Assignment 1

Justin Kennelly

University Of Arizona

CYBV 454 MALWARE THREATS & ANALYSIS

Professor Galde

February 7, 2022

LAB 1-1

* Lab01-01exe : MD5 Checksum Value: bb7425b82141a1c0f7d60e5106676bb1
* Lab01-01dll : MD5 Checksum Value: 290934c61de9176ad682ffdd65f0a669

LAB 1-2

* Lab01-02exe : MD5 Checksum Value: 8363436878404da0ae3e46991e355b83

LAB 1-3

* Lab01-03exe : MD5 Checksum Value: 9c5c27494c28ed0b14853b346b113145

LAB 1-4

* Lab01-04exe : MD5 Checksum Value: 625ac05fd47adc3c63700c3b30de79ab

# LAB 1-1

# Files Examined: Lab01-01.exe and Lab01-01.dll

## LAB 1-1 Question 1

Upload the files to http://www.VirusTotal.com/ and view the reports. Does either file match any existing antivirus signatures?

Graphical user interface

Description automatically generated

*Figure 1: Virus Total Findings for Lab01-01.exe*

Graphical user interface

Description automatically generated

*Figure 2: Virus Total Findings for Lab01-01.dll*

As you can see in Figure 1, Lab01-01.exe matches antivirus signatures on 49/69 of the scanned security vendors. Many of these vendors flagged this file as a Trojan or Malware with high confidence. After running Lab01-01.dll through Virus Total, as seen in Figure 2, we see that this file has matched antivirus signatures on 40/67 security vendors. Like Lab01-01.exe many of these vendors also flagged this file as a Trojan or malware with high confidence.

## LAB 1-1 Question 2

When were these files compiled?

Graphical user interface, text, application

Description automatically generated

*Figure 3: PEview data for Lab01-01.exe*

Graphical user interface, text, application

Description automatically generated

*Figure 4: PEview data for Lab01-01.dll*

I opened PEview and opened Lab01-01.exe. I used PEview because the file headers in the PE file format hold valuable metadata of the file. Navigating to the IMAGE\_FILE\_HEADER shows us various data about the file including a time and date stamp when this particular file was compiled. As shown in Figure 3, we can see that Lab01-01.exe was compiled on Sunday December 12, 2010, at 16:16:19 UTC time. Following the same process as with Lab01-01.exe, I opened Lab01-01.dll in PEview to look at the metadata within its’ file headers. Figure 4 shows us that this file was compiled on Sunday December 12,2010, at 16:16:38 UTC time.

## LAB 1-1 Question 3

Are there any indications that either of these files is packed or obfuscated? If so, what are these indicators?

Graphical user interface, text, application

Description automatically generated

*Figure 5: PEiD results for Lab01-01.exe*

*Graphical user interface, text, application, email

Description automatically generated*

*Figure 6: PEiD results for Lab01-01.dll*

PEiD is a tool that can be used to detect packed or obfuscated files. I opened Lab01-01.exe in PEiD (Figure 5) and could see that this program was compiled using Microsoft Visual C++ 6.0 this indicates to me that this program has not been packed. Going through similar steps with Lab01-01.dll seen in Figure 6, I saw the same results showing the file was compiled using Microsoft Visual C++ 6.0, this file was not packed either. Also referring back to Figures 3 and 4, we can see many imports and a lot of metadata with PEview about these files which also indicates to me that these files are neither packed nor obfuscated.

## LAB 1-1 Question 4

Do any imports hint at what this malware does? If so, which imports are they?

