# **[Practical Malware Analysis by Sikorski & Hoing](https://github.com/mikesiko/PracticalMalwareAnalysis-Labs)**

## **Lab 1-1**

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

## Lab01-01.exe has a detection ratio of 53/76. Lab-01-01.dll has a detection ratio of 43/76.

## 2. When were these files compiled?

## Using PEview Lab01-01.exe & it's corresponding DLL both show IMAGE\_FILE\_HEADER has a time stamp of 12/19/2010.

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

## Using strings -> 'strings lab01-01.exe' revealed a lot of specific functions and .dll calls. This would indicate the file is unlikely packed. To further confirm the unlikelyhood of the file being packed, running PeID reveals the .exe is compiled using Microsoft Visual Studio C++ 6.0. Using strings -> 'strings lab01-01.dll' also reveals many specific functions and strings, which would also indicate the file is not packed. PeID revealed the .dll to be compiled using Microsoft Visual C++ 6.0.

## 4. Do any imports hint at what this malware does? If so, what imports are they?

## Strings revealed that lab01-01.exe calls on KERNEL32.dll & MSVCRT.dll. Using Dependency Walker on the .exe Shows the kernel32.dll calls notable functions such as "CreateFileA, CopyFileA, FineFirstFileA, and FindNextFileA." MSVCRT.dll calls runtime functions such as malloc, stricmp, set\_app\_type, adjust fdiv whichh indicate both string and integer comparison along with allocating memory. Another call is made to NTDLL.dll which indicates it's attempting to bridge and make calls to windows kernel. This is highly indicative that this file can potentially corrupt the Windows system. Immediately delving into lab01-01.dll with strings notates an IP address '127.26.152.13' It also indicates usage of kernel32.dll, ws2\_32.dll, and msvcrt.dll. There are further strings that indicate the .dll sends a 'hello' parameter-- common in communication with a network. The sleep function is also called. Popping this bad boy into Dependency Walker further reveals Kernel32.dll calls for 'Close Handle, CreateMutexA, CreateProcessA, OpenMutexA, Sleep.' This information leads me to believe the malicious file is creating a new process, threading between other applications to run synchronized, and an ability to close the handle. Sleep indicates at some point in time this program can be called to stop execution for a period of time. ADVAPI32.dll is called with functions that show the application is going to change, create, enumerate, delete, open, and query Registry Keys. WS2\_32.dll being called is creating an interface to connect to an outside target. Given the IP in strings and that there was no packing involved, we can assume this is the only malicious IP being called in immediacy. User32.dll calls for PostMessageA, PeekMessageA, DispatchMessageA, TranslateMessage which given previous information leads me to believe this program is a remote access trojan and capable of destroying the entire operating system.

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

Outside of running this dynamically, static analysis would lead me to believe the best possible solution to finding these would be to check Event Log. Another indication could be random 'errors,' netstat showing unknown connections, and registry probably retains the virus to run at startup.

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

Given my information from 'strings -> lab01-01.dll' any network monitor or netstat would indicate 127.26.152.13 is the malicious connection.

## 7. What would you guess is the purpose of these files?

A hostile takeover from the address 127.26.152.13 as the command & control center. Given all of the functions being called in these two files an entire ability to communicate with the host and tamper with everything from the registry to potentially the kernel.