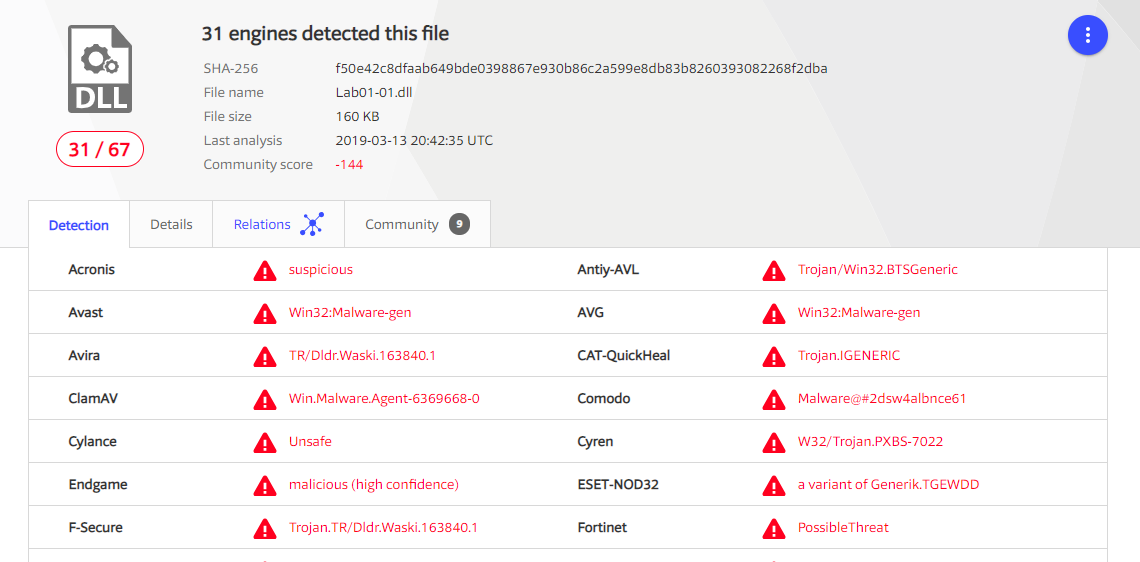
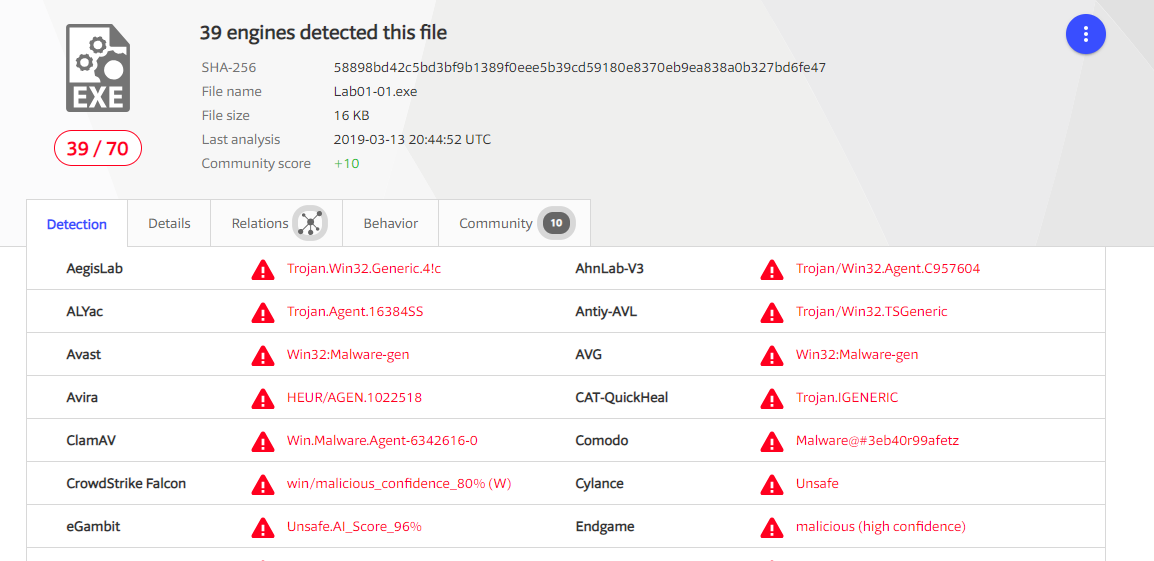
분석보고서1 2016111566 이나경

실습 1-1)

Lab01-01.exe, Lab01-01.dll

◦ Virustotal에 업로드 하고 보고서를 확인하자. 기존 안티바이러스 정의된 것과 일치하는가?

