72 79 20 0A 0D 20 64 65  ry .. de
00405040  6C 20 25 25 30 00 00 00  l %%0...
00405048  6E 74 64 6C 6C 00 00 00  ntdll...
```

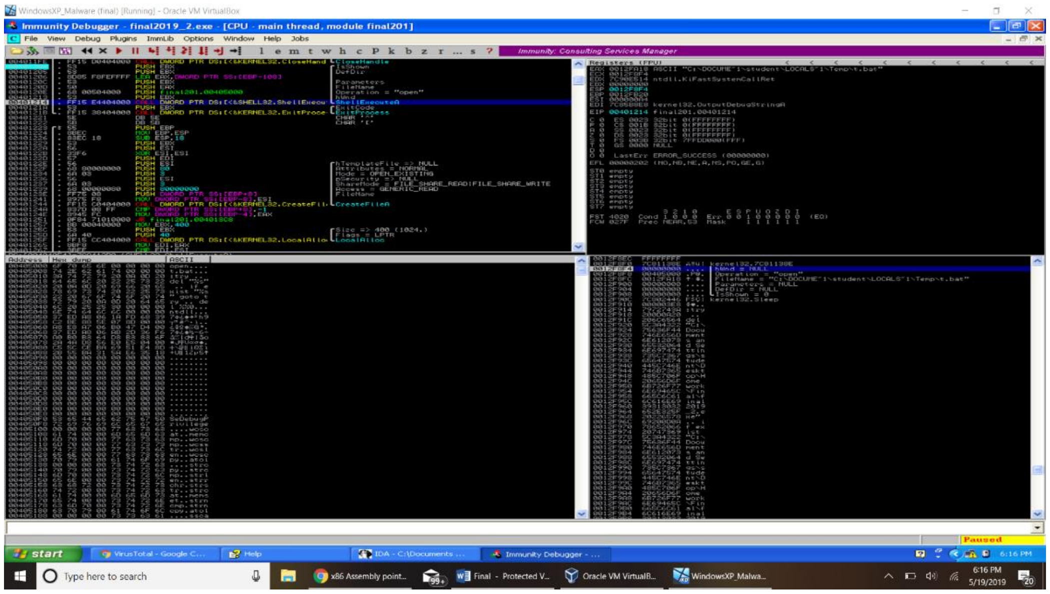
```
Registers (FPU)
EAX 0012F804 ASCII "C:\Program files\MSDN\*"
ECX 00000000
EDX 00000000
EBX 00000000
ESP 0012F6B0
EBP 0012FB20
ESI 7C802446 kernel32.Sleep
EDI 0012F911 ASCII "ndoC:\Program files\MSDN"
EIP 004028E4 final201.004028E4
C 0  ES 0023 32bit 0(FFFFFFFF)
P 1  CS 001B 32bit 0(FFFFFFFF)
A 0  SS 0023 32bit 0(FFFFFFFF)
Z 0  DS 0023 32bit 0(FFFFFFFF)
S 0  FS 003B 32bit 7FFDD000(FFF)
T 0  GS 0000 NULL
D 0
O 0  LastErr ERROR_ALREADY_EXISTS (000000B7)
EFL 00000206 (NO,NB,NE,A,NS,PE,GE,G)
ST0 empty
ST1 empty
ST2 empty
ST3 empty
ST4 empty
ST5 empty
ST6 empty
ST7 empty
               3 2 1 0      E S P U O Z D I
FST 4020  Cond 1 0 0 0  Err 0 0 1 0 0 0 0 0  (EQ)
FCW 027F  Prec NEAR,53  Mask    1 1 1 1 1 1
```

