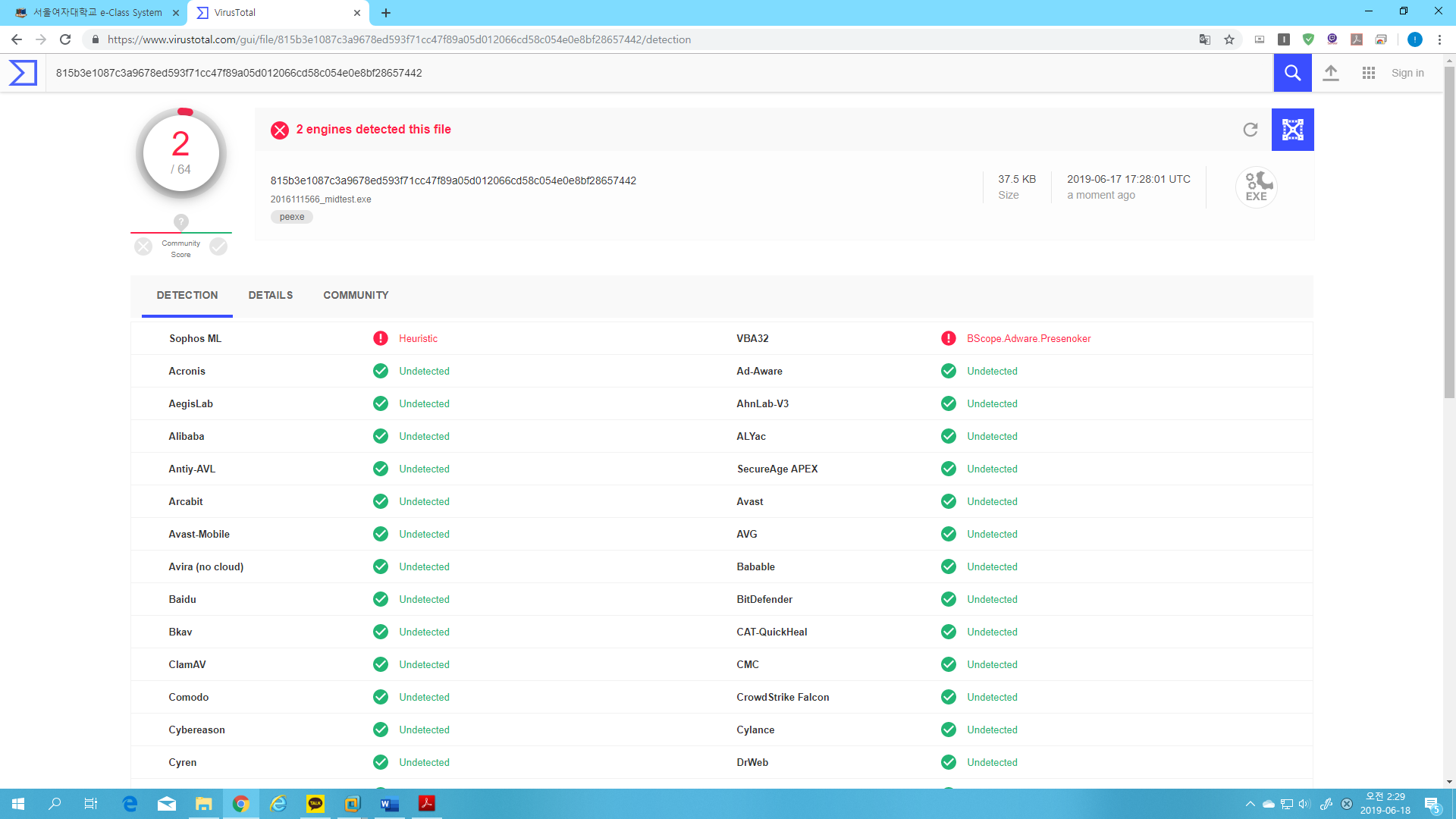
**기말 분석 보고서**

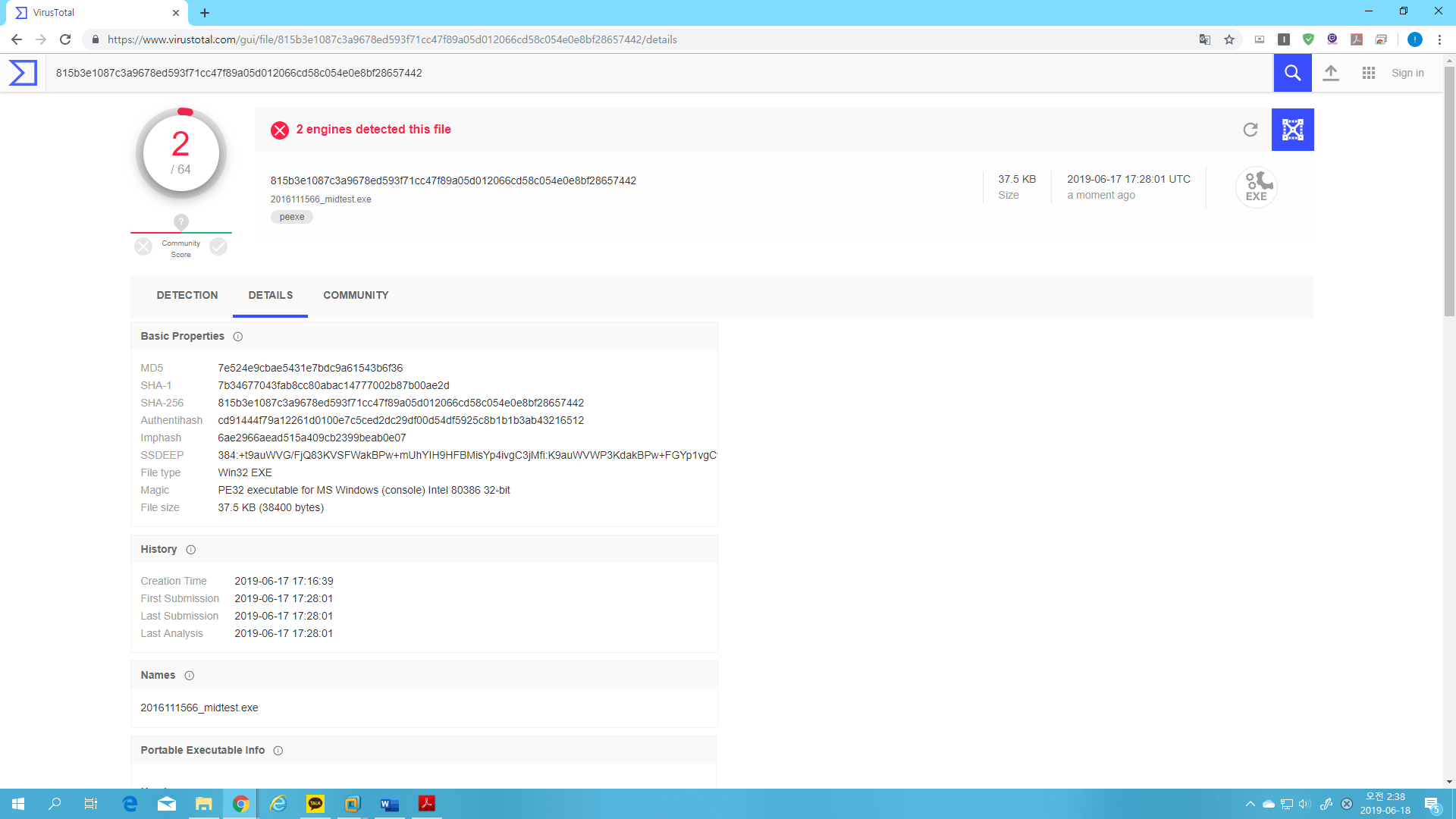
소프트웨어융합학과 이나경

**<정적분석>**

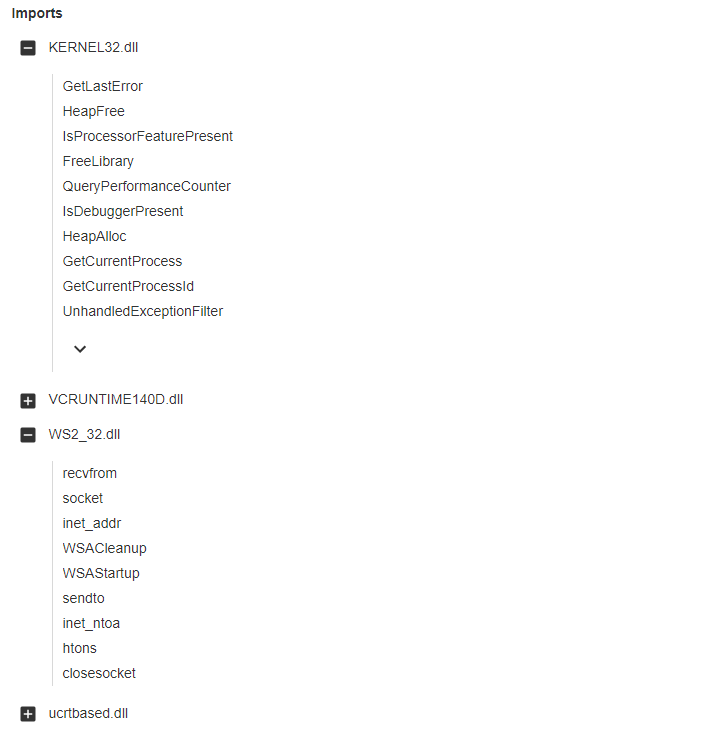
<http://www.virustotal.com>을 통해 안티바이러스 스캐닝을 해본 결과



총 두곳에서 위험이 검출되었다. 한곳에서는 불확실한 악성코드로 진단하였고 한곳에서는 Adware로 진단하였다.



DETAILS에 가보면 상세한 파일의 정보를 확인할 수 있는데 파일 생성시간, 컴파일 된 시간, 파일의 크기, 형식 등을 알 수 있다.