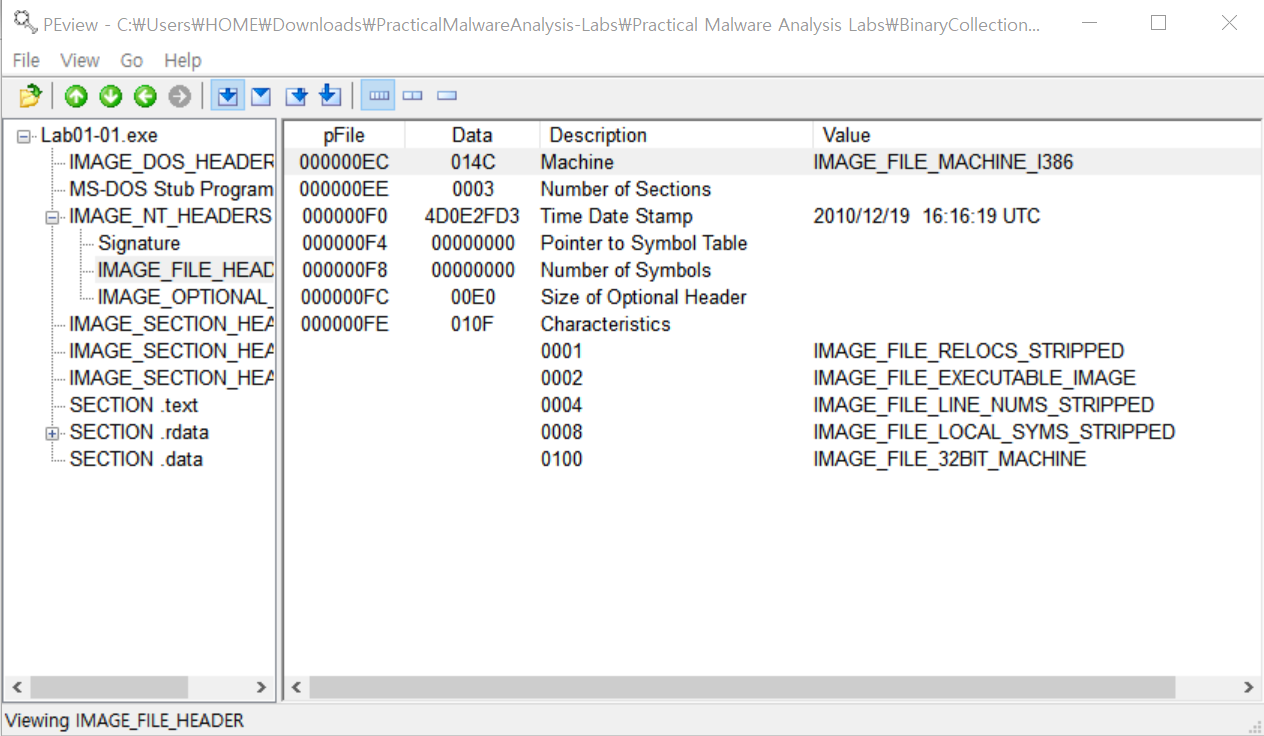




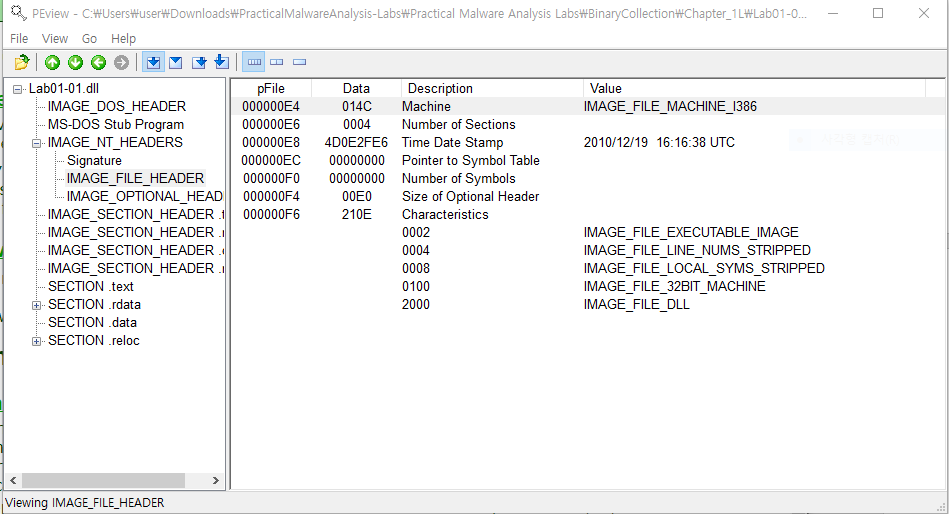
Lab01-01.dll은 67개의 엔진 중에 31개에서 기존의 안티바이러스가 감지되었고

Lab01-01.exe은 70개의 엔진 중에 39개에서 기존의 안티바이러스가 감지되었다.

◦ 이 파일은 컴파일 시점은?