Graphical user interface, application, table

Description automatically generated

*Figure 7: PEview Import data of Lab01-01.exe*

*Graphical user interface, text, application

Description automatically generated*

*Figure 8: PEview Import data of Lab01-01.dll*

Table

Description automatically generated

*Figure 9: Dependency Walker Data for Lab01-01.dll*

Using PEview once again I was able to view the IMPORT Name Tables for Lab01-01.exe (Figure 7). Here I saw two Dynamic Linked Libraries: KERNEL32.dll and MSCVRT.dll. I know that KERNEL32.dll is a commonly used library that offers core functionality like accessing or manipulating memory, files, and hardware. We notice file manipulation occurring because of the functions imported like CopyFile, FindFirstFile, FindNextFile, CreateFile, etc. Next taking a look at MSVCRT.dll. This DLL is used as a C standard library for Visual C++ from ver. 4.2 to 6.0. This does not strike me as odd because in Figure 5, we already knew that this program was compiled using Microsoft Visual C++ 6.0. I took a deeper look at the functions and could not find anything that indicated foul play. Taking a look at Lab01-01.dll I used the same process of opening the file in PEview to look at its; imports (Figure 8). This file has three DLL’s being imported: KERNEL32.dll, MSVCRT.dll, and WS2\_32.dll. I know what the first two libraries are used for after analyzing the first file, and after looking into it I found that WS2\_32.dll is a networking library meaning this is could potentially indicate that this file connects to a network or performs network tasks. I went back to look closer at the functions starting with KERNEL32.dll. Sleep is used to force an application to sleep for an extended number of milliseconds, CreateProcess is used to start an external app then use the PID for calls, CreateMutex is used as a signal to run or stop processes in multithreaded programs, OpenMutex allows multiple process to open handles of the same mutex object, and close handle closes an open object handle. Again, I did not see anything out of the ordinary with MSVCRT.dll, so I moved onto WS2\_32.dll. Within PEview in Figure 8 I was unable to see the functions for WS2\_32.dll because they were called by ordinal. I opened Lab01-01.dll in Dependency Walker (Figure 9) for further examination. This library uses the functions: closesocket, connect, htons (translates short integer from host byte to network byte order), inet\_addr (convert IPv4 decimal address into proper structure), recv (read incoming data on sockets), send, shutdown, socket, WSAStartup (First function called to ID required socket versions to implement sockets), and WSACleanup (Terminates sockets). I believe Lab01-01.dll is used to attach to a system by being installed, from there it will connect to the attackers machine using the built in WS2\_32.dll library functions at which point it can use CreateProcess to execute Lab01-01.exe. From there Lab01-01.exe can use its KERNEL32.dll functions to traverse through the systems file structures and steal files or create new files as needed.

## LAB 1-1 Question 5

## Are there any other files or host-based indicators that you could look for on infected systems?

Text

Description automatically generated

*Figure 10: Result after running Strings command on Lab01-01.exe*

By running the strings command on Lab01-01.exe we are able to see identifiable information about the programs functionality. In Figure 10 we see that Lab01-01.dll is used by Lab01-01.exe in some fashion. Lab01-01.dll could be used as an indication of compromise on an infected machine.

## LAB 1-1 Question 6

## What network-based indicators could be used to find this malware on infected machines?

Text

Description automatically generated

*Figure 11: Result after running Strings command on Lab01-01.dll*

At this point I know that Lab01-01.dll has a WS2\_32.dll library that utilizes network connections. I opened CMD and ran the strings command on Lab01-01.dll to see if there is any useful information (Figure 11). I noticed that the command returned an interesting result of an IP address 127.26.152.13 (Figure 11). This tells me this IP could be the address of the command server used to control the malware.

## LAB 1-1 Question 7

What would you guess is the purpose of these files?

Concluding from the analysis of each file I believe that the purpose of these files is to gain access to a host machine and allow the attacker to access files for infiltration. The attacker can connect to the host machine using Lab01-01.dll and also communicate back to their C2 server implementing Lab01-01.exe to carry out the traversal of file systems.

# LAB 1-2

# Files Examined: Lab01-02.exe

## LAB 1-2 Question 1

## Upload the Lab01-02.exe file to http://www.VirusTotal.com/. Does it match any existing antivirus definitions?

Graphical user interface

Description automatically generated

*Figure 12: Virus Total Findings for Lab01-02.exe*

Running Lab01-02.exe through virus total shows that this file matches antivirus signatures on 55/70 security vendors. There is a very high probability this is malware.

## LAB 1-2 Question 2

## Are there any indications that this file is packed or obfuscated? If so, what are these indicators? If the file is packed, unpack it if possible.

Graphical user interface, text, application, email

Description automatically generated

*Figure 13: PEiD results for Lab01-02.exe*

*Text

Description automatically generated*

*Figure 14: Unpacking Lab01-02.exe using the UPX command in CMD*

*Graphical user interface, text, application

Description automatically generated*

*Figure 15: PEiD results for Lab01-02.exe after unpacking*

I opened Lab01-02.exe in PEiD and determined that the file is packed using UPX (Figure 13). After I determined the algorithm used to pack the file, I was able to determine how to unpack it. I opened the command prompt in windows and navigated to the directory where Lab01-02.exe is located, then I ran the command < upx -d Lab01-02.exe > to unpack it (Figure 14). Lastly, I opened the unpacked file in PEiD to verify that it has been unpacked (Figure 15).