```
0012F680   00000000  ....
0012F684   00000000  ....
0012F688   0012F911  ◄·‡.  ASCII "ndoC:\Program files\MSDN"
0012F68C   004028D3  ╙(@.  final201.004028D3
0012F690   0012F6C4  ─÷‡.
0012F694   00000000  ....
0012F698   00000140  @@..
0012F69C   0012F804  ♦°‡.  ASCII "C:\Program files\MSDN\*"
0012F6A0   004053AC  ¼S@.  ASCII "\*"
0012F6A4   0012FA18  ↑·‡.  ASCII "C:\Program files\MSDN"
0012F6A8   0012F914  ¶·‡.  ASCII "C:\Program files\MSDN"
0012F6AC   0012F804  ♦°‡.  ASCII "C:\Program files\MSDN\*"
0012F6B0   0012F804  ♦°‡.  ┌FileName = "C:\Program files\MSDN\*"
0012F6B4   0012F6C4  ─÷‡.  └pFindFileData = 0012F6C4
0012F6B8   7C85B8E8  Φ╕à|  kernel32.OutputDebugStringA
0012F6BC   7C802446  F$Ç|  kernel32.Sleep
0012F6C0   000003E8  Φ♥..
0012F6C4   00000000  ....
0012F6C8   00000000  ....
0012F6CC   00000000  ....
0012F6D0   00000000  ....
0012F6D4   00000000  ....
0012F6D8   00000000  ....
0012F6DC   00000000  ....
0012F6E0   00000000  ....
0012F6E4   00000000  ....
0012F6E8   00000000  ....
0012F6EC   00000000  ....
0012F6F0   00000000  ....
0012F6F4   00000000  ....
0012F6F8   00000000  ....
0012F6FC   00000000  ....
0012F700   00000000  ....
0012F704   00000000  ....
```

5) What is the value of the third argument passed to ShellExecuteA at 0x401214? Why is the malware doing this? (8 pts)

The third argument passed to ShellExecuteA at 401214 is C:\DOCUME~1\student\LOCALS~1\Temp\t.bat.

It is passing a .bat file which is used to store command line instructions in plain text. Malware is using this t.bat file to pass instructions that will be executed in a shell.



WindowsXP_Malware (final) [Running] - Oracle VM VirtualBox

Immunity Debugger - final2019_2.exe - [CPU - main thread, module final201]

```
004011FE  .  FF15 D0404000  CALL DWORD PTR DS:[<&KERNEL32.CloseHand  CloseHandle
00401204  .  53             PUSH EBX                                 IsShown
00401205  .  53             PUSH EBX                                 DefDir
00401206  .  8D85 F8FEFFFF  LEA EAX,DWORD PTR SS:[EBP-108]
0040120C  .  53             PUSH EBX                                 Parameters
0040120D  .  50             PUSH EAX                                 FileName
0040120E  .  68 00504000    PUSH final201.00405000                   Operation = "open"
00401213  .  53             PUSH EBX                                 hWnd
00401214  .  FF15 E4404000  CALL DWORD PTR DS:[<&SHELL32.ShellExecu  ShellExecuteA
0040121A  .  53             PUSH EBX                                 ExitCode
0040121B  .  FF15 38404000  CALL DWORD PTR DS:[<&KERNEL32.ExitProce  ExitProcess
00401221     5E             DB 5E                                    CHAR '^'
00401222     5B             DB 5B                                    CHAR '['
00401223  $  55             PUSH EBP
00401224  .  8BEC           MOV EBP,ESP
00401226  .  83EC 18        SUB ESP,18
00401229  .  53             PUSH EBX
0040122A  .  56             PUSH ESI
0040122B  .  33F6           XOR ESI,ESI
0040122D  .  57             PUSH EDI
0040122E  .  56             PUSH ESI                                 hTemplateFile => NULL
0040122F  .  68 80000000    PUSH 80                                  Attributes = NORMAL
00401234  .  6A 03          PUSH 3                                   Mode = OPEN_EXISTING
00401236  .  56             PUSH ESI                                 pSecurity => NULL
00401237  .  6A 03          PUSH 3                                   ShareMode = FILE_SHARE_READ|FILE_SHARE_WRITE
00401239  .  68 00000080    PUSH 80000000                            Access = GENERIC_READ
0040123E  .  FF75 08        PUSH DWORD PTR SS:[EBP+8]                FileName
00401241  .  8975 F8        MOV DWORD PTR SS:[EBP-8],ESI
00401244  .  FF15 C0404000  CALL DWORD PTR DS:[<&KERNEL32.CreateFil  CreateFileA
0040124A  .  837D 08 FF     CMP DWORD PTR SS:[EBP+8],-1
0040124E  .  8945 FC        MOV DWORD PTR SS:[EBP-4],EAX
00401251  .  0F84 71010000  JE final201.004013C8
00401257  .  BB 00040000    MOV EBX,400
0040125C  .  53             PUSH EBX                                 Size => 400 (1024.)
0040125D  .  6A 40          PUSH 40                                  Flags = LPTR
0040125F  .  FF15 CC404000  CALL DWORD PTR DS:[<&KERNEL32.LocalAllo  LocalAlloc
00401265  .  8BF8           MOV EDI,EAX
00401267  .  3BFE           CMP EDI,ESI
```

