Computer Science 491/691
Malware Analysis
Homework 5
Assigned: April 10, 2019
Due: April 17, 2019

Submitted by: Sai Sasaank Srivatsa Pallerla
ID: HG13015

How to turn this in for grading: You can edit your answers right into this file. Email it to the TAs as described in class. Make sure your name appears in the body of the document.

Hint: Chapters 9, 11, and 15 of your Practical Malware Analysis textbook are useful references! In addition, the "Malware Behavior and the Windows API" PowerPoint will be helpful.

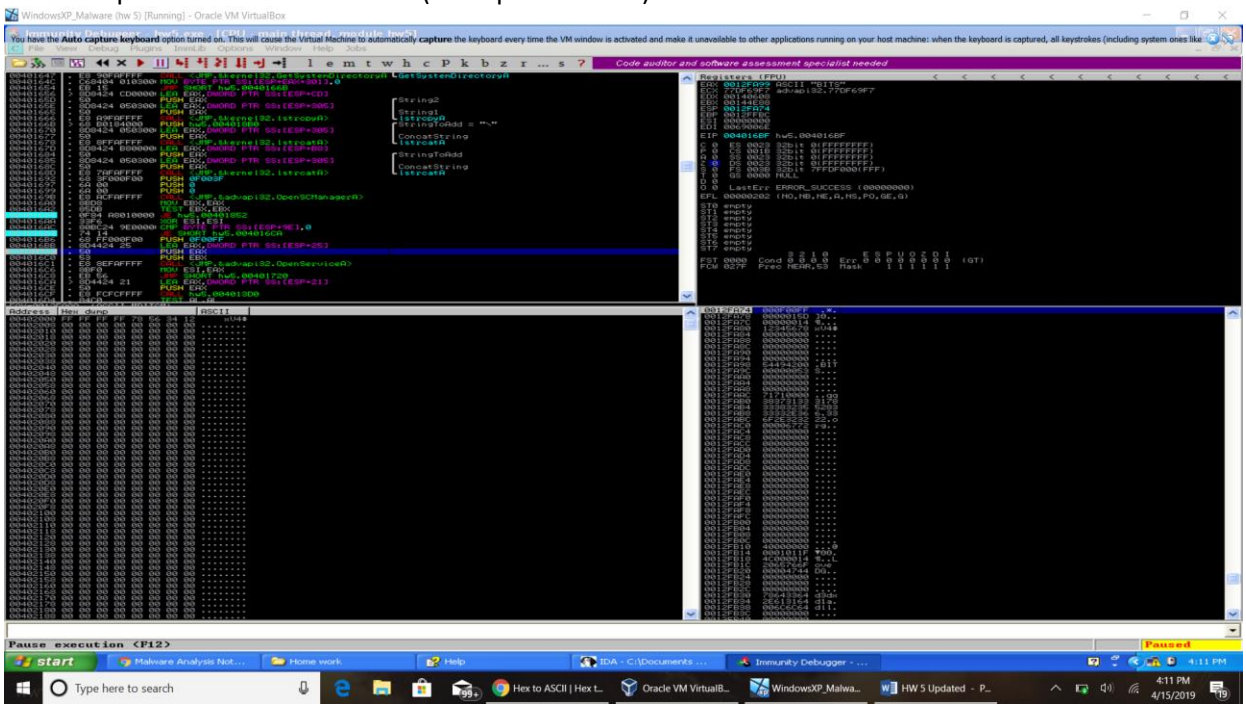
Download hw5.7z and extract it. The password is "infected". Answer the following questions:

1) **What is unusual about the assembly code from 0x4019AA to 0x4019C1? Why might the malware be doing this? (10 pts)**

4019AA is the start function: The programs execution starts here. We can see that the malware is adding data to the stack pointer and pushing some data into the edx register, then immediately popping the stack and register. At the end of the function we can see two jump statements JZ and JNZ which jump to the same address, i.e., 40198C. By looking at these jump statements, we can say the malware wants to execute the code/call the function at 40198C.
Later at 40198C we can see the function 401520 is enclosed between two null-functions. Few more things to notice is that all of the assembly instruction are in .CODE segment and the main (start) is in the end. The malware might be doing this throw off the antivirus or distract the analyst.