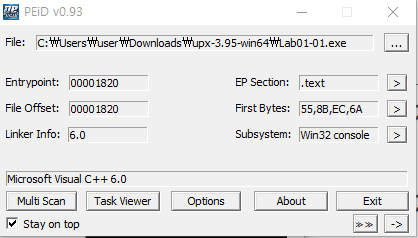
PeView로 확인하면 Lab01-01.exe의 컴파일 시점은 2010년 12월 19일 16시 16분 19초인 것을 알 수 있고

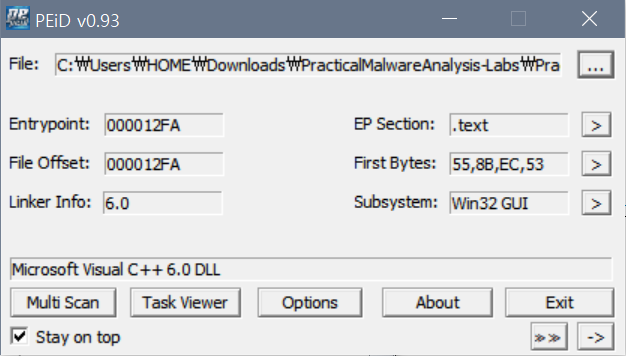


Lab01-01.dll의 컴파일 시점은 2010년 12월 19일 16시 16분 38초인 것을 알 수 있다.

따라서 이 두개 파일은 같은 패키지로 판단이 되며 exe가 실행되고 dll을 동적으로 호출하는 구조로 판단된다.

◦ 패킹이나 난독화의 흔적이 있는가? 이유는?

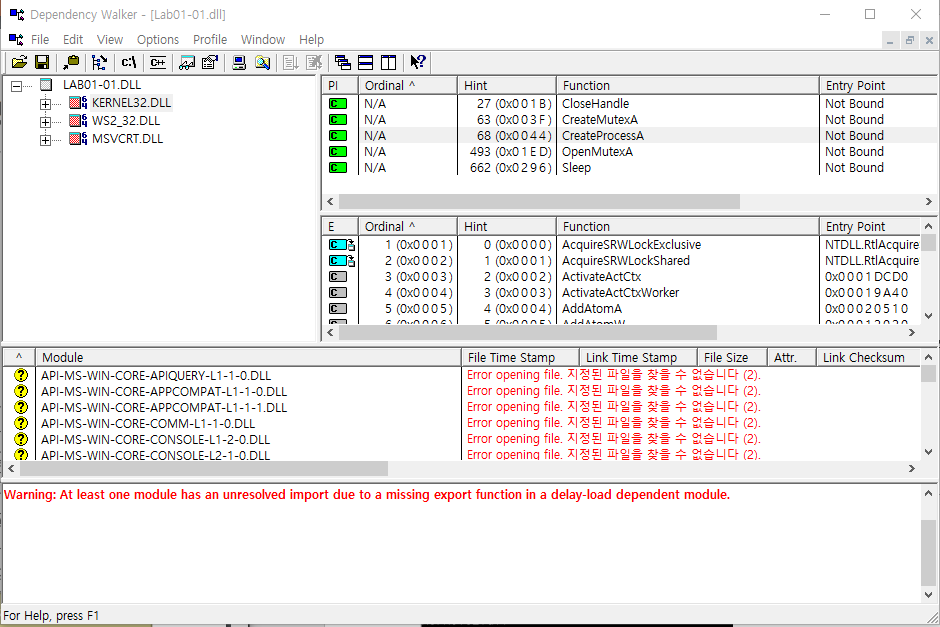




PEiD로 확인한 결과 Lab01-01.exe, Lab01-01.dll 두 파일 모두 Visual C++에서 정상적으로 컴파일한 파일로 패킹된 흔적은 없었다.

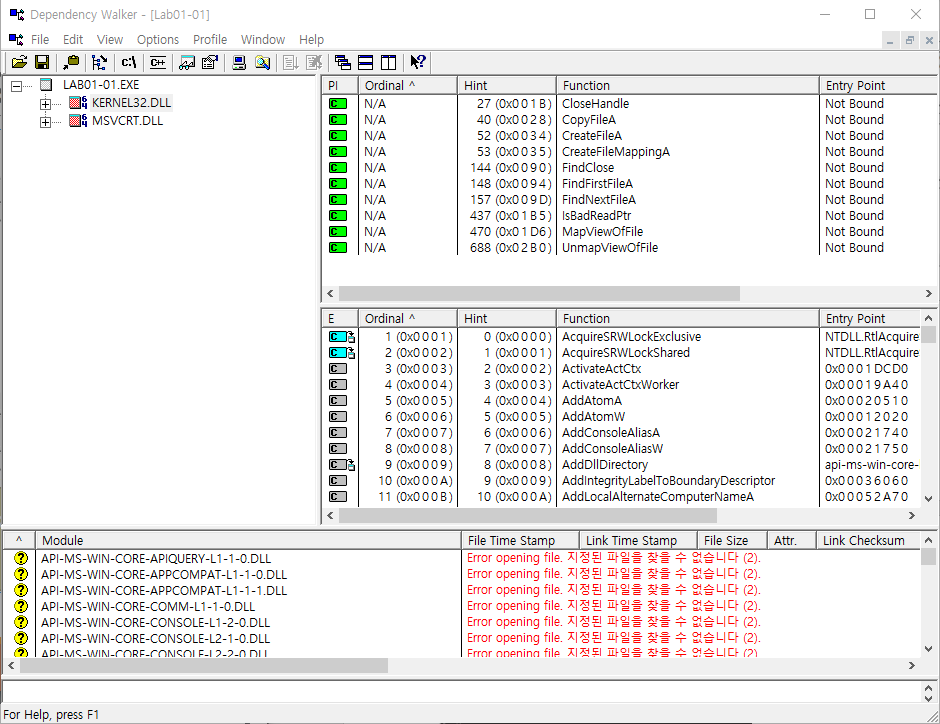
◦ 감염된 시스템에서 검색할 수 있는 다른 파일이나 호스트 기반의 증거가 존재하는가?