```
Registers (FPU)
EAX 0012FA18 ASCII "C:\DOCUME~1\student\LOCALS~1\Temp\t.bat"
ECX 0012F8F4
EDX 7C90E514 ntdll.KiFastSystemCallRet
EBX 00000000
ESP 0012F8F4
EBP 0012FB20
ESI 000000A4
EDI 7C85B8E8 kernel32.OutputDebugStringA
EIP 00401214 final201.00401214
C 0  ES 0023 32bit 0(FFFFFFFF)
P 0  CS 001B 32bit 0(FFFFFFFF)
A 0  SS 0023 32bit 0(FFFFFFFF)
Z 0  DS 0023 32bit 0(FFFFFFFF)
S 0  FS 003B 32bit 7FFDD000(FFF)
T 0  GS 0000 NULL
D 0
O 0  LastErr ERROR_SUCCESS (00000000)
EFL 00000202 (NO,NB,NE,A,NS,PO,GE,G)
ST0 empty
ST1 empty
ST2 empty
ST3 empty
ST4 empty
ST5 empty
ST6 empty
ST7 empty
               3 2 1 0      E S P U O Z D I
FST 4020  Cond 1 0 0 0  Err 0 0 1 0 0 0 0 0  (EQ)
FCW 027F  Prec NEAR,53  Mask    1 1 1 1 1 1
```