## LAB 1-2 Question 3

## Do any imports hint at this program’s functionality? If so, which imports are they and what do they tell you?

Table

Description automatically generated

*Figure 16: Import data returned by PEview for Lab01-02.exe*

Running Lab01-02.exe in PEview we can see that there are four libraries imported by this program (Figure 16): ADVAPI32.dll, KERNEL32.dll, MSVCRT.dll, and WININET.dll. ADVAPI32.dll tells us this program can access core Windows components like the Service Manager and Registry. Within ADVAPI32 the CreateService function could be used to make a service that can be loaded at boot time. In some malware this function is an indicator of persistence, stealth, or loading kernel drivers. KERNEL32.dll uses GetModuleFileName which can be used by malware to modify files in a running process. OpenMutex is used to ensure that single instances of malware can be run at any given time and CreateMutex offers similar functionality. WININET.dll offers high level networking functionality and can be used to implement FTP, HTTP, NTP protocols. The function in this library InternetOpenUrl opens a URL to make a connection to a protocol. InternetOpen is what initializes the InternetOpenUrl. These imports indicate that this malware may create a service on the host machine that makes a connection to a malicious URL at boot time.

## LAB 1-2 Question 4

What host- or network-based indicators could be used to identify this malware on infected machines?

Text

Description automatically generated

*Figure 17: Result after running Strings command on Lab01-02.exe*

We can identify two network-based indicators to identify this malware by running the strings command on Lab01-02.exe (Figure 17). Once the command is run, we see MalService which could be the service created using the ADVAPI.dll library as well as http://www.malwareanalysisbook.com URL which could be the URL connected to by the WININET library.

# LAB 1-3

# Files Examined: Lab01-03.exe

## LAB 1-3 Question 1

## Upload the Lab01-03.exe file to http://www.VirusTotal.com/. Does it match any existing antivirus definitions?

Graphical user interface

Description automatically generated with medium confidence

*Figure 18: Virus Total Findings for Lab01-03.exe*

Running Lab01-03.exe through virus total we can see that this file matches malicious signatures on 60/68 security vendors. It is safe to say that this is potentially a malware file.

## LAB 1-3 Question 2

## Are there any indications that this file is packed or obfuscated? If so, what are these indicators? If the file is packed, unpack it if possible.

Graphical user interface, text, application

Description automatically generated

*Figure 19: PEiD results for Lab01-03.exe*

*Text

Description automatically generated with medium confidence*

*Figure 20: PEview data from opening Lab01-03.exe*

After running Lab01-03.exe through PEiD I noticed that this file is packed using FSG 1.0 -> dulek/xt (Figure 19). To back up this claim I went ahead and opened the file in PEview and noticed that there are no imports shown which is another indicator that this file is indeed packed. I attempted to search ways to unpack FSG files and stumbled upon ways to unpack using OllyDbg. I attempted this because this software is available in the VLE and at this point I inadvertently ran the malicious file. I then referred to the book and noticed that this type of unpacking is covered, but not until later chapters so I am going to assume that we cannot unpack this file at this time. I have reset my VM due to this mistake and learned a valuable lesson.

## LAB 1-3 Question 3

## Do any imports hint at this program’s functionality? If so, which imports are they and what do they tell you?

Text

Description automatically generated

*Figure 21: Results from running strings command on Lab01-03.exe*

I am unable to view the imports as seen in Figure 20, because Lab01-03.exe is packed using an algorithm I do not yet understand how to unpack. I did run the strings command on Lab01-03.exe (Figure 21) and can see that there is a string called KERNEL32.dll which is a very common DLL, there is a string called LoadLibraryA which is imported by almost all Win32 programs, and another string GetProcAddress which is used to import functions from DLLs or functions in the file header. None of these help me glean information on the functionality of this program.

## LAB 1-3 Question 4

What host- or network-based indicators could be used to identify this malware on infected machines?

Due to the fact that I cannot unpack this file and view the imports I am unable to determine any functionality of this file at this time. Additionally, after running the strings command on Lab01-03.exe (Figure 21) I do not see any network-based indicators.

# LAB 1-4

# Files Examined: Lab01-04.exe

## LAB 1-4 Question 1

Upload the file Lab01-04.exe to http://www.VirusTotal.com/ and view the reports. Does it match any existing antivirus signatures?