임포트된 함수의 목록도 확인할 수 있었다.

KERNEL32.dll의 함수 중 몇가지만 살펴보자면

GetCurrentProcess : 현재 실행되고 있는 프로세스의 핸들을 얻어오는 함수

HeapAlloc : Windows 운영체제에서 동작하는 프로그램의 Heap영역에서 동적으로 메모리를 할당할 때 사용하는 함수

HeapFree : HeapAlooc 함수를 사용하여 할당받은 메모리를 해제하는 함수

등등으로 메모리를 할당받아 사용하는 프로그램임을 알 수 있다.

WS2\_32.dll의 함수에 대해 살펴보면

WSAStartup : 윈속 기능을 사용하기 위해 윈속 초기화하는 함수

WSACleanup : 윈속을 해제하는 함수

Socket : 소켓을 생성하여 반환하는 함수

Sendto : 데이터를 송신하는 함수 (buffer형식으로 송신)

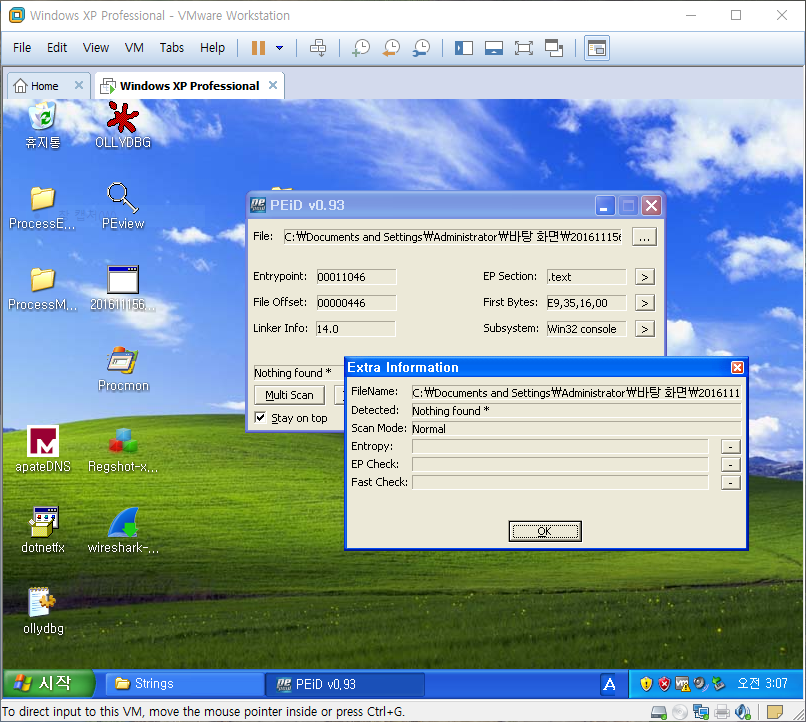
Recvfrom : 소켓으로부터 데이터를 읽는 함수 (buffer형식으로 데이터를 받는다)

inet\_addr : 숫자와 점으로 이루어진 IP 문자열을 long형의 숫자 IP 주소로 바꾸어주는 함수로

sockaddr\_in 에서 .sin\_addr.s\_add 값을 long형의 숫자 IP값을 넣어 주어야 하는데 이 때 사용된다.

