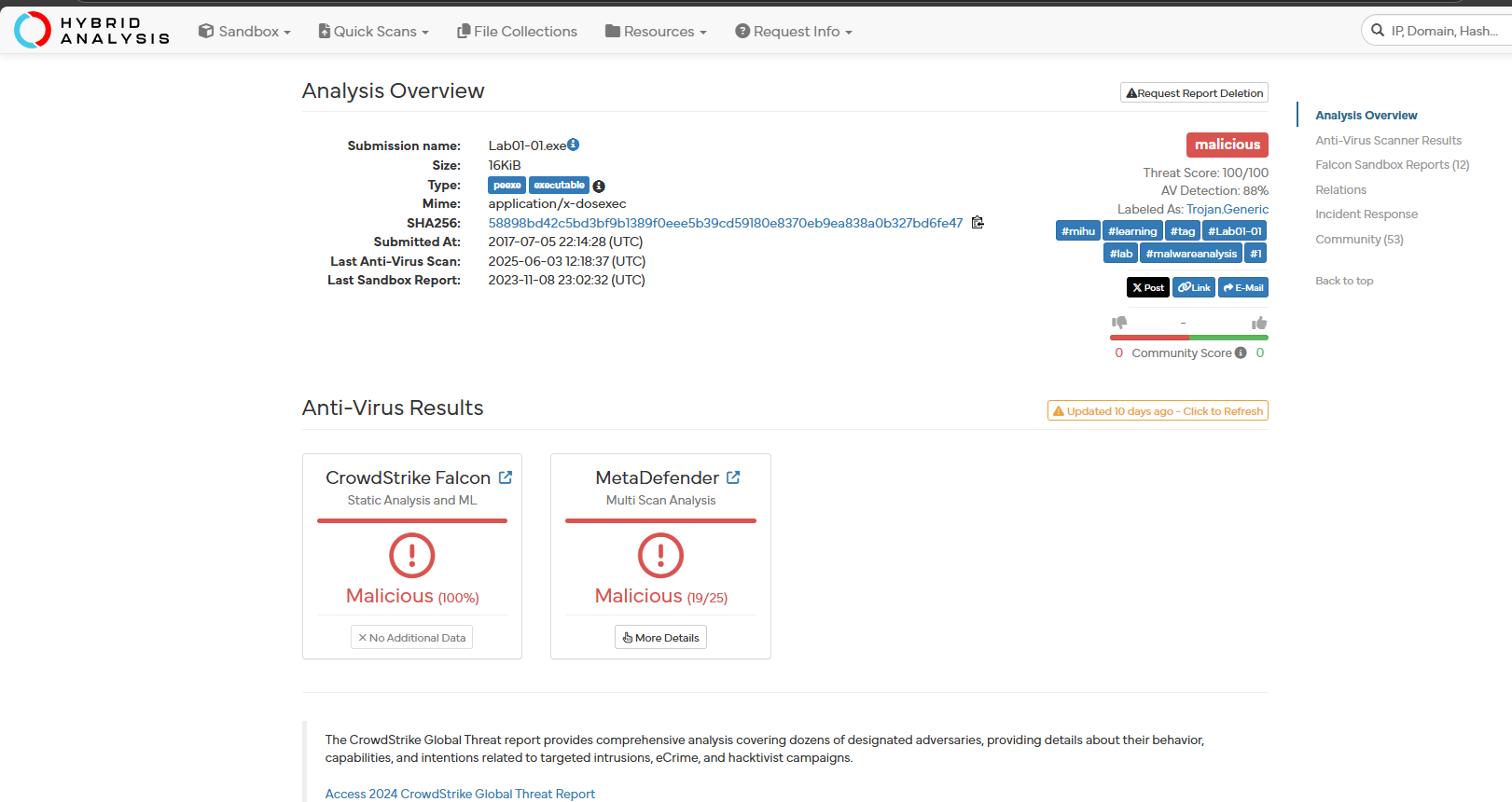
**NGUYEN NAM KHANH – HE191159 – IA1902 – IAM302**