```
Address   Hex dump                  ASCII
00405000  6F 70 65 6E 00 00 00 00   open....
00405008  74 2E 62 61 74 00 00 00   t.bat...
00405010  3A 74 72 79 20 0A 0D 20   :try ..
00405018  64 65 6C 20 22 25 73 22   del "%s"
00405020  20 0A 0D 20 69 66 20 65    .. if e
00405028  78 69 73 74 20 22 25 73   xist "%s
00405030  22 20 67 6F 74 6F 20 74   " goto t
00405038  72 79 20 0A 0D 20 64 65   ry .. de
00405040  6C 20 25 25 30 00 00 00   l %%0...
00405048  6E 74 64 6C 6C 00 00 00   ntdll...
```

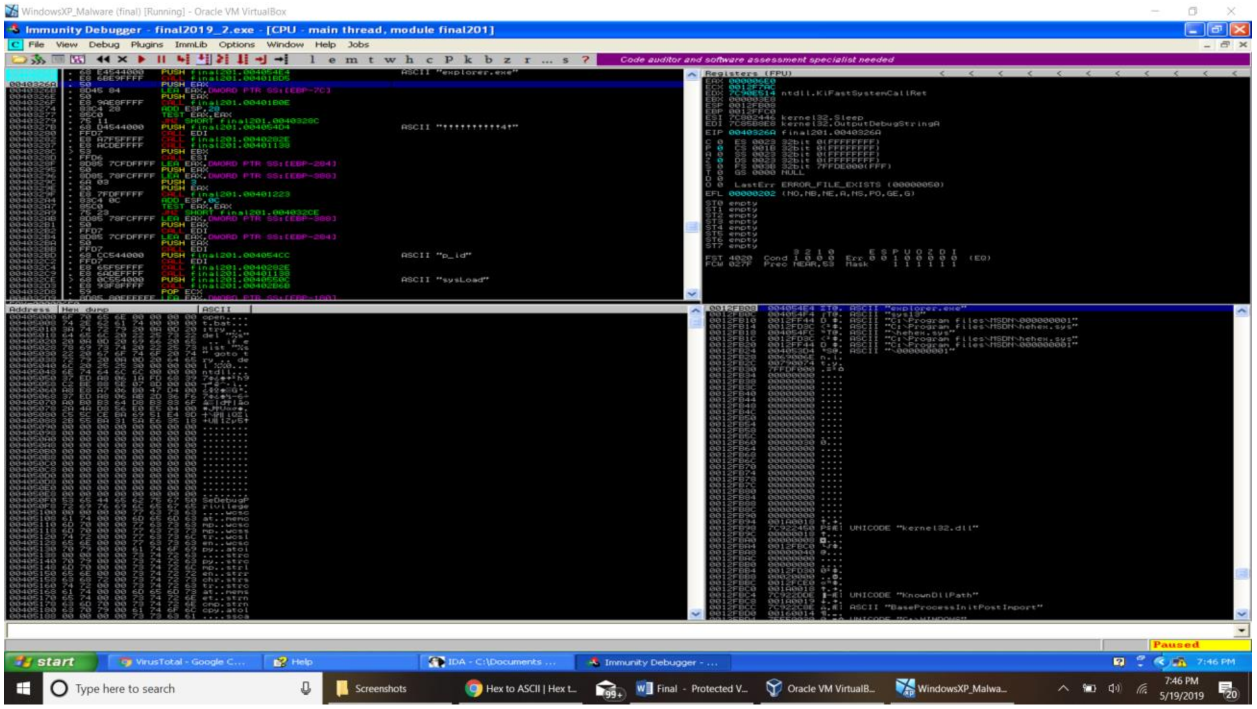
```
0012F8EC  FFFFFFFF
0012F8F0  7C81138E  kernel32.7C81138E
0012F8F4  00000000  ....  hWnd = NULL
0012F8F8  00405000  .P@.  Operation = "open"
0012F8FC  0012FA18  ..    FileName = "C:\DOCUME~1\student\LOCALS~1\Temp\t.bat"
0012F900  00000000  ....  Parameters = NULL
0012F904  00000000  ....  DefDir = NULL
0012F908  00000000  ....  IsShown = 0
0012F90C  7C802446  F$€|  kernel32.Sleep
0012F910  000003E8  è...
0012F914  7972743A  :try
0012F918  200D0A20   ..
0012F91C  206C6564  del
0012F920  5C3A4322  "C:\
0012F924  75636F44  Docu
0012F928  746E656D  ment
0012F92C  6E612073  s an
0012F930  65532064  d Se
0012F934  6E697474  ttin
0012F938  735C7367  gs\s
0012F93C  65647574  tude
0012F940  445C746E  nt\D
0012F944  746B7365  eskt
0012F948  485C706F  op\H
0012F94C  20656D6F  ome
0012F950  6B726F77  work
0012F954  6E69465C  \Fin
0012F958  665C6C61  al\f
0012F95C  6C616E69  inal
0012F960  39313032  2019
0012F964  652E325F  _2.e
0012F968  20226578  xe"
0012F96C  6920D0A0  .. i
0012F970  78652066  f ex
0012F974  20747369  ist
0012F978  5C3A4322  "C:\
```

Paused

start | VirusTotal - Google C... | Help | IDA - C:\Documents ... | Immunity Debugger - ... | 6:16 PM

Type here to search | x86 Assembly point... | Final - Protected V... | Oracle VM VirtualB... | WindowsXP_Malwa... | 6:16 PM 5/19/2019

6) **In a few sentences, describe what sub_401BD5 does. What does the function return? (7 pts)**
Creates a snapshot of all the processes (probably running in explorer.exe) using CreateToolHelp32Snapshot and then retrieves information about the first process in the snapshot using Process32First. Then it converts all the lower-case characters in upper case by doing so it also converts hem from Unicode to ansi. Later retrieves information about the next process in the snapshot and again converts in into uppercase. It repeats this process until it finishes all the processes in the snapshot and closes the handle.
Upon examining the eax register, which usually stores the return value of the function we can see 6E0 (in the screenshot below). This return value is a handle to the process that is of interest.


WindowsXP_Malware (final) [Running] - Oracle VM VirtualBox

Immunity Debugger - final2019_2.exe - [CPU - main thread, module final201]