2) What is the value of lpServiceName when the malware calls OpenServiceA at 0x4016C1? (5 pts)

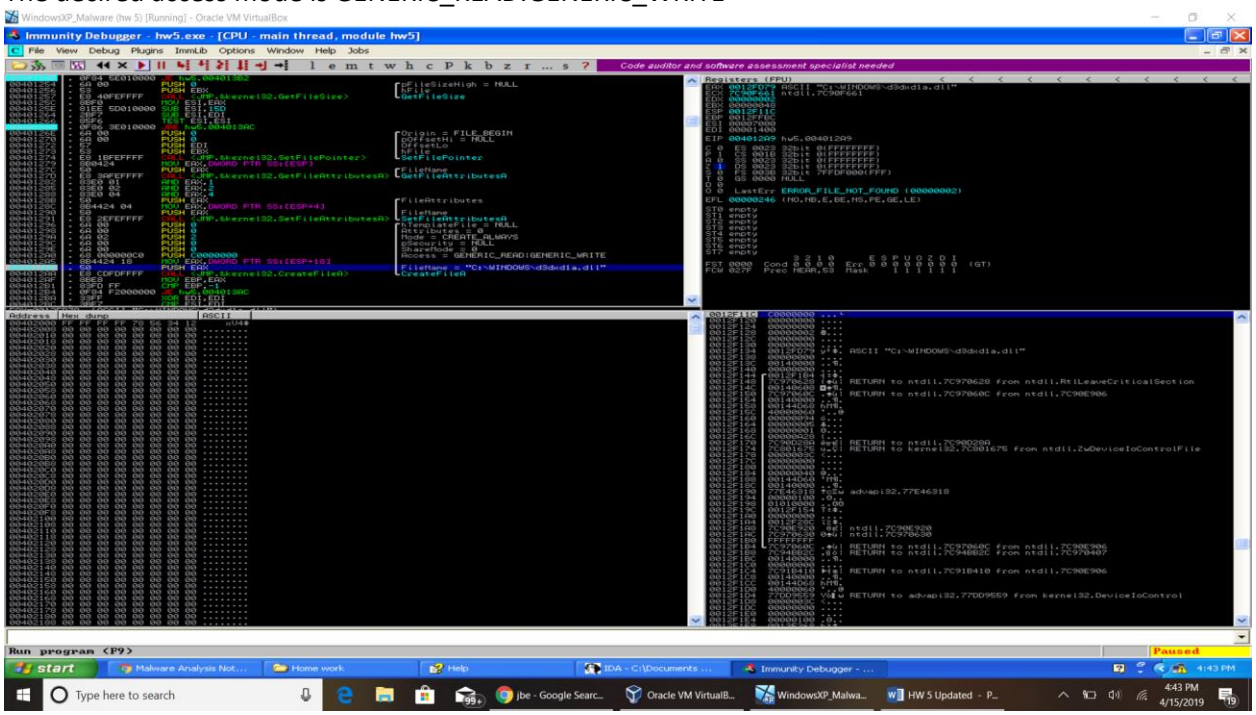
Value of lpServiceName at 4016C1 (call OpenserviceA) is BITS



3) What is the value of the <u>first</u> argument to the call to CreateFileA at 0x4012AA? What is the desired access mode? (Expecting the name of the access mode, not an integer) (5 pts)