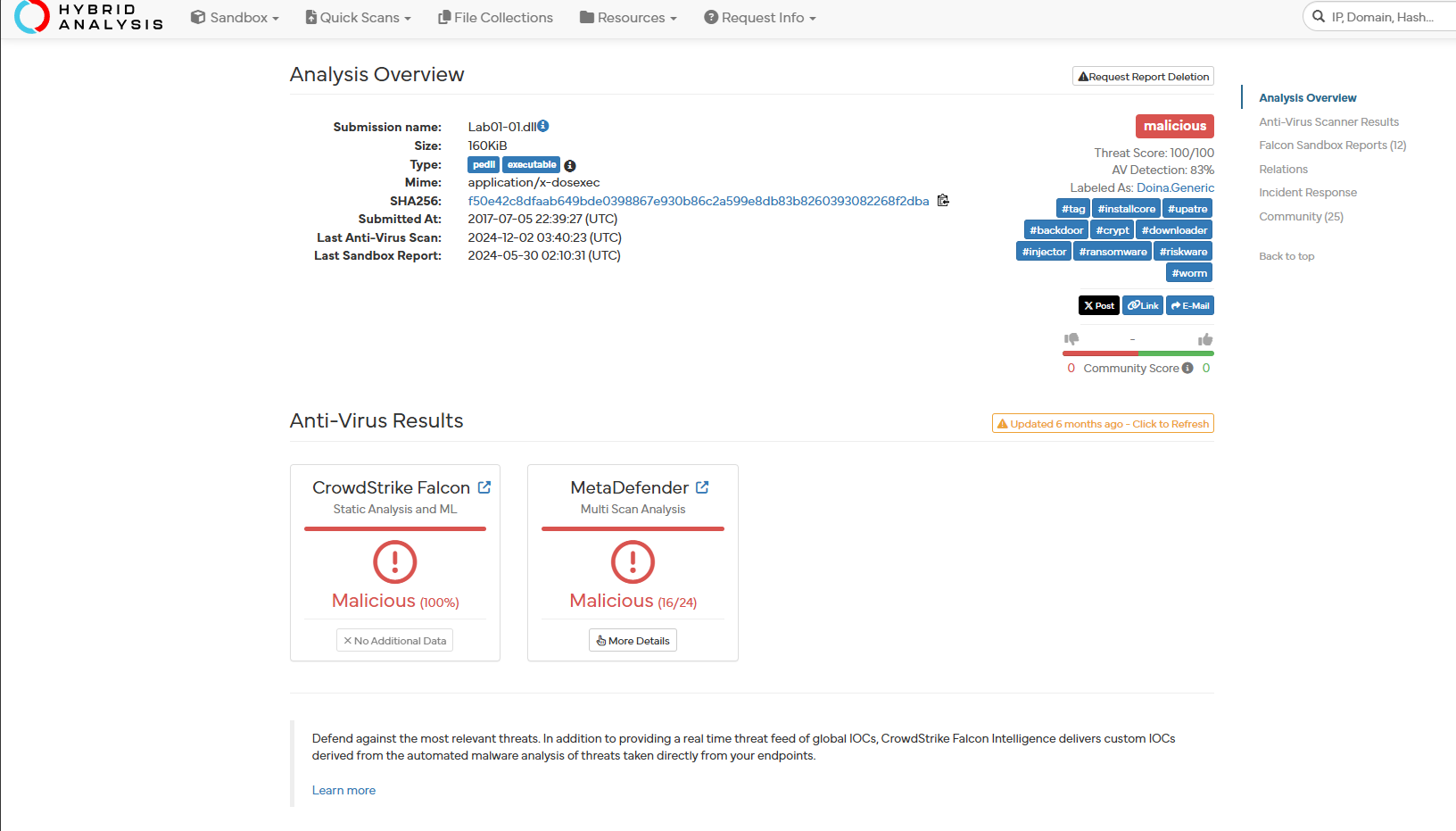
**LAB 8 Configuring a Malware Lab**

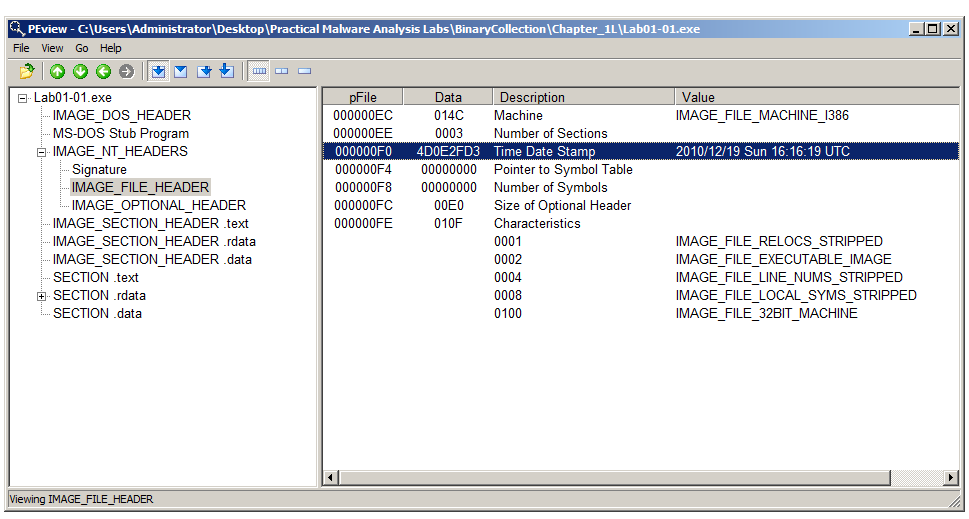
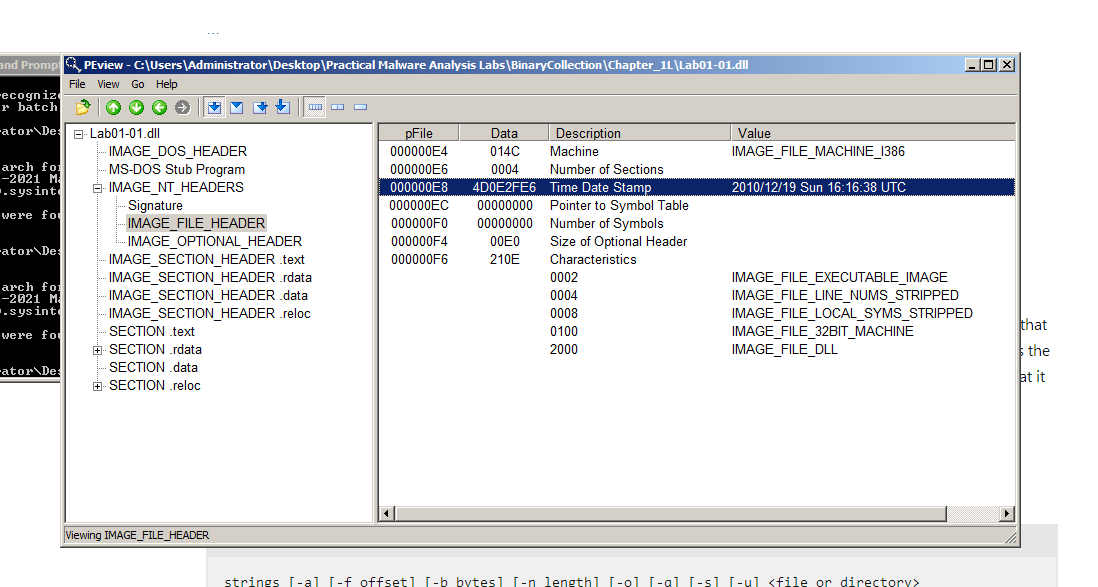
**Part 1: Basic Static Techniques**