```
         . 68 E4544000    PUSH final201.004054E4                ASCII "explorer.exe"
         . E8 6BE9FFFF    CALL final201.00401BD5
0040326A . 50             PUSH EAX
0040326B . 8D45 84        LEA EAX,DWORD PTR SS:[EBP-7C]
0040326E . 50             PUSH EAX
0040326F . E8 9AE8FFFF    CALL final201.00401B0E
00403274 . 83C4 20        ADD ESP,20
00403277 . 85C0           TEST EAX,EAX
00403279 . 75 11          JNZ SHORT final201.0040328C
0040327B . 68 D4544000    PUSH final201.004054D4                ASCII "???????????4?"
00403280 . FFD7           CALL EDI
00403282 . E8 A7F5FFFF    CALL final201.0040282E
00403287 . E8 ACDEFFFF    CALL final201.00401138
0040328C > 53             PUSH EBX
0040328D . FFD6           CALL ESI
0040328F . 8D85 7CFDFFFF  LEA EAX,DWORD PTR SS:[EBP-284]
00403295 . 50             PUSH EAX
00403296 . 8D85 78FCFFFF  LEA EAX,DWORD PTR SS:[EBP-388]
0040329C . 6A 03          PUSH 3
0040329E . 50             PUSH EAX
0040329F . E8 7FDFFFFF    CALL final201.00401223
004032A4 . 83C4 0C        ADD ESP,0C
004032A7 . 85C0           TEST EAX,EAX
004032A9 . 75 23          JNZ SHORT final201.004032CE
004032AB . 8D85 78FCFFFF  LEA EAX,DWORD PTR SS:[EBP-388]
004032B1 . 50             PUSH EAX
004032B2 . FFD7           CALL EDI
004032B4 . 8D85 7CFDFFFF  LEA EAX,DWORD PTR SS:[EBP-284]
004032BA . 50             PUSH EAX
004032BB . FFD7           CALL EDI
004032BD . 68 CC544000    PUSH final201.004054CC                ASCII "p_id"
004032C2 . FFD7           CALL EDI
004032C4 . E8 65F5FFFF    CALL final201.0040282E
004032C9 . E8 6ADEFFFF    CALL final201.00401138
004032CE > 68 0C554000    PUSH final201.0040550C                ASCII "sysLoad"
004032D3 . E8 93F8FFFF    CALL final201.00402B6B
004032D8 . 59             POP ECX
```

```
Registers (FPU)
EAX 000006E0
ECX 0012F7AC
EDX 7C90E514 ntdll.KiFastSystemCallRet
EBX 000003E8
ESP 0012FB08
EBP 0012FFC0
ESI 7C802446 kernel32.Sleep
EDI 7C85B8E8 kernel32.OutputDebugStringA