임포트를 보고 악성코드 행위 판단

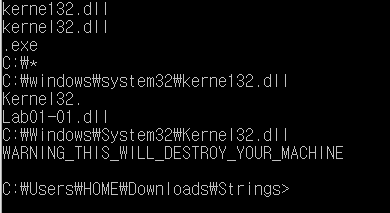


Lab01-01.dll은 WS2\_32.DLL을 임포트하기 때문에 네트워크 연결작업을 수행할 것으로 판단된다.

임포트 함수 부분에서 CreateProcess와 Sleep을 사용하고 있는데 이를 이용해 백도어 프로그램을 실행하고 백그라운드에서 실행을 대기하고 있도록 한다.

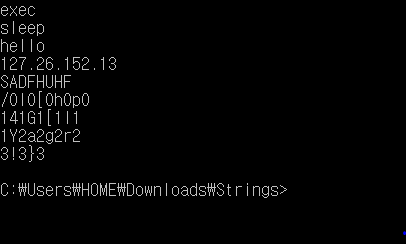


Lab01-01.exe파일은 Kernel32.dll과 Msvcrt.dll을 임포트 하고 있으나 이는 대부분의 파일에서 임포트하는 파일이며 CopyFile과 CreateFile로 파일을 복사하거나 생성하는 기능을 하며 FindFirstFile, FindNextFile로 파일을 탐색하는 행위를 하고있다.



Strings로 Lab01-01.exe을 확인한 결과 kerne132.dll을 kernel32.dll인것처럼 위장하여 임포트하는 것으로 판단되며 시스템에 악성코드 kerne132.dll을 배포하여 실행하도록 할 수 있다. 또한 Lab01-01.dll파일을 실행시킨다.

◦ 감염된 장비에서 이 악성코드를 발견하기 위해 사용한 네트워크 기반의 증거는 무엇인가?



Strings로 Lab01-01.dll을 확인한 결과 ws2\_32.dll과 함께 ex, sleep이 수행되는데 프로그램을 네트워크상으로 수행시키고 백그라운드에서 대기하도록 한다. 또한 127.26.152.13라는 IP를 확인할 수 있는데 해당 IP에 접근하여 수행되는 것으로 판단된다.

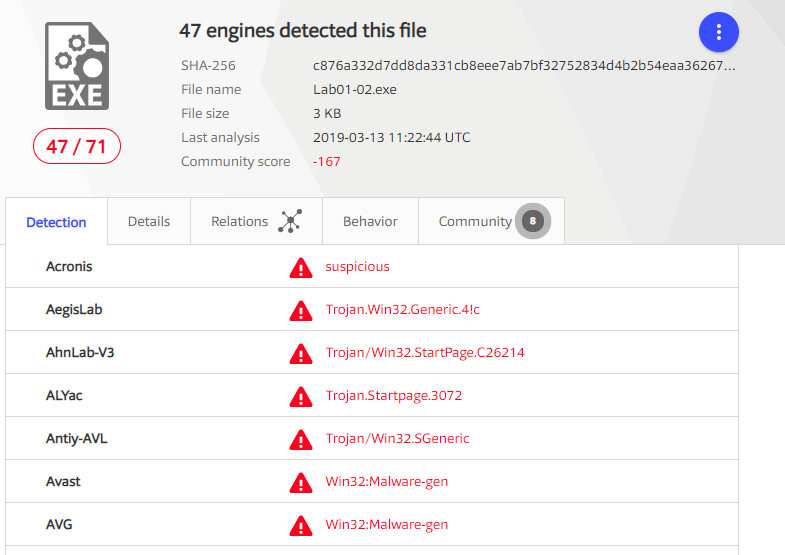
◦ 이 파일의 목적은 무엇이라고 판단하였는가?

두개의 파일이 비슷한 시기에 만들어졌고 exe파일이 파일을 검색하고 생성하고 dll을 실행시키는 것을 보면 dll에서 네트워크 기반의 작동을 하기 때문에 dll이 일종의 백도어로 판단된다.

실습 1-2)

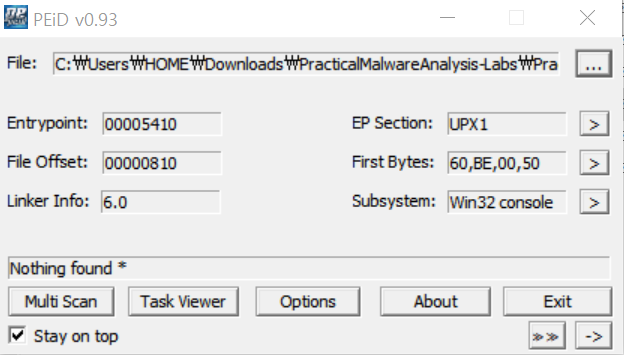
Lab01-02.exe

◦ Virustotal에 업로드 하고 보고서를 확인하자. 기존 안티바이러스 정의된 것과 일치하는가?