**a. Analyzing Lab01-01.exe and Lab01-01.dll**

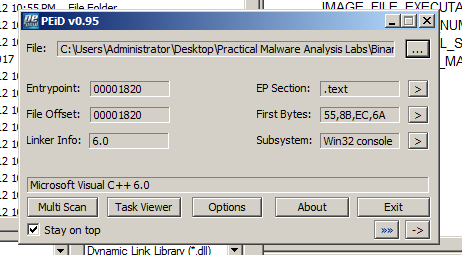
Using Hybrid Analysis for Lab01-01.exe



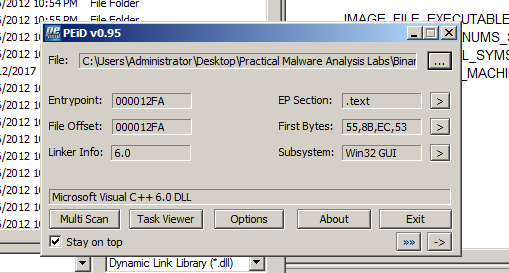
Lab01-01.dll

Using PEView for Lab01-01.exe  Lab01-01.dll 

Using PEiD for Lab01-01.exe



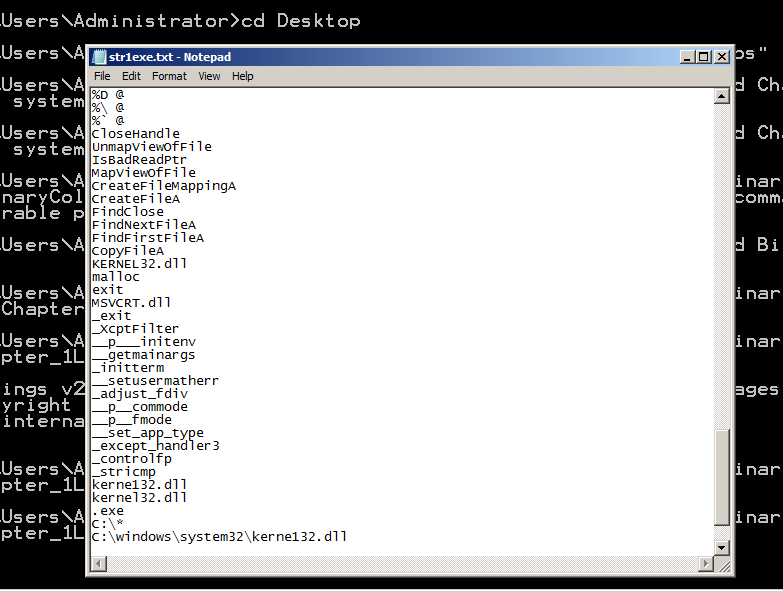
Lab01-01.dll



Using Strings

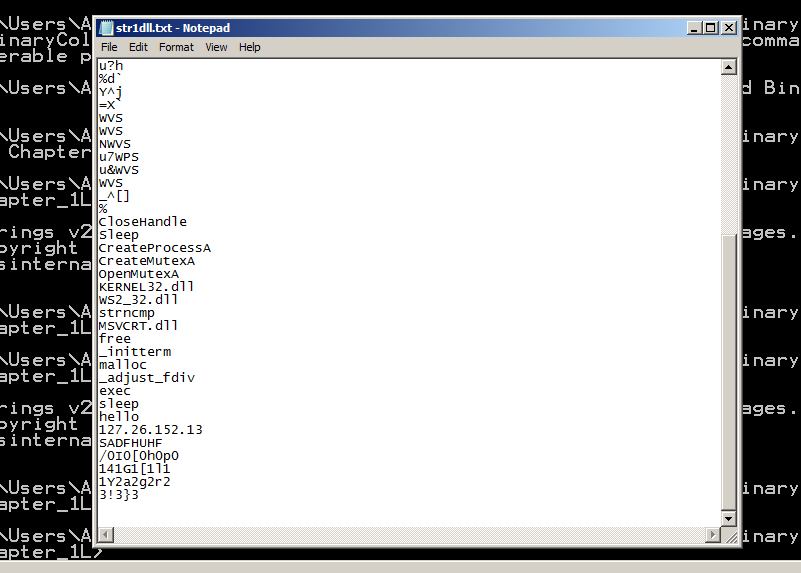
strings Lab01-01.exe > str1exe.txt

notepad str1exe.txt



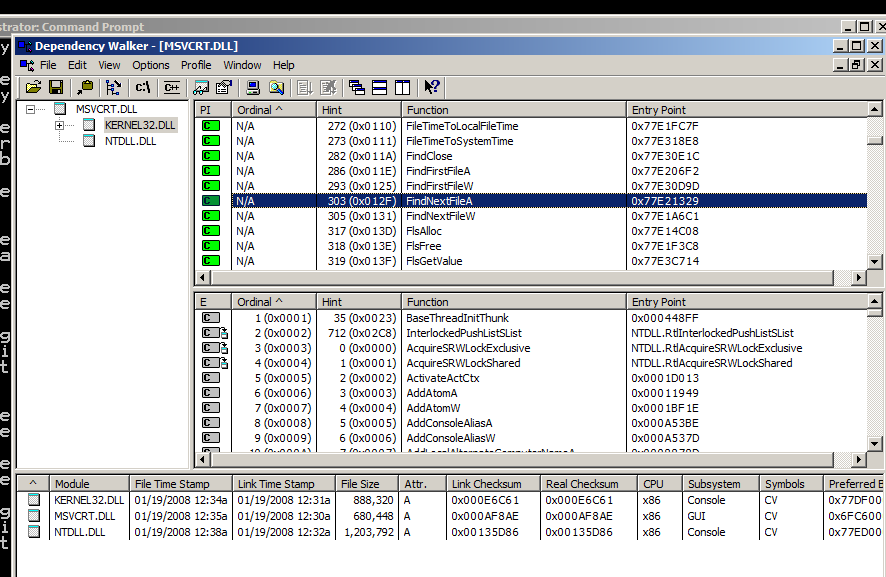
strings Lab01-01.dll > str1dll.txt

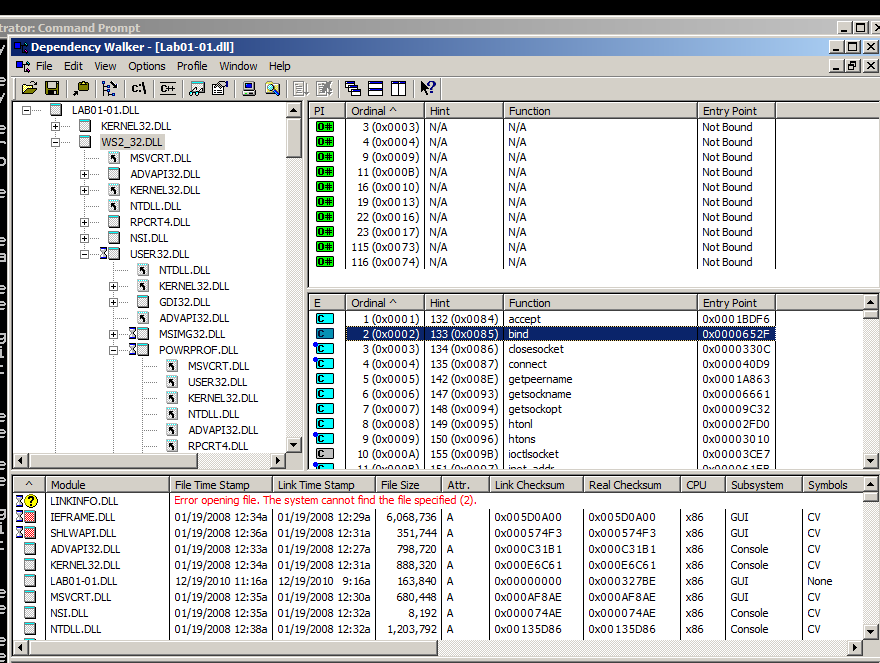
notepad str1dll.txt

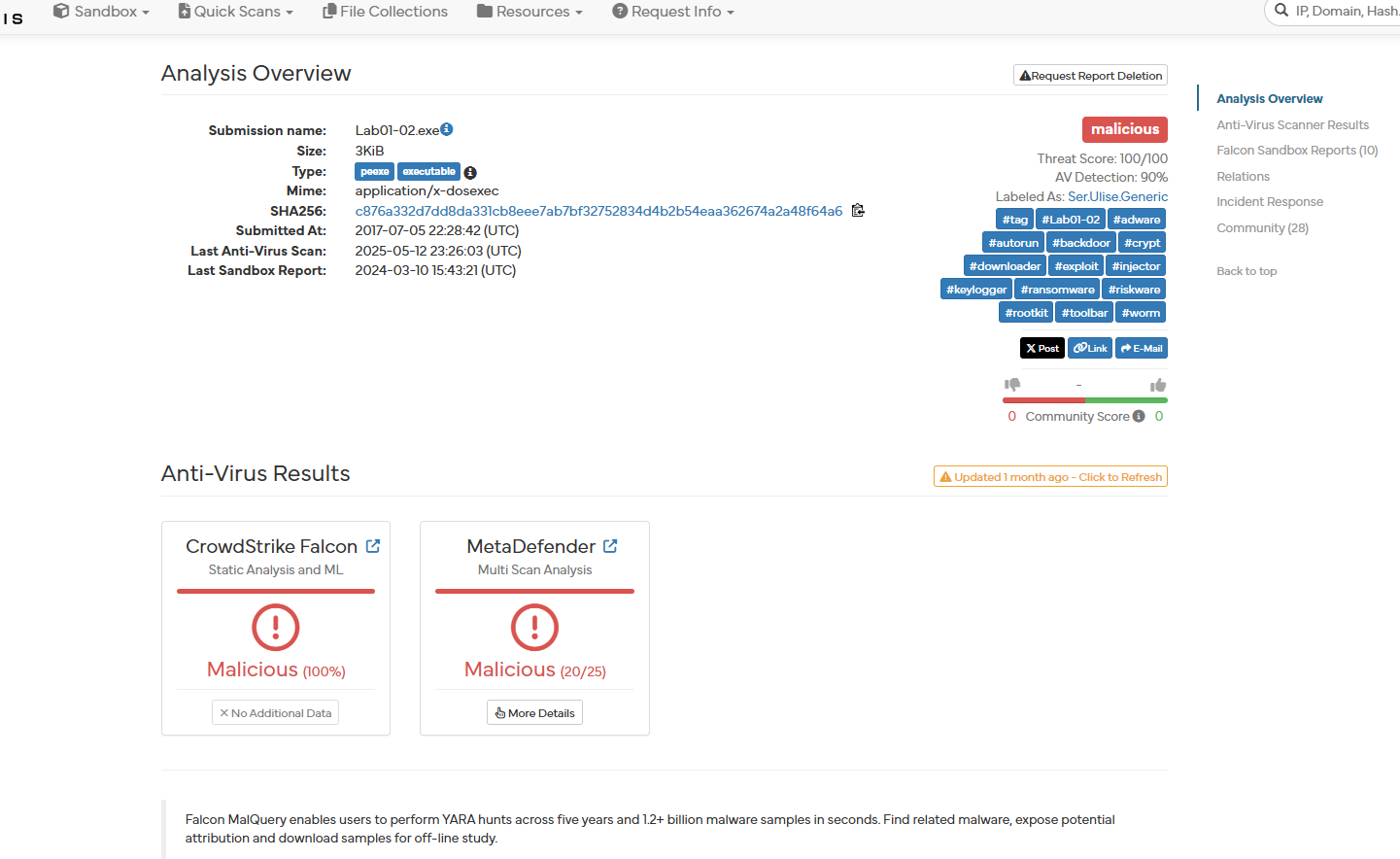