The value of the first argument at 4012AA (call CreateFileA): d3dxd1a.dll

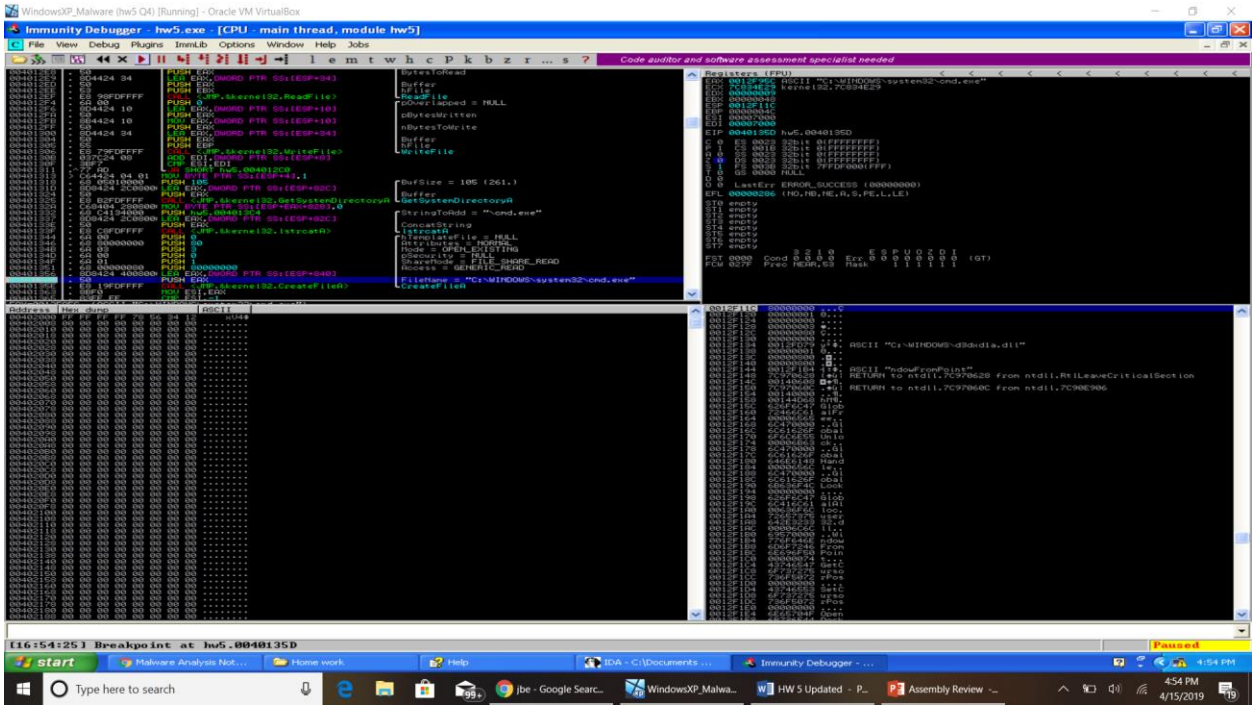
The decired access mode is GENERIC_READ:GENERIC_WRITE



4) What is the value of the <u>first</u> argument to the call to CreateFileA at 0x40135E? What is the desired access mode? (Expecting the name of the access mode, not an integer) (5 pts)

The value of the first argument at 40135E (call CreateFileA): cmd.exe

The decired access mode is GENERIC_READ



5) What is happening from 0x40136A to 0x401395? (12 pts)

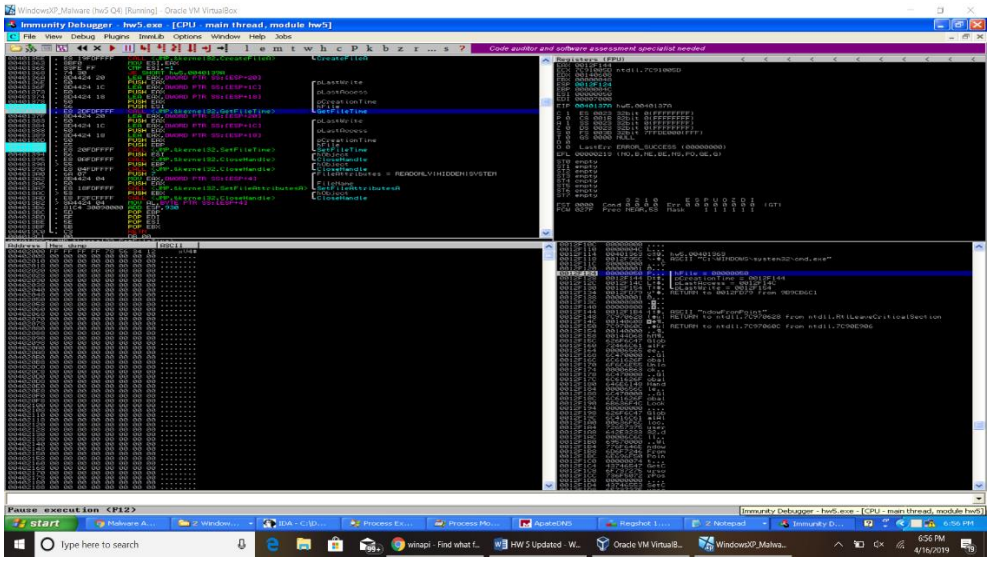
At 40137A the malware calls GetFileTime for which the parameters are