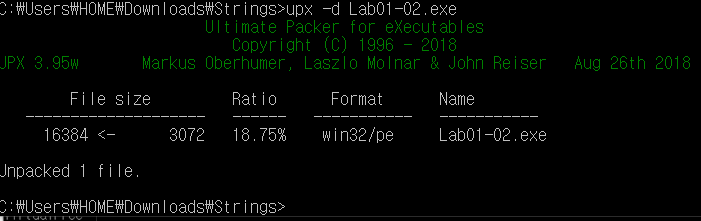
71개의 엔진 중에 47개에서 기존의 안티바이러스가 감지되었다.

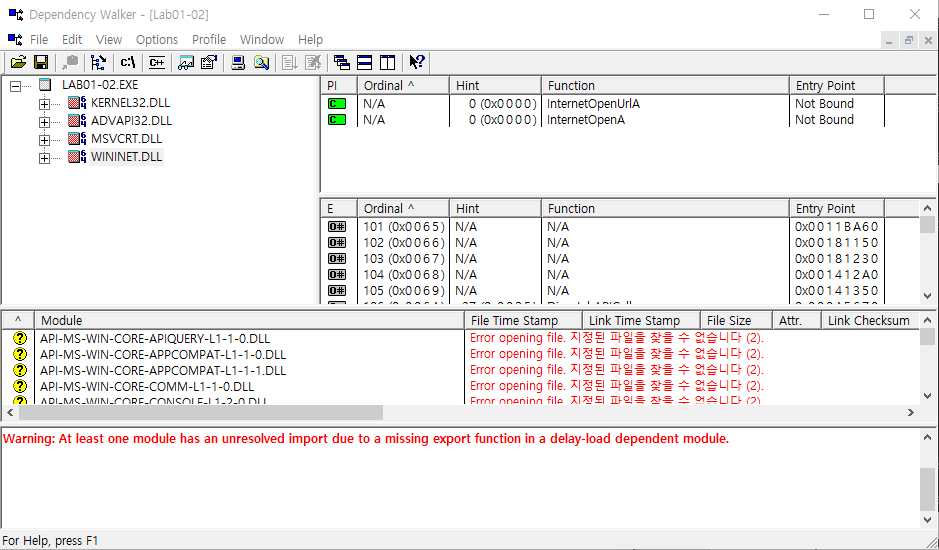
◦ 패킹이나 난독화의 흔적이 있는가? 이유는?

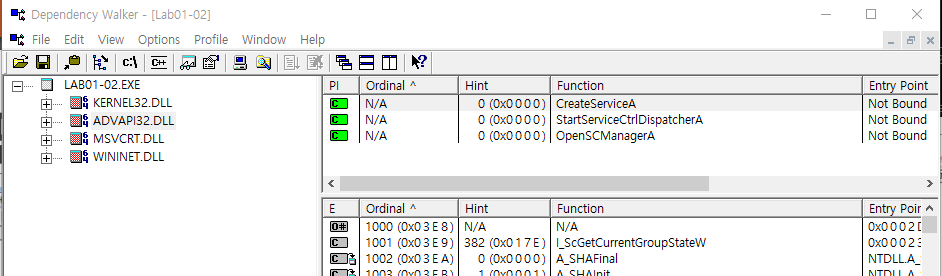


PEiD로 확인한 결과 UPX로 패킹되어있는 것을 알 수 있었다.

◦ 임포트를 보고 악성코드의 기능을 알아 낼 수 있는가? 그렇다면 어떤 임포트를 보고 알 수 있었는가?









Upx로 언패킹한 후 확인한 결과 CreateServiceA, InternetOpenUrlA, InternetOpenA 등의 함수를 확인할 수 있었다.

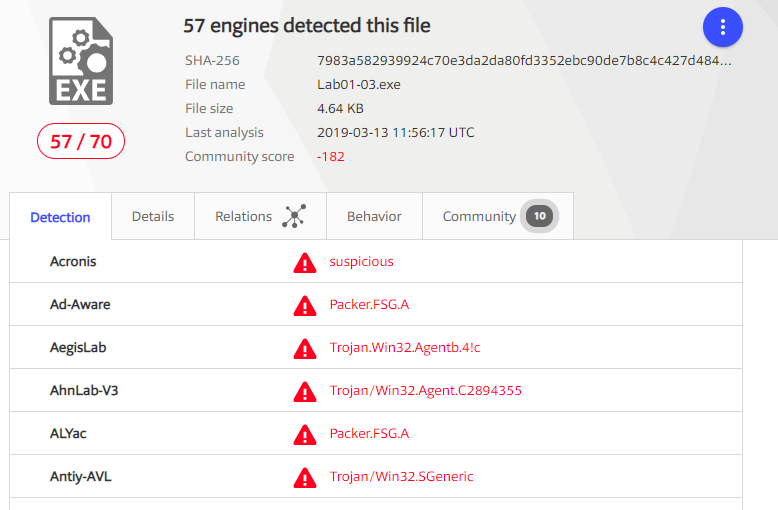
◦ 감염된 장비에서 이 악성코드를 발견하기 위해 사용한 네트워크 기반의 증거는 무엇인가?