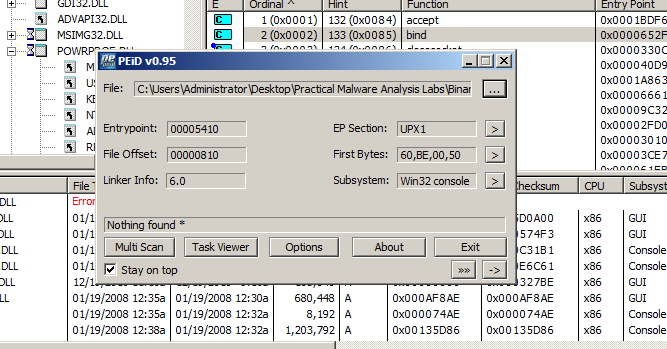
Using Dependency Walker

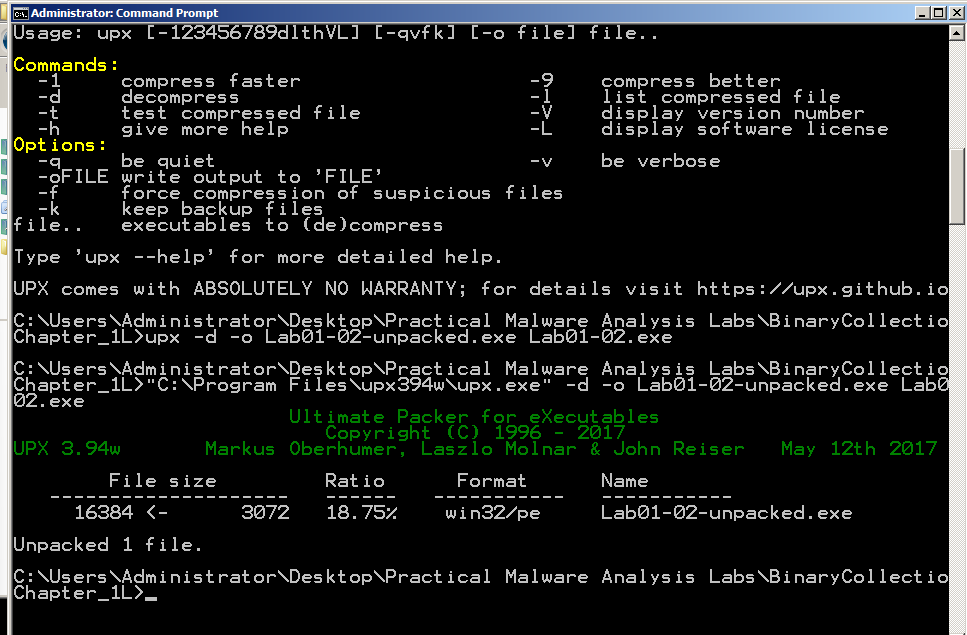
Turn in the image showing your analysis of Lab01-01.exe as shown below. In the "PI^" section (Parent Import), you should see FindNextFileA and FindFirstFileA as shown below.



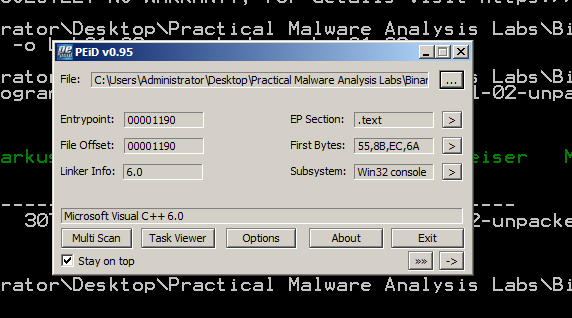
Open Lab01-01.dll in Dependency Walker. Notice that it imports functions from "WS2\_32.DLL". WS2\_32.DLL has networking functions. The right center pane shows function names that perform networking tasks, such as "bind", "closesocket", and "connect", as shown below. 