LastWrite: 0012F154

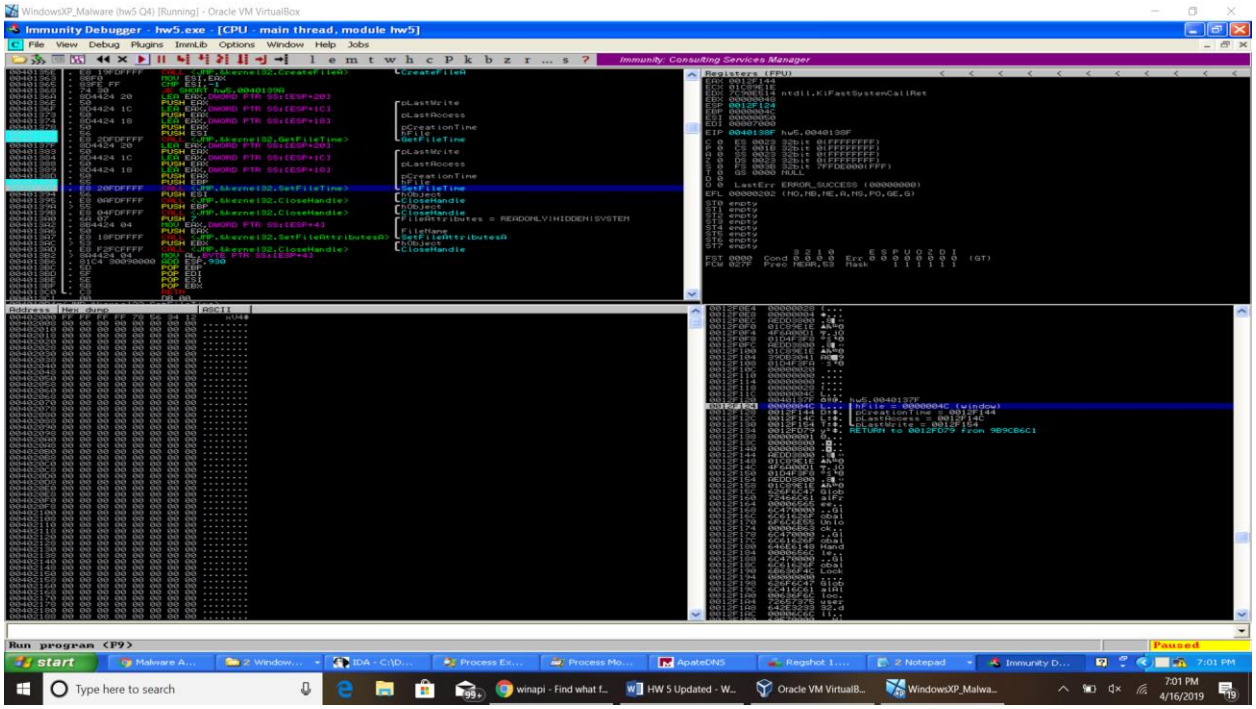
LastAccess: 0012F14C

CreationTime: 0012F144

hfile: 50



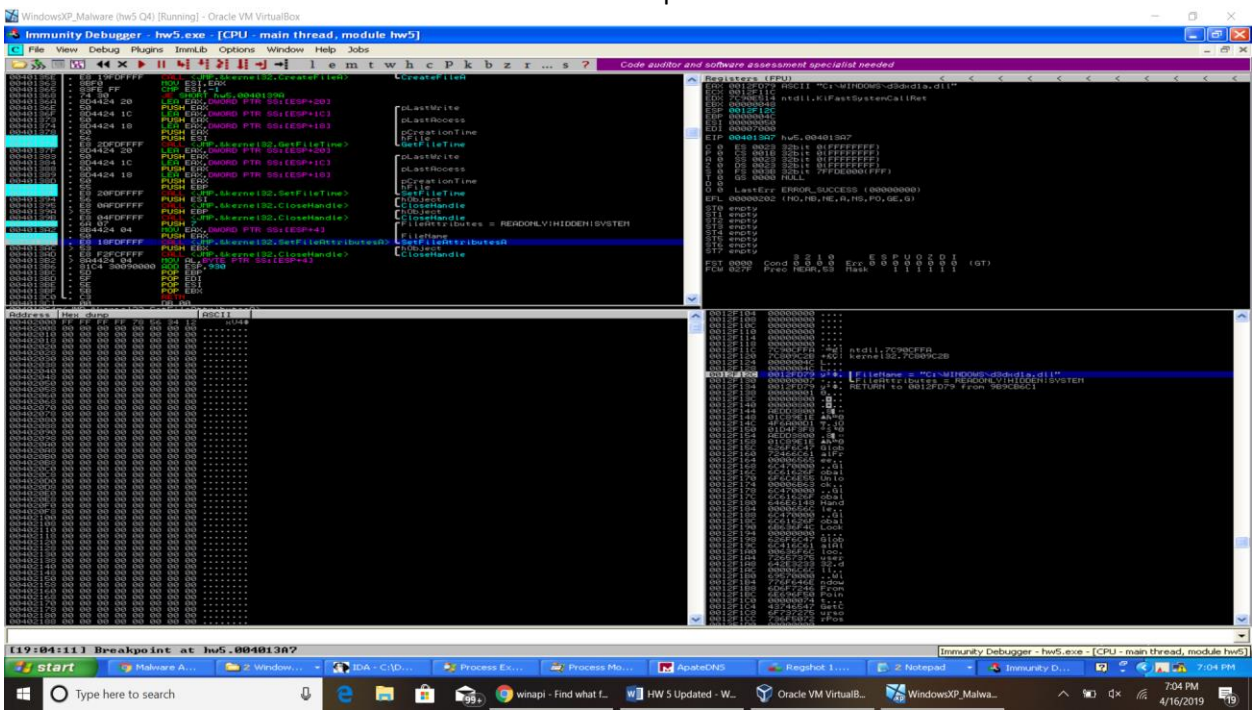
At 40138F the malware calls SetFileTime for which the parameters are
LastWrite: 0012F154
LastAccess: 0012F14C
CreationTime: 0012F144
hfile: 4C



By looking the function calls GetFileTime, SetFileTIme and the parameters that are being passed into it, we can say that the malware is setting the write, access and creation time of one file (whose file handle is 50) to another file (whose file handle is 4C). The malware might be doing this to cover its tracks, i.e., by changind when a file was created, written and last accessed.

6) What is the value of the <u>second</u> argument to the call to SetFileAttributesA at 0x4013A7? What is the name of the file that SetFileAttributesA modifies, and why is this significant? (Expecting the names of the attributes, not an integer) (10 pts)

The value of second argument at 4013A7 (call SetFileAttributesA) is READONLY:HIDDEN:SYSTEM
Name of the file that SetAttributesA modifies is d3dxd1a.dll
Considering the modified value, READONLY:HIDDEN:SYSTEM, and the file, Dll, its not suspicious that its READONLY because we don't want anybody changing or adding function in the library, but the suspicious part is the HIDDEN:SYSTEM. Any legitimate program wouldn't try to hide the executioin of a Dll which makes d3dxd1a.dll and the modification suspicious.