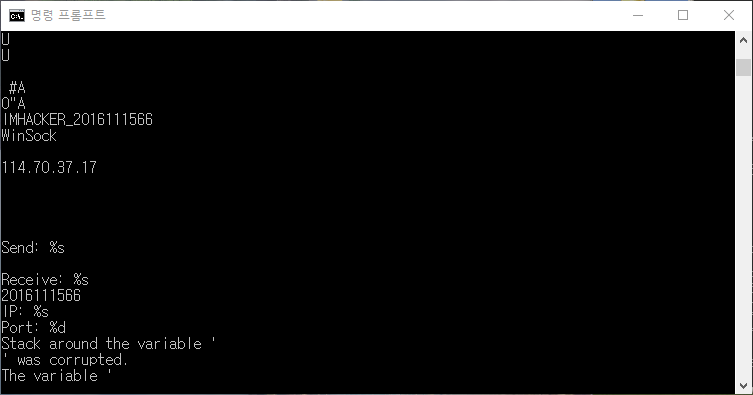
윈도우 소켓을 이용한 데이터 송수신을 하는 파일인 것을 알 수 있다.

PEiD를 통해 패킹 여부를 확인했지만



패킹은 되어있지 않았다.

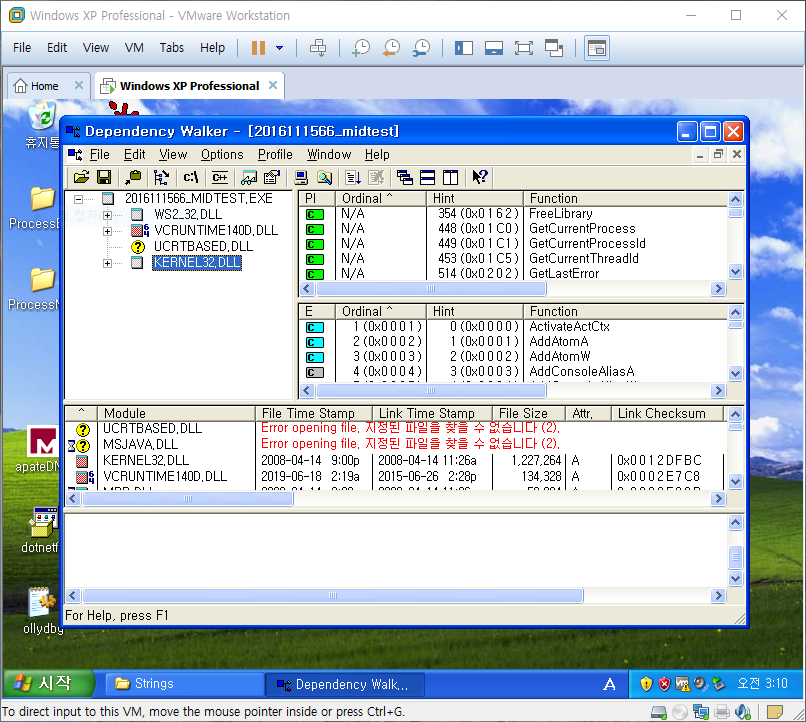
Strings를 통해 코드 내의 문자열을 검색하고 추출한 결과



AttackBuffer, CnCServer\_Size, AttackServer, CnCServer, IMHACKER\_2016111566 WinSock, Send, Receive 등등의 문자열을 확인할 수 있었고 114.70.37.17이라는 IP주소도 찾아볼 수 있었다.