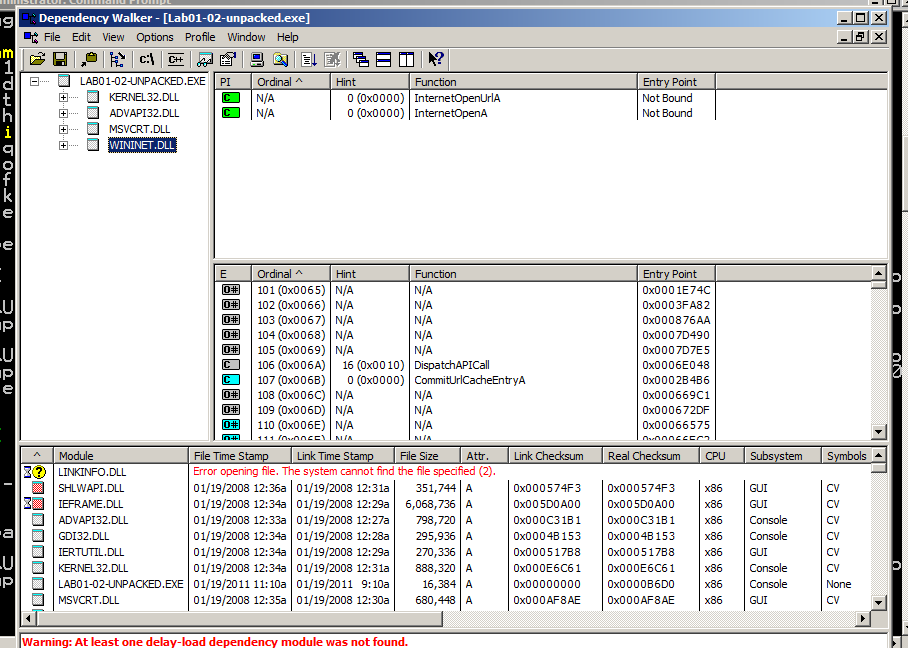
**b. Analyzing Lab01-02.exe**  Unpacking the File

Run PEiD on the file. It shows that the file is packed with UPX, as shown in the "EP Section" below. 

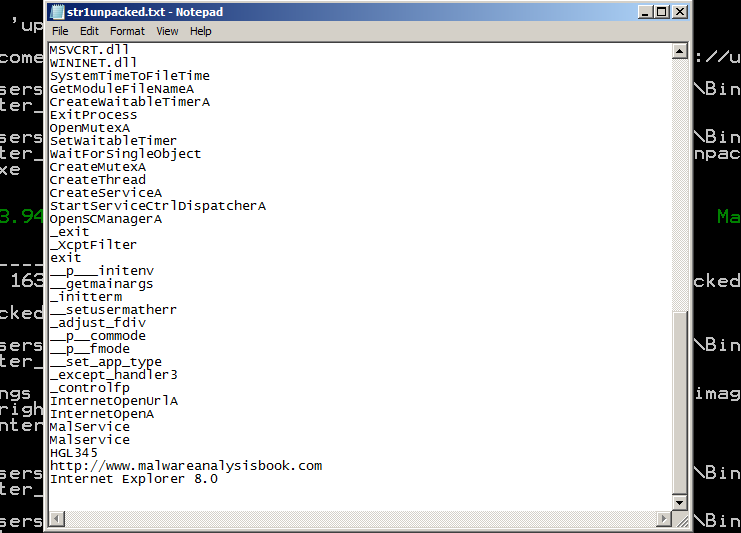
Execute this command to unpack the file: UPX -d -o Lab01-02-unpacked.exe Lab01-02.exe 

Analyze the unpacked file with PEiD. It now is regognized as a "Microsoft Visual C++ 6.0" file, as shown below.

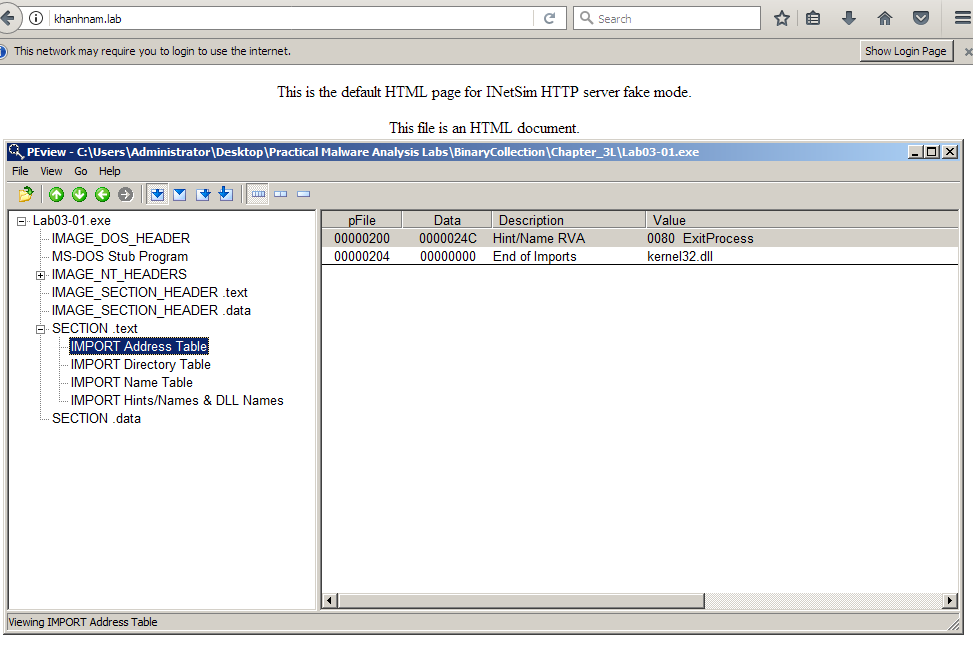


Turn in the image showing the two functions InternetOpenUrlA and InternetOpenA as shown in the upper right pane of the image below

Using Strings

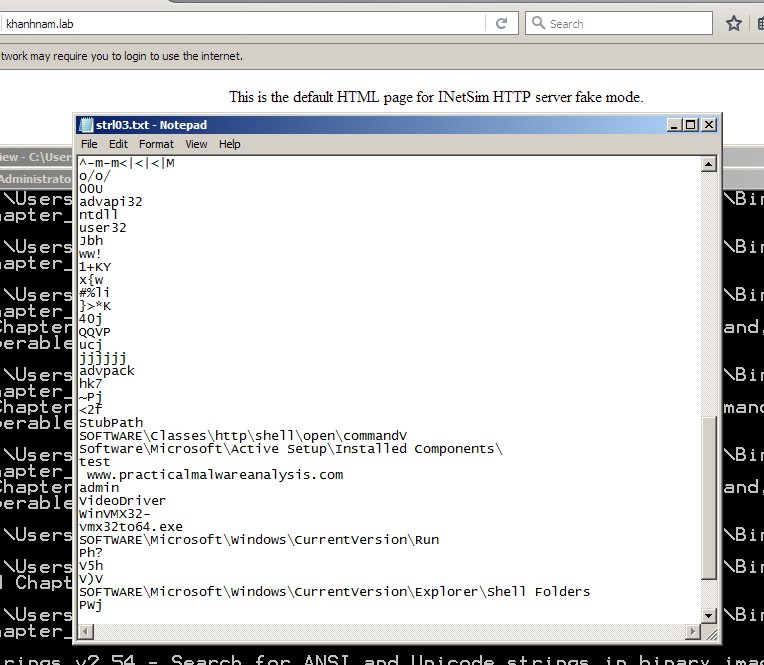
Strings Find the strings in the unpacked file. You should see MalService and http://www.malwareanalysisbook.com as shown below. These suggest that infected machines will connect to http://www.malwareanalysisbook.com and will show a running service named MalService. 