EIP 0040326A final201.0040326A

C 0  ES 0023 32bit 0(FFFFFFFF)
P 0  CS 001B 32bit 0(FFFFFFFF)
A 0  SS 0023 32bit 0(FFFFFFFF)
Z 0  DS 0023 32bit 0(FFFFFFFF)
S 0  FS 003B 32bit 7FFDE000(FFF)
T 0  GS 0000 NULL
D 0
O 0  LastErr ERROR_FILE_EXISTS (00000050)
EFL 00000202 (NO,NB,NE,A,NS,PO,GE,G)
ST0 empty
ST1 empty
ST2 empty
ST3 empty
ST4 empty
ST5 empty
ST6 empty
ST7 empty
               3 2 1 0         E S P U O Z D I
FST 4020  Cond 1 0 0 0   Err 0 0 1 0 0 0 0 0  (EQ)
FCW 027F  Prec NEAR,53   Mask    1 1 1 1 1 1
```

```
Address   Hex dump                  ASCII
00405000  6F 70 65 6E 00 00 00 00   open....
00405008  74 2E 62 61 74 00 00 00   t.bat...
00405010  3A 74 72 79 20 0A 0D 20   :try ..
00405018  64 65 6C 20 22 25 73 22   del "%s"
00405020  20 0A 0D 20 69 66 20 65    .. if e
00405028  78 69 73 74 20 22 25 73   xist "%s
00405030  22 20 67 6F 74 6F 20 74   " goto t
00405038  72 79 20 0A 0D 20 64 65   ry .. de
00405040  6C 20 25 25 30 00 00 00   l %%0...
00405048  6E 74 64 6C 6C 00 00 00   ntdll...
```

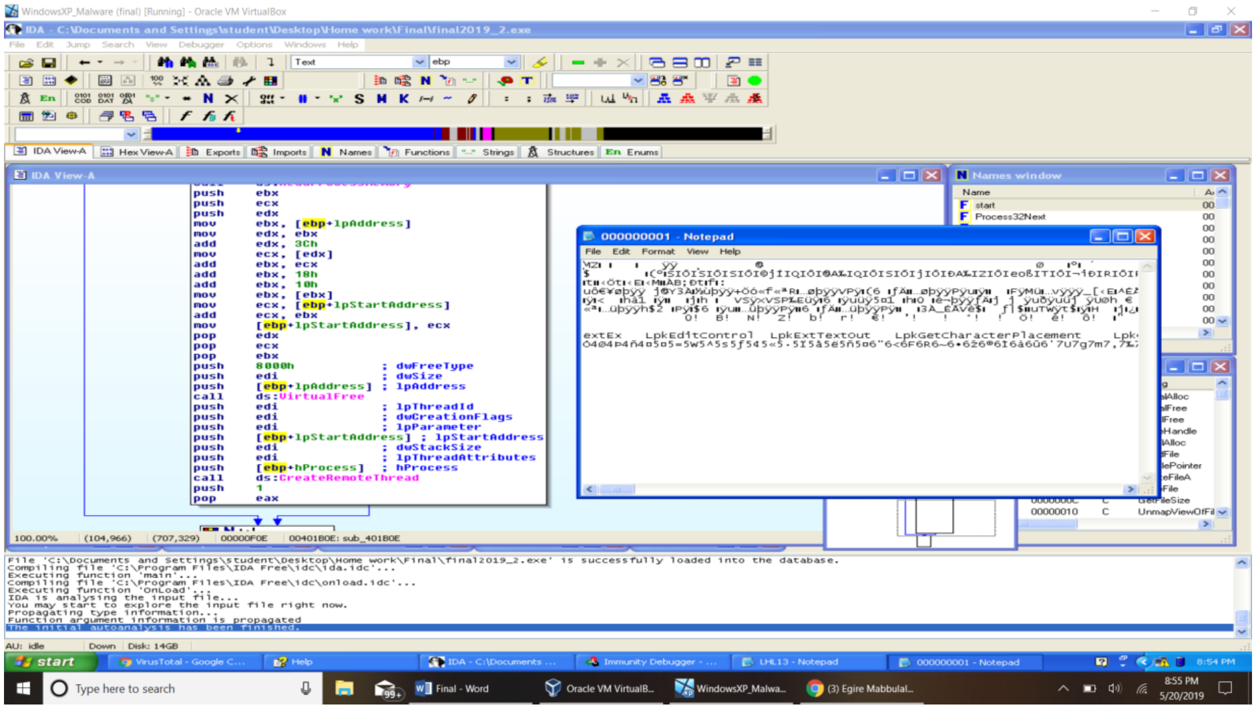
```
0012FB08  004054E4  ΣT@.  ASCII "explorer.exe"
0012FB0C  004054F4  ⌠T@.  ASCII "sys13"
0012FB10  0012FF44  D ‡.  ASCII "C:\Program Files\MSDN\000000001"
0012FB14  0012FD3C  <²‡.  ASCII "C:\Program Files\MSDN\hehex.sys"
0012FB18  004054FC  ⁿT@.  ASCII "\hehex.sys"
0012FB1C  0012FD3C  <²‡.  ASCII "C:\Program Files\MSDN\hehex.sys"
0012FB20  0012FF44  D ‡.  ASCII "C:\Program Files\MSDN\000000001"
0012FB24  004053D4  ╘S@.  ASCII "\000000001"
0012FB28  0069006E  n.i.
0012FB2C  00790074  t.y.
0012FB30  7FFDF000
```

UNICODE "kernel32.dll"
UNICODE "KnownDllPath"
ASCII "BaseProcessInitPostImport"

Paused

7) What kind of malicious behavior does the malware perform in the function sub_401B0E? Justify your answer in a sentence or two. (8 pts)

The function sub_401B0E first calls two more function sub_4019E8 (which is discussed in question 1) and sub_402712, which pushes many predefined C function and calls them. My guess is it gathers information about the files in question (on using Immunity Debugger we can see the file is 000000001.sys). Later it opens a process and calls one more function sub_401662 which allots virtual memory, creates a files, writes on the allotted virtual memory, reads it and then frees it. And if it the write was successful it will then allot virtual memory(again), read it (again) create a remote thread to execute it. Upon inspecting the file I could find that 000000001.sys is a binary file which contains code. So writing, reading and creating remote thread makes sense because it is trying to execute the binary.