위의 사진에서 확인할 수 있듯이 strings로 확인한 결과 <http://www.malwareanalysisbook.com> 라는 url을 확인할 수 있었고 Internet Explorer 8.0등 네트워크를 이용하는 것을 알 수 있다.

실습 1-3)

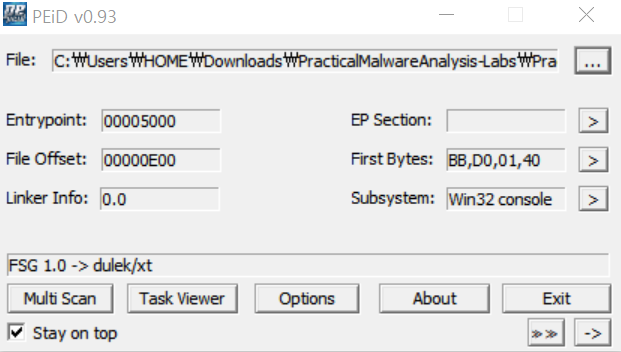
Lab01-03.exe

◦ Virustotal에 업로드 하고 보고서를 확인하자. 기존 안티바이러스 정의된 것과 일치하는가?



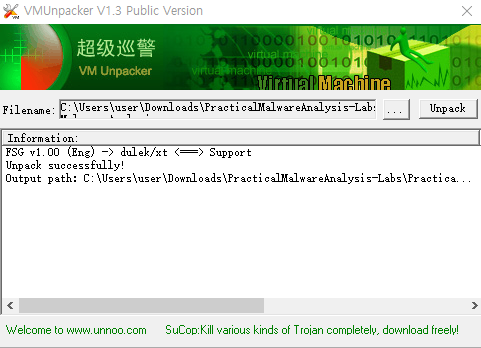
70개의 엔진 중에 57개에서 기존의 안티바이러스가 감지되었다.

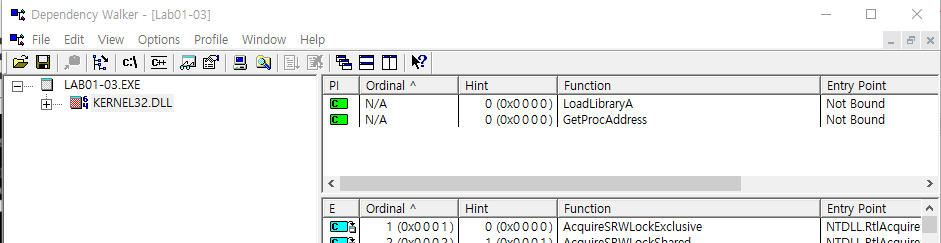
◦ 패킹이나 난독화의 흔적이 있는가? 이유는?

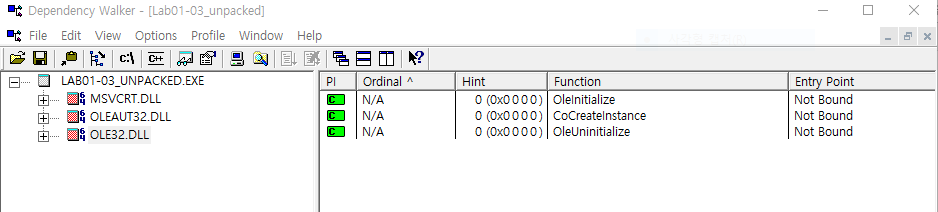


PEiD로 확인한 결과 FSG로 패킹되어있는 것을 알 수 있었다.

◦ 임포트를 보고 악성코드의 기능을 알아 낼 수 있는가? 그렇다면 어떤 임포트를 보고 알 수 있었는가?

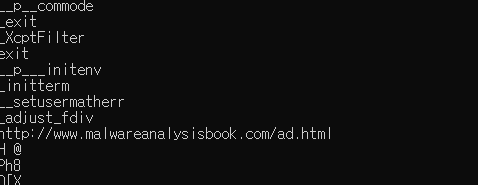






언팩하여 확인한 결과 LoadLibraryA, GetProcAddress, CoCreateInstance 등의 함수를 확인할 수 있었다.

◦ 감염된 장비에서 이 악성코드를 발견하기 위해 사용한 네트워크 기반의 증거는 무엇인가?

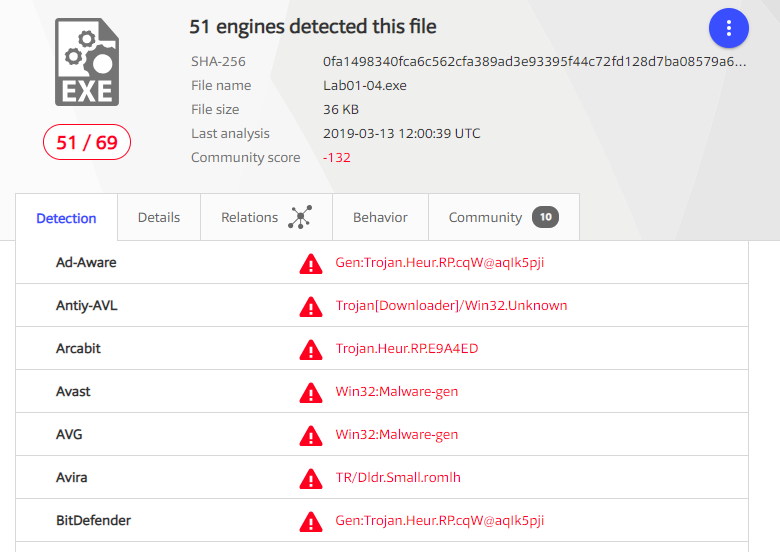


strings로 확인한 결과 <http://www.malwareanalysisbook.com/ad.html> 라는 url을 확인할 수 있었다.