Graphical user interface

Description automatically generated with medium confidence

*Figure 22: Virus Total Findings for Lab01-04.exe*

Running Lab01-04.exe through virus total we can see that this file matches malicious signatures on 59/69 security vendors. Due to the high percentage of signatures this file has high potential of being malware.

## LAB 1-4 Question 2

## Are there any indications that this file is packed or obfuscated? If so, what are these indicators? If the file is packed, unpack it if possible.

Graphical user interface, text, application, email

Description automatically generated

*Figure 23: PEiD results for Lab01-04.exe*

*Table

Description automatically generated*

*Figure 24: PEview Data returned from opening Lab01-04.exe*

After running Lab01-04.exe through PEiD (Figure 23). I am able to see that this file was compiled using Microsoft Visual C++ 6.0, this indicates to me that this file is unpacked. To back this up I went ahead and opened the file in PEview and can see a large data set of Imports and functions (Figure 23) which is also an indication that this file is not packed.

## LAB 1-4 Question 3

## When was this program compiled?

Text

Description automatically generated

*Figure 25: IMAGE\_FILE\_HEADER data from PEview*

Looking at the data of Lab01-04.exe in PEview I am able to see information in the IMAGE\_FILE\_HEADER section including the Time Date Stamp (Figure 25). This file was compiled on Friday August 30, 2019, at 22:26:59 UTC time.

## LAB 1-4 Question 4

## Do any imports hint at this program’s functionality? If so, which imports are they and what do they tell you?

Table

Description automatically generated

*Figure 26: Data returned from opening Lab01-04.exe in PEview*

I opened lab01-04.exe in PEview to analyze what imports are used (Figure 26). Lab01-04.exe imports ADVAPI32.dll, KERNEL32.dll, and MSVCRT.dll. Within ADVAPI32.dll the one function that stood out was AdjustTokenPrivileges. This function is implemented by some malware to enable or disable specific access privileges. The KERNEL32.dll import has several functions associated. The ones that stand out are WinExec (Executes a new program) CreateFile/WriteFile/MoveFile (used to manipulate files within the system), CreateRemoteThread (Malware can use this to inject code into a different process), FindResource (Used to find resources in an exe or loaded dll - \*check .rsrc section in PE headers), GetModuleHandle (Can be used to locate/modify code in a loaded module), GetWindowsDirectory (Can be used to determine which directory to install malware), GetTempPath (Could read/write to temporary file path), LoadResource (Loads resource from PE file into memory). MSVCRT.dll is used in the compilation process and none of the functions hold any hints to the functionality of the file. Based on these imports it looks like this program could be used to get privileged access to a machine by finding and modifying its access tokens. From there it could be concluded that the malware may load something malicious from another dll or memory then look for a directory or temporary path to write the file too. I would guess that at that point it executes this file because of the WinExec import.

## LAB 1-4 Question 5

## What host- or network-based indicators could be used to identify this malware on infected machines?

Text

Description automatically generated

*Figure 27: Data returned after running the strings command on Lab01-04.exe*

By running the strings command on Lab01-04.exe I am able to see three strings that stand out to me: \winup.exe, \system32\wupdmgrd.exe, and http://www.practicalmalwareanalysis.com/updater.exe (Figure 27). The first two strings listed look like file system paths which tells me these could be host-based indicators on an infected machine. One could navigate to system32 and look for the executable wupdmgrd.exe. The last string is a URL that can be used as a network-based indicator on an infected machine.

## LAB 1-4 Question 6

This file has one resource in the resource section. Use Resource Hacker to examine that resource, and then use it to extract the resource. What can you learn from the resource?

Text

Description automatically generated

*Figure 28: Opening Lab01-04.exe in Resource Hacker*

Table

Description automatically generated

*Figure 29: Imports found in IDA pro with the Binary extraction from Resource Hacker*

I opened Lab01-04.exe in ResourceHacker and found one resource in a BIN folder (Figure 28). I extracted this file by saving the resource as a .bin file. From there I opened IDA pro and opened my saved binary file. Navigating through the tabs I opened the Imports tab in IDA pro and noticed a new library urlmon and its import called URLDownloadToFileA (Figure 29). I actually noticed this earlier when I ran the strings command on Lab01-04.exe earlier (Figure 27), and it makes sense why I did not see it in the initial investigation of the imports now. This tells me that the program locates this resource using FindResource and loads it during runtime using LoadResource.