```
push    ebx
push    ecx
push    edx
mov     ebx, [ebp+lpAddress]
mov     edx, ebx
add     edx, 3Ch
mov     ecx, [edx]
add     ebx, ecx
add     ebx, 18h
add     ebx, 10h
mov     ebx, [ebx]
mov     ecx, [ebp+lpStartAddress]
add     ecx, ebx
mov     [ebp+lpStartAddress], ecx
pop     edx
pop     ecx
pop     ebx
push    8000h           ; dwFreeType
push    edi             ; dwSize
push    [ebp+lpAddress] ; lpAddress
call    ds:VirtualFree
push    edi             ; lpThreadId
push    edi             ; dwCreationFlags
push    edi             ; lpParameter
push    [ebp+lpStartAddress] ; lpStartAddress
push    edi             ; dwStackSize
push    edi             ; lpThreadAttributes
push    [ebp+hProcess]  ; hProcess
call    ds:CreateRemoteThread
push    1
pop     eax
```

```
100.00%   (104,966)   (707,329)   00000F0E   00401B0E: sub_401B0E
```

```
extEx   LpkEditControl   LpkExtTextOut   LpkGetCharacterPlacement
```

```
File 'C:\Documents and Settings\student\Desktop\Home work\Final\final2019_2.exe' is successfully loaded into the database.
Compiling file 'C:\Program Files\IDA Free\idc\ida.idc'...
Executing function 'main'...
Compiling file 'C:\Program Files\IDA Free\idc\onload.idc'...
Executing function 'OnLoad'...
IDA is analysing the input file...
You may start to explore the input file right now.
Propagating type information...
Function argument information is propagated
The initial autoanalysis has been finished.
```