실습 1-4)

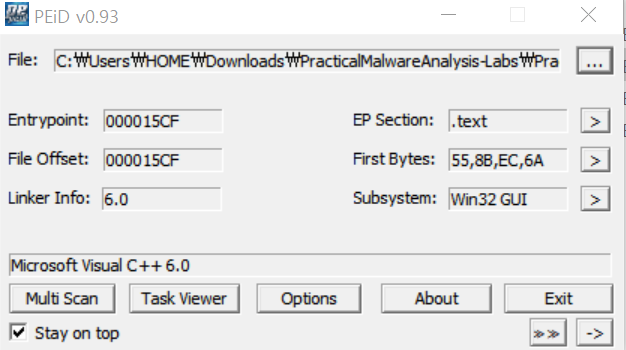
Lab01-04.exe

◦ Virustotal에 업로드 하고 보고서를 확인하자. 기존 안티바이러스 정의된 것과 일치하는가?



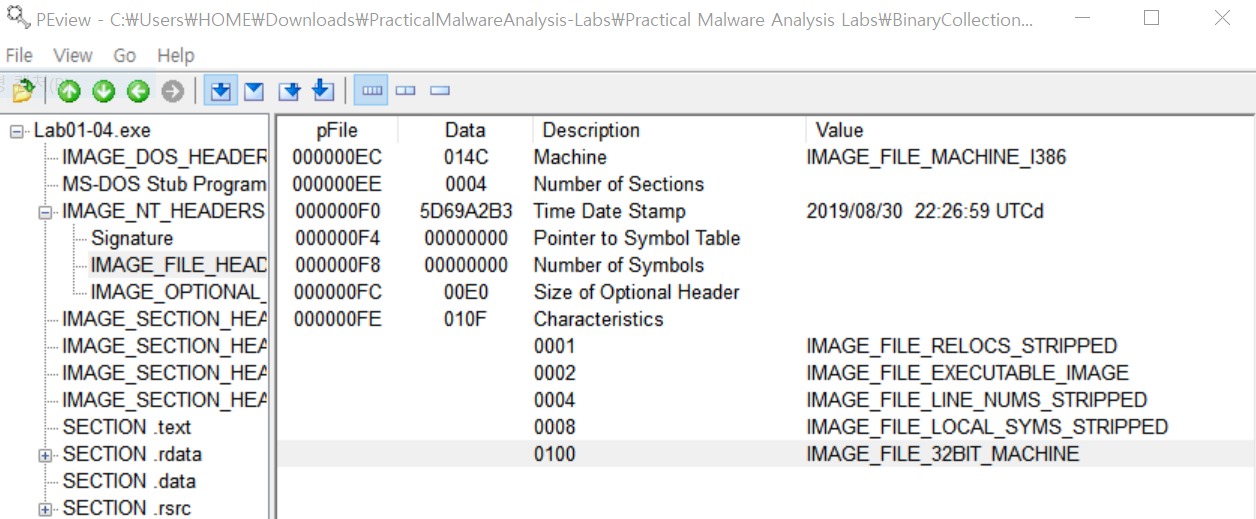
69개의 엔진 중에 51개에서 기존의 안티바이러스가 감지되었다.

◦ 패킹이나 난독화의 흔적이 있는가? 이유는? 파일이 패킹되어 있다면 언패킹하라.



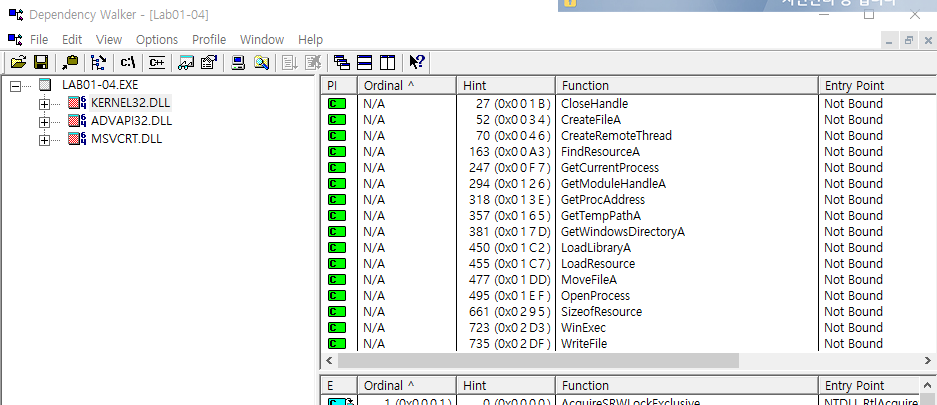
PEiD로 확인한 결과 패킹된 흔적은 없었다.