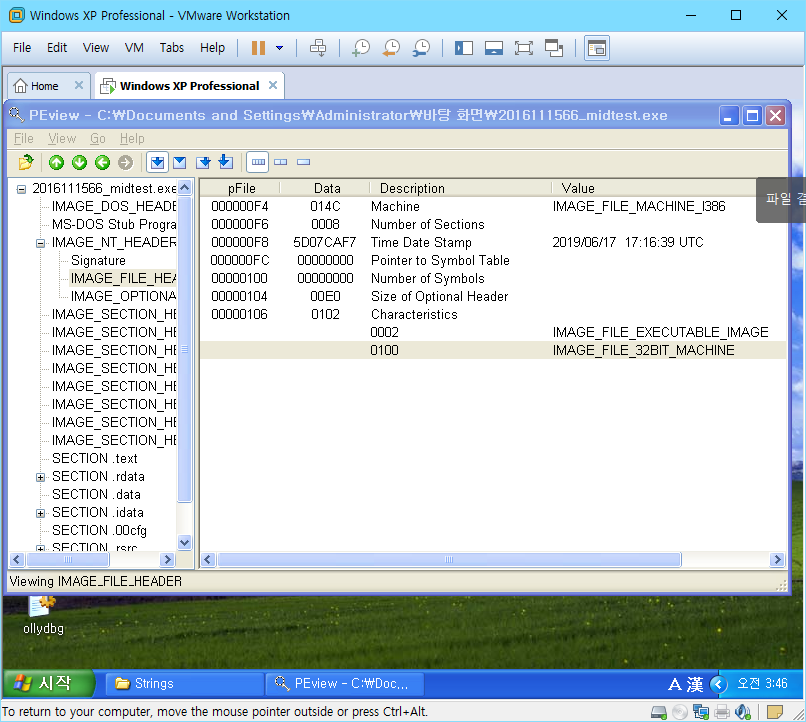
114.70.37.17이라는 서버와 네트워크 통신을 하며 데이터를 주고받는 프로그램인 것으로 추측된다.

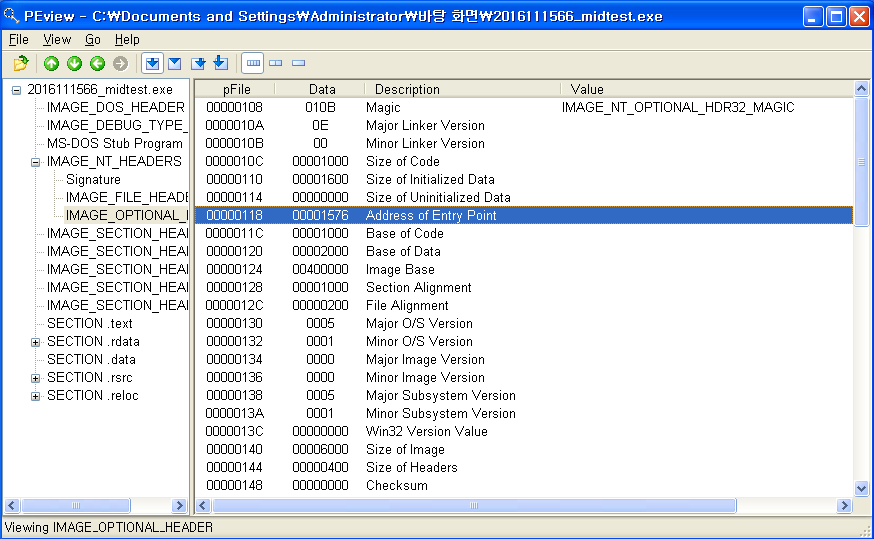
Dependency Walker로 동적 링크 함수를 탐색해본 결과



WS2\_32.DLL이 포함된 것을 확인할 수 있었는데 해당 파일이 네트워크 연결 및 관련 작업을 수행한다는 것을 알 수 있다.