7) In a few sentences, describe what sub_4011B4 does. (12 pts)

This function is responsible for creating registry keys (RegCreateKey - Creates the specified registry key. If the key already exists, the function opens it.) and setting values to the registry key (RegSetValue - Sets the data and type of a specified value under a registry key).
RegCreateKey

Syntax

C++ Copy

```
LSTATUS RegCreateKeyExA(
  HKEY                        hKey,
  LPCSTR                      lpSubKey,
  DWORD                       Reserved,
  LPSTR                       lpClass,
  DWORD                       dwOptions,
  REGSAM                      samDesired,
  const LPSECURITY_ATTRIBUTES lpSecurityAttributes,
  PHKEY                       phkResult,
  LPDWORD                     lpdwDisposition
);
```

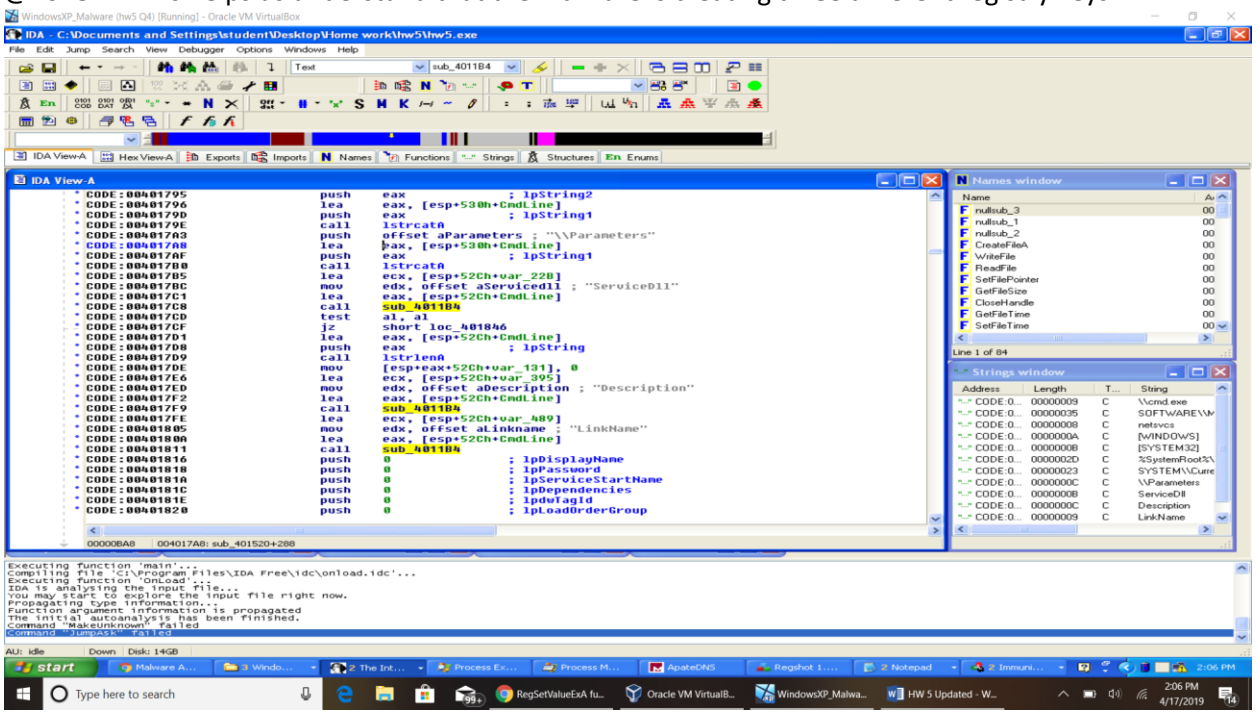
RegSetValue

Syntax

C++ Copy

```
LSTATUS RegSetValueExA(
  HKEY       hKey,
  LPCSTR     lpValueName,
  DWORD      Reserved,
  DWORD      dwType,
  const BYTE *lpData,
  DWORD      cbData
);
```

This function take two input parameters hkey and dwDisposition. When call this function hkey is stored in ecx and dwDispositon is stored in eax. This function is called three times @4017C8 @4017F9 @401811. This helps us understand that the malware is creating three different registry keys.