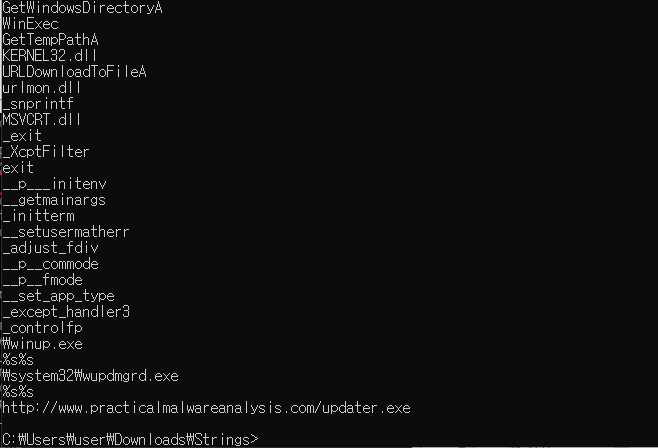
◦ 이 프로그램은 언제 컴파일됐는가?



PeView로 확인한 결과 Lab01-04.exe의 컴파일 시점은 2019년 8월 30일 22시 26분인 것을 알 수 있었다.

◦ 임포트를 보고 악성코드의 기능을 알아 낼 수 있는가? 그렇다면 어떤 임포트를 보고 알 수 있었는가?





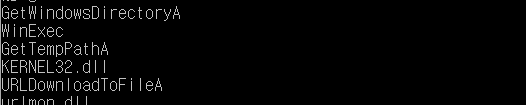
확인한 결과 CreateFileA, WriteFile, FindResourceA, LoadResource, WinExec 등의 함수를 확인할 수 있었다.

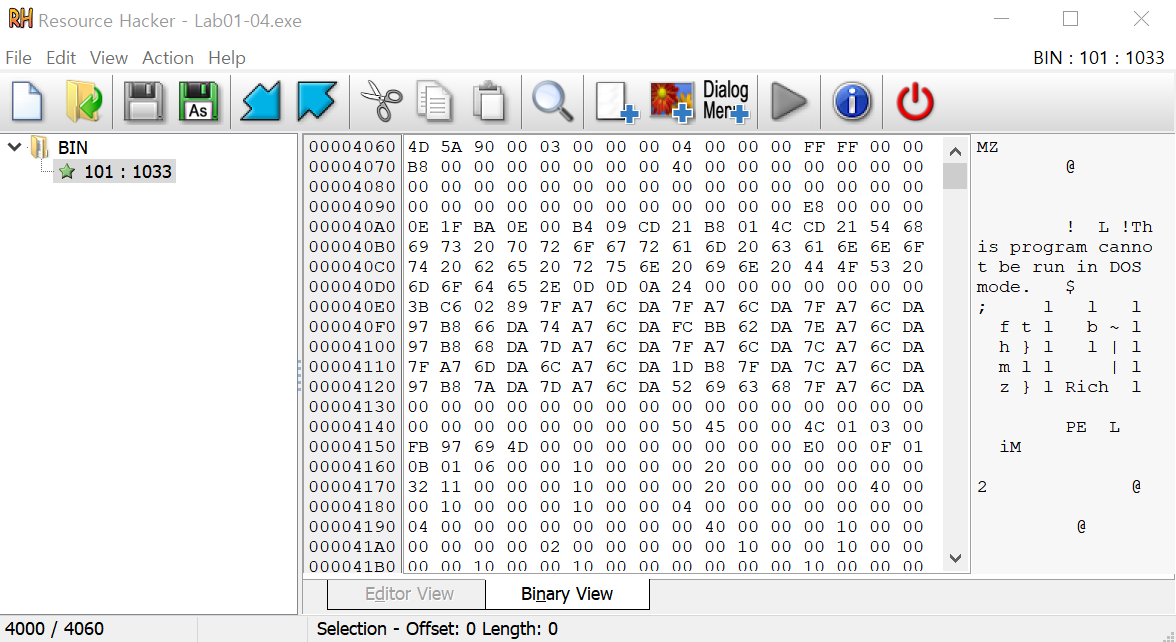
◦ 감염된 장비에서 이 악성코드를 발견하기 위해 사용한 네트워크 기반의 증거는 무엇인가?

위의 사진에서 확인할 수 있듯이 strings로 확인한 결과 URLDownloadToFileA라는 함수와

<http://www.practicalmalwareanalysis.com/updater.exe> 라는 url을 확인할 수 있었다.

◦ 이 파일은 리소스 섹션에 하나의 리소스를 Resource Hacker를 이용해 리소스를 점검하고 추출해 보자. 리소스로부터 무엇을 알 수 있는가?





Resource Hacker를 이용해 리소스를 추출해 strings로 확인한 결과 GetWindowsDirectoryA, Get TempPathA, URLDownloadToFileA 등의 함수를 확인할 수 있었다. 이를 통해 무언가를 다운로드 하게 하는 파일임을 확인할 수 있었다.