**Part 2: Basic Dynamic Techniques**

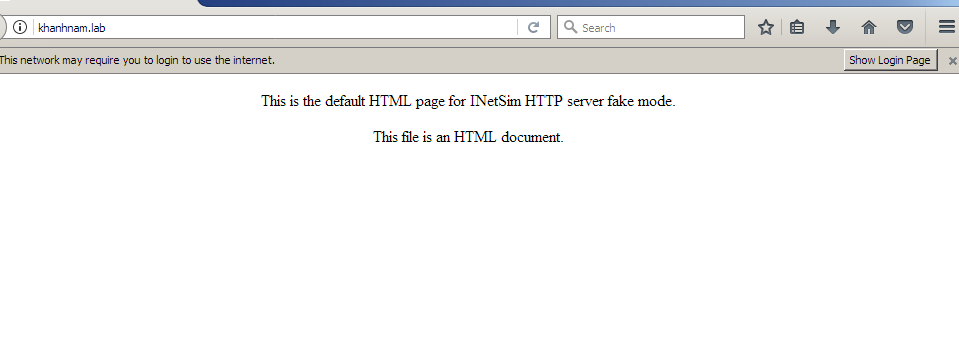
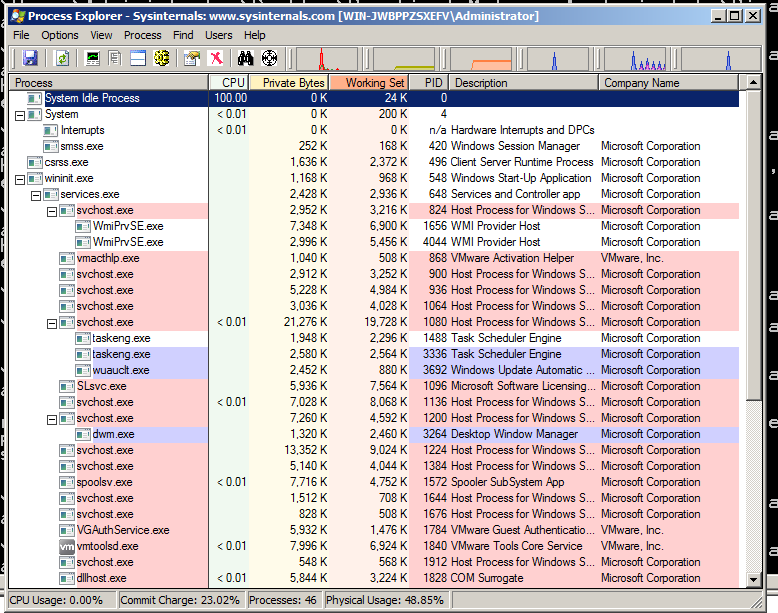
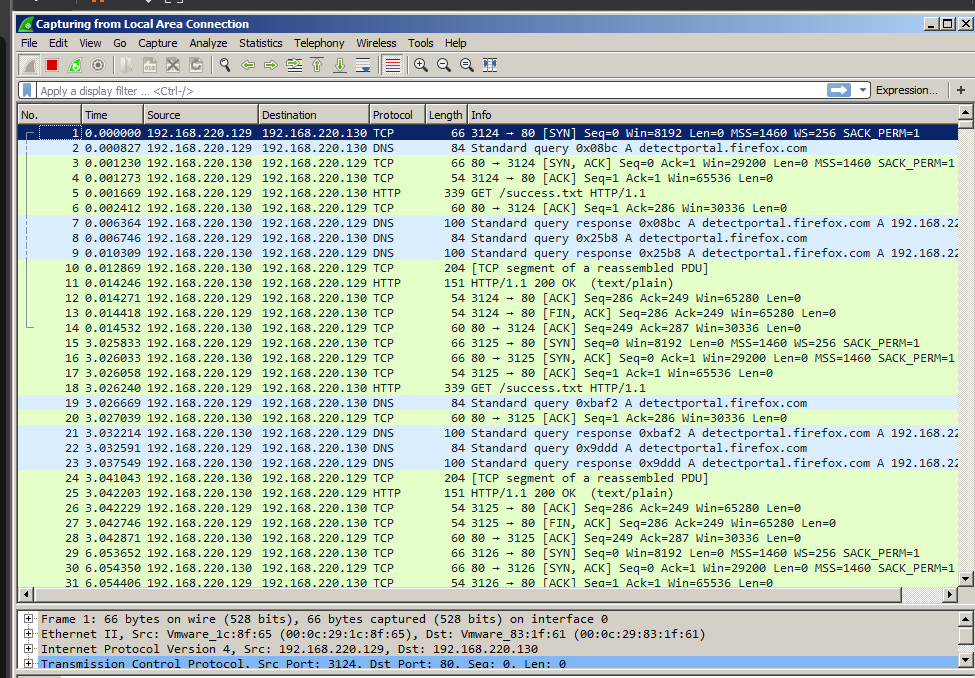
Using PEview Open Lab03-01.exe in PEview. As shown below, the only DLL imported is kernel32.dll, and the only function imported is ExitProcess. That doesn't tell us much--perhaps this malware is packed and the real imports will come at runtime. Turn in the image showing the imports of Lab03-01.exe as shown below. We will grade it by checking the Data value. 