8) What is the value of the <u>second</u> argument to RegSetValueExA when sub_4011B4 is called at the following locations? (9 pts)

a) At 0x4017C8?: ServiceDll

b) At 0x4017F9?: Description

c) At 0x401811?: LinkName

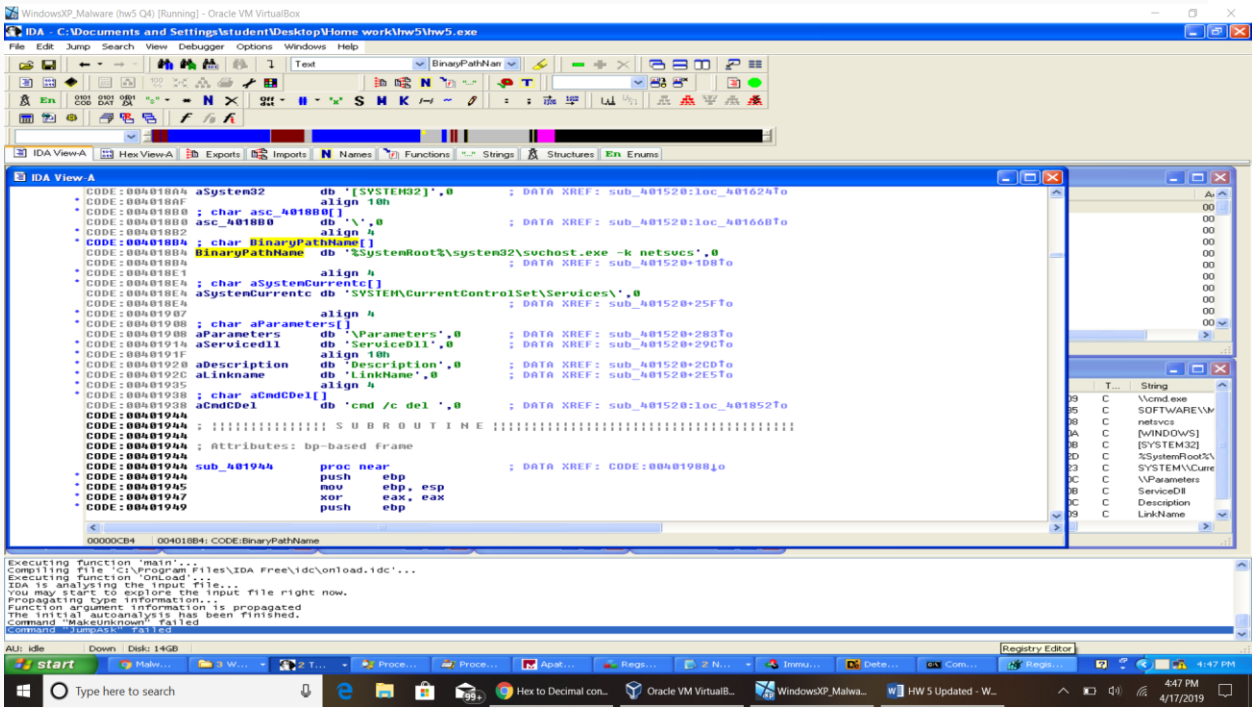
9) How does the malware gain persistence? (8 pts)

HKLM\System\CurrentControlSet\Services\SharedAccess

The keys located here get loaded by the Service Controller at various times during the operation of the computer. Some are loaded at system startup and others are loaded on demand or when triggered by other events. The attackers want to load at startup so that even if no user logs in they can connect to the computer.This is where the "Share Process" DLLs are configured to be loaded by the host process, SvcHost.exe. This has been the most common location for attackers to locate their remote access tools, often called "RATs" by response investigators.

Installing malware for persistence as an *svchost.exe* DLL makes the malware blend into the process list and the registry better than a standard service.

The malware to blend in the network, uses the netsvcs. We get this by this value by looking at %SystemRoot%/System32/svchost.exe –k GroupName, where group name is netsvcs.

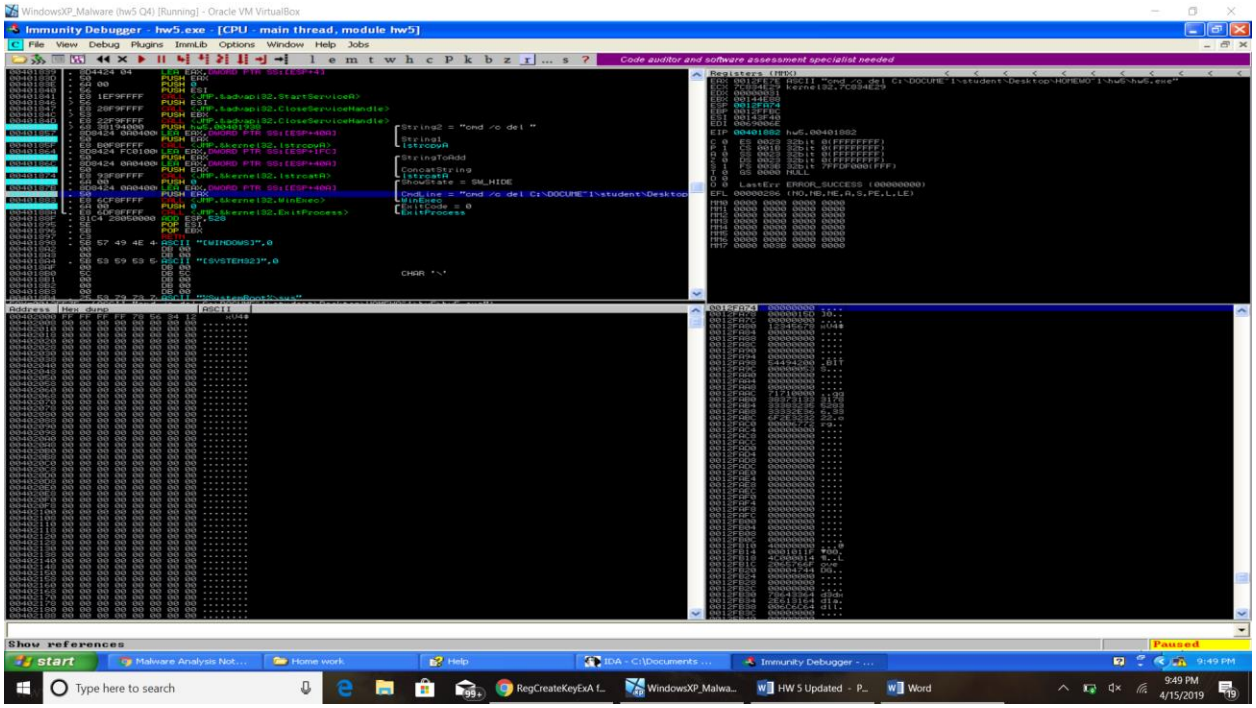


10) What is the value of the <u>first</u> argument to WinExec at 0x401883? What does the call to WinExec do? (9 pts)

First Argument at 401883 (call WinExec) is

"cmd /c del C:\Student\Dersktop\HOMEWORK\hw5\hw5.exe"

This command when entered in command prompt deletes hw5.exe



11) Investigate the file created during question 3 (Hint: The file may have some unusual attributes that make it difficult to find). Using any type of analysis you wish, list 3 <u>malicious</u> behaviors you suspect this file can perform. Justify your answers. <u>Extra credit will be awarded if very good analysis is provided.</u> (15 pts)

a) Process Injection Taking a look at the imports in Dll, we can see VirtualAlloc, WriteProcessMemory and CreateRemoteThread. VirtualAlloc - Allocates spaces in an external process memory, WriteProcessMemory - Write data to allocated space (can be executed as a thread), CreateRemoteThread - Creates thread out of the written data and executes it as a thread. This combination gives away that the malware is performing process injection.

b) Polling Keylogger Taking a look at the imports in the Dll, we can see few functions – keybd_event, mouse_event, SetCursorpos, GetCursorpos, MapVirtualKey. Mouse_event is used to synthesize mouse events by applications that need to do so. It is also used by applications that need to obtain more information from the mouse than its position and button state. Using Keybd_event an application can simulate a press of the PRINTSCRN key to obtain a screen snapshot and save it to the clipboard.
Looking at the functions we can say that the malware is trying to simulate mouse clicks and press of keyboardkeys and it is all running in hidden mode without giving any access to other file. Thus, considered as malicious behavior.

c) RunTime Linking Taking a look at the imports in Dll, we can see few function – LadLibrary and GetProcAdress. LoadLibrary loads the address of the Dll into the memory and GetProcAddress gets the address of the function from Dll in memory. So this combination of imports show that

the malware is performing runtime linking.