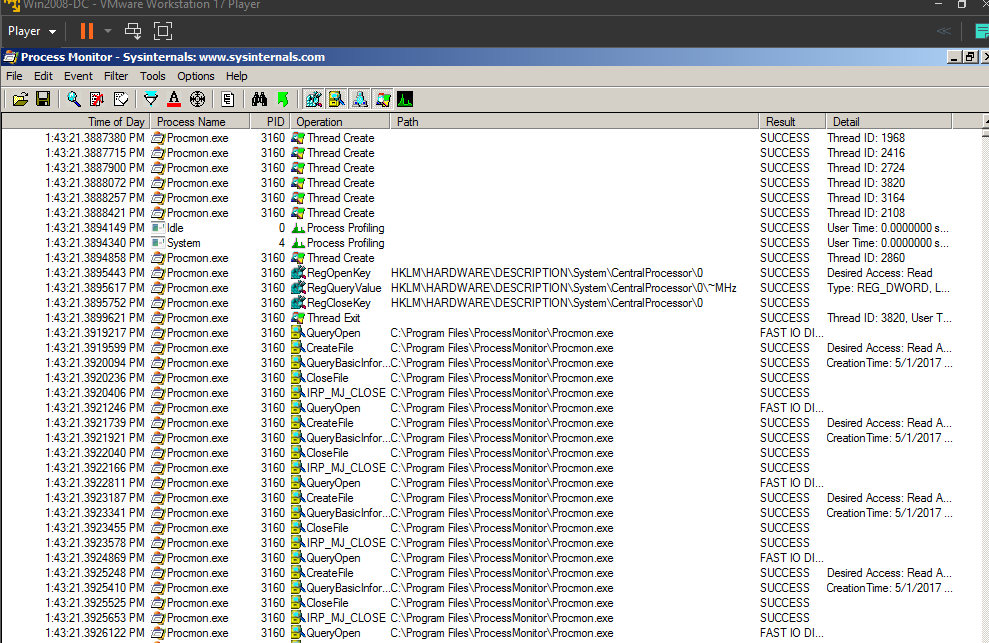
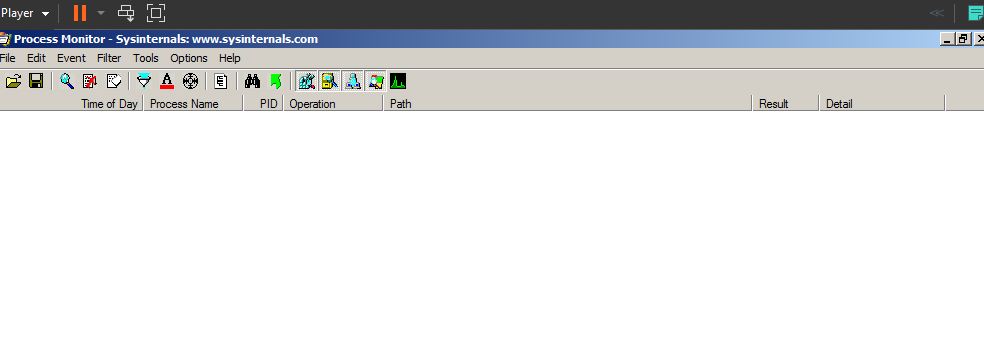
Using Strings

Examine the strings in Lab03-01.exe and find these items, as shown below. SOFTWARE\Classes\http\shell\open\commandV -- A registry location www.practicalmalwareanalysis.com -- a URL VideoDriver

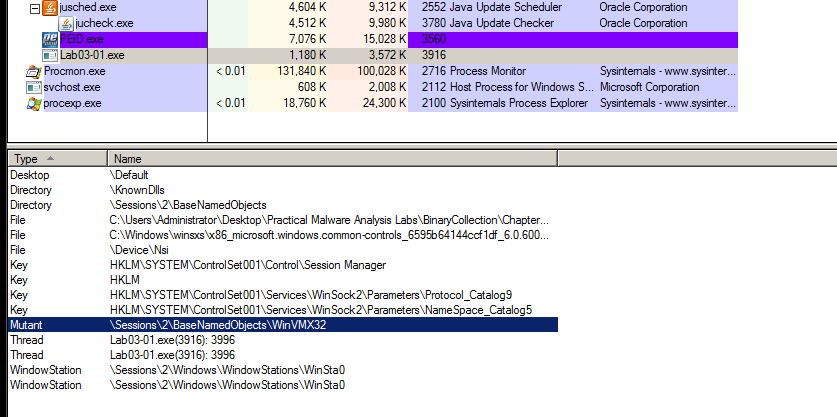
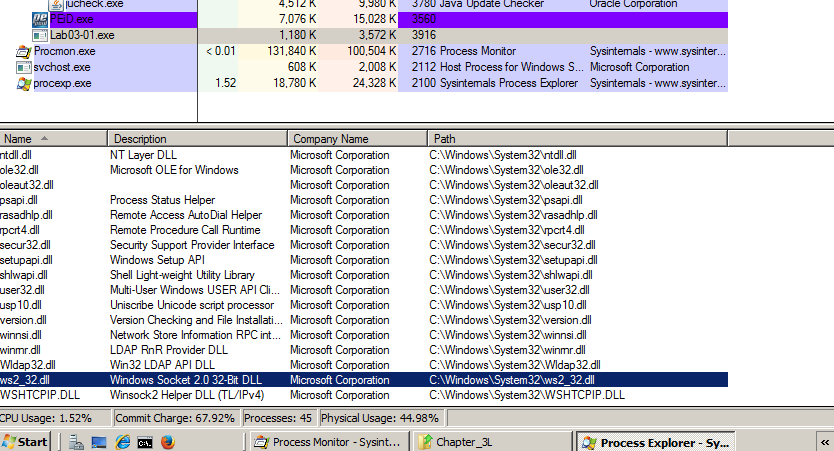
These readable strings are surprising--if the malware were packed, the strings would not be readable. 

Preparing for Dynamic Analysis

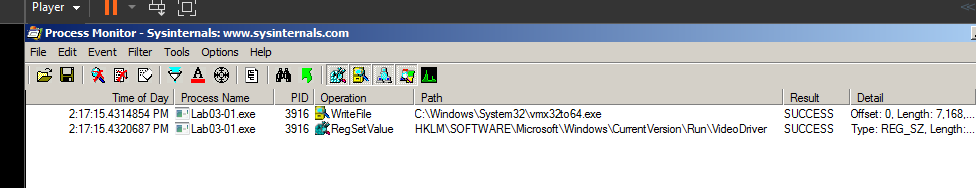
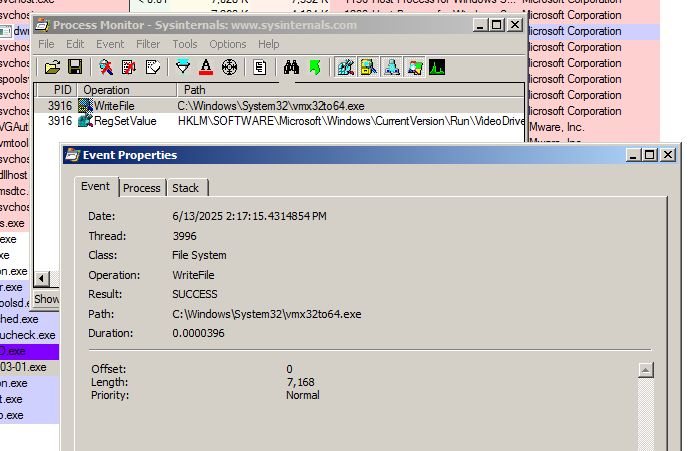
Configuring for Inetsim environment  Run Process Explorer  Run Wireshark 

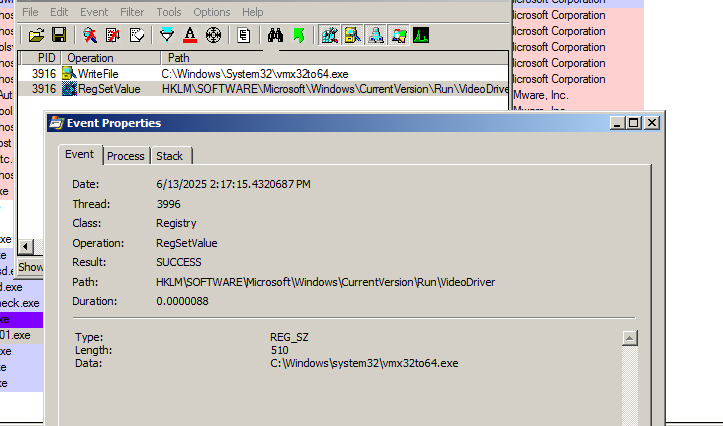
Start Process Monitor  Excluding Harmless Processes  Run the Lab03-01.exe File

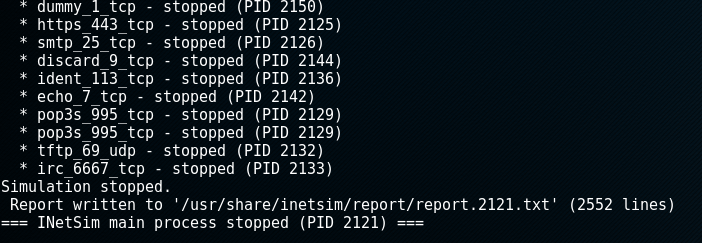
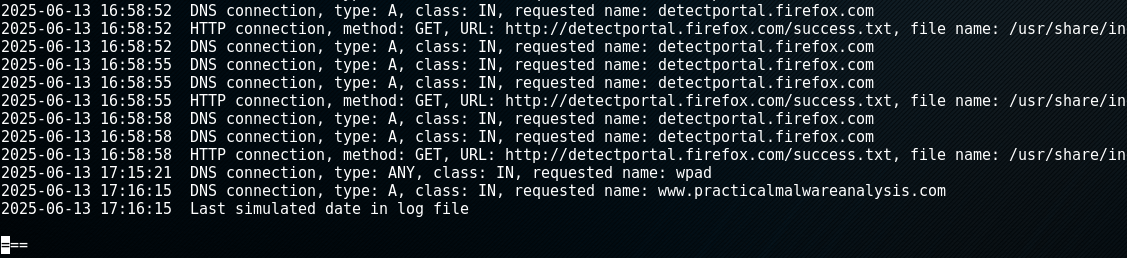
Viewing the Running Malware in Process Explorer

In Process Explorer, click View, "Lower Pane View", Handles. You see the WinVMX32 mutant, as highlighted below. A mutant, also called a mutex, is used for interprocess connunication. In Process Explorer, click View, "Lower Pane View", DLLs. Scroll to the bottom to find ws2\_32.dll and WSHTCPIP.DLL, as shown below. This shows that the malware has networking functionality.

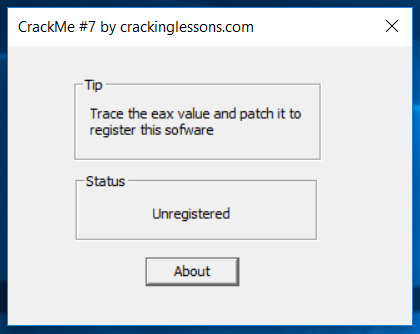
Viewing the Malicious Process's Events in Process Monitor

You end up the two events shown below. Double-click the event with a Path ending in vmx32to64.exe. The Properties sheet shows that this event creates a file named vmx32to64.exe, as shown below. As explained in more detail in the book, this event has copied the malware itself to a file named vmx32to64.exe, so that filename is a useful indicator of infection. 

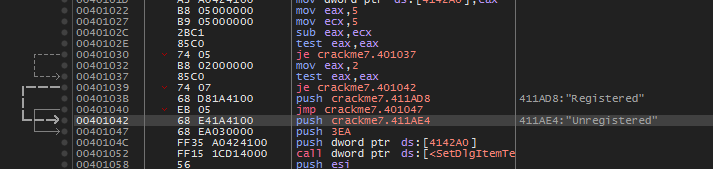
Double-click the with a Path ending in VideoDriver. This creates a new a Run key in the registry named "VideoDriver" with a value of "C:\WINDOWS\system32\vmx32to64.exe" -- this is a persistence mechanism, to relaunch the malware when the machine restarts. 

Viewing INetSim Logs You should see DNS connections to www.practicalmalwareanalysis.com, as shown below:

**CRACK ME 7**

This CrackMe teaches a specific method of cracking which is to trace the eax value and patch it. 

Follow the hint, we will notice any EAX value in the scope. But first let trace into the code of string “Unregistered” displaying in Status box.

After string referencing and trace above the code, we notice this code partition below: 

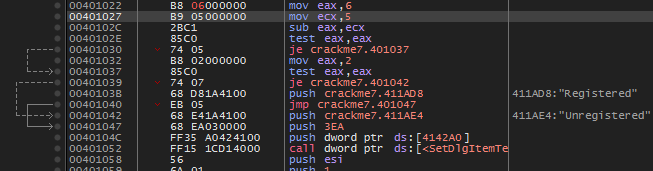
From “mov EAX, 5”, EAX is assigned value 5, happening too with ECX.

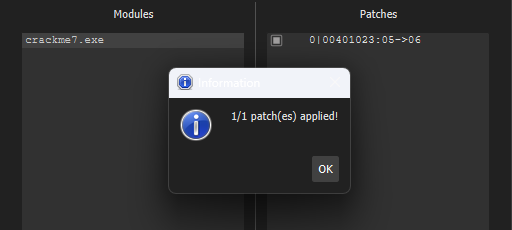
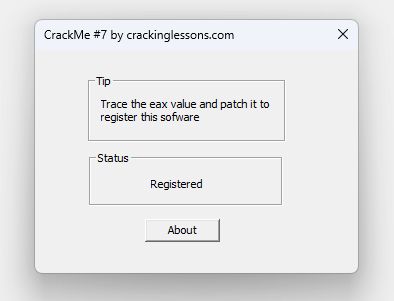
“Sub eax, ecx” created value 0 and assigned to EAX. After subtracting, “test eax, eax” executes “AND bit-wise” operation and put the flag if it’s 0 value – in this condition, EAX is 0 meaning the JE command below is executed. Jumping to another Test command, and another JE command jump to the address of status Unregistered (00401042).

If the first Test command is not operated, ignoring JE command and we have new value of EAX is 2, leading to failure of the second Test and second JE. This make status having “Registed” flag in adress 0040103B.

Hence, I try to change the first value of EAX in address 00401022, leading to value not equals 0 of EAX after subtraction. The idea is prevent the execution of the first Test command.

Mov eax, 5 🡪 Mov eax, 6



Patch file and run again. 

DONE!