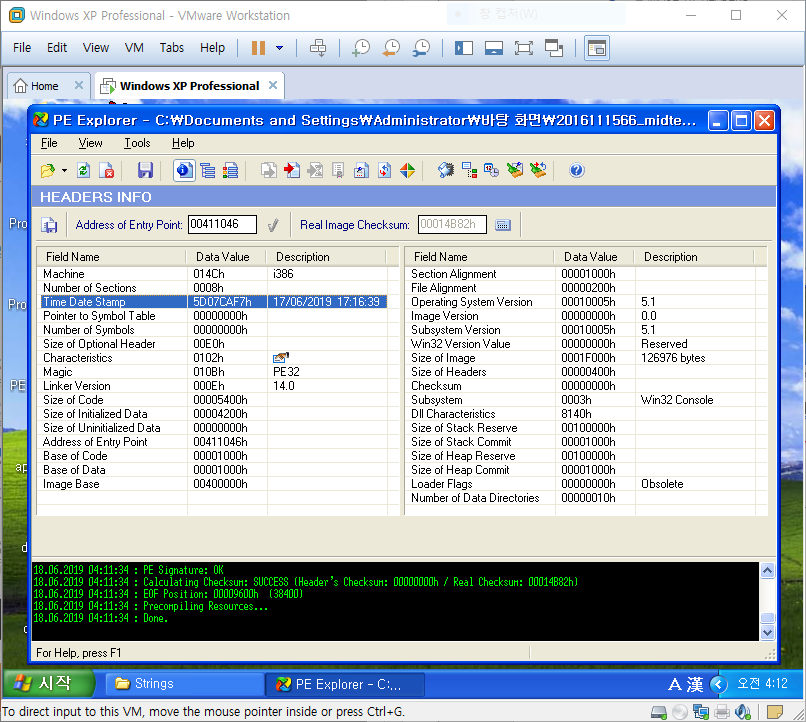
PEview로 파일 헤더를 확인한 결과 파일이 컴파일된 시간 등을 확인할 수 있었다.



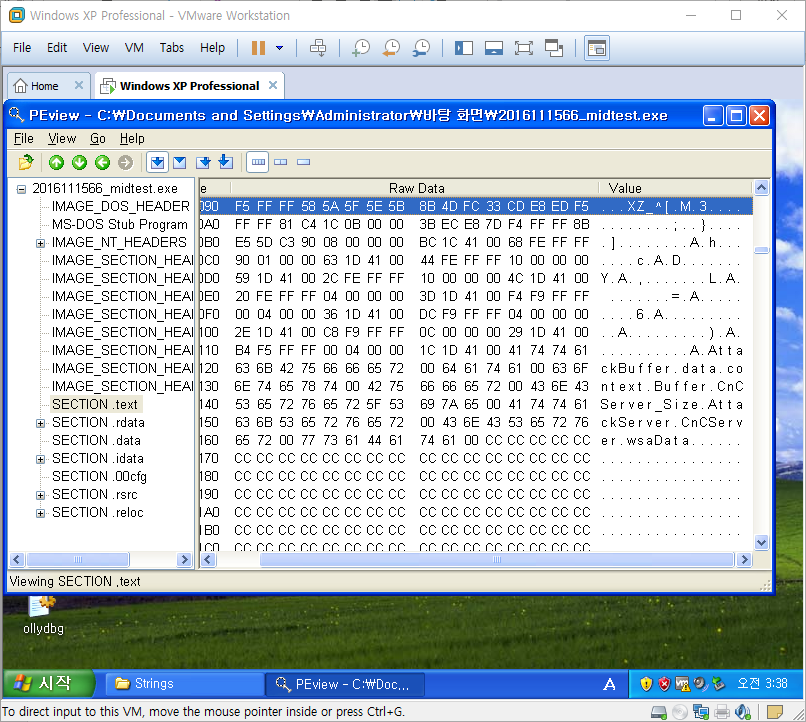


파일의 엔트리 포인트 또한 확인할 수 있다.

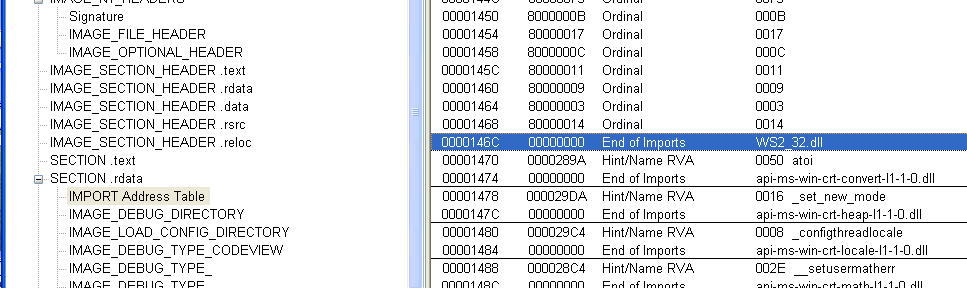
PE파일 헤더 정보는 PEBrowse나 PE Explorer에서도 확인할 수 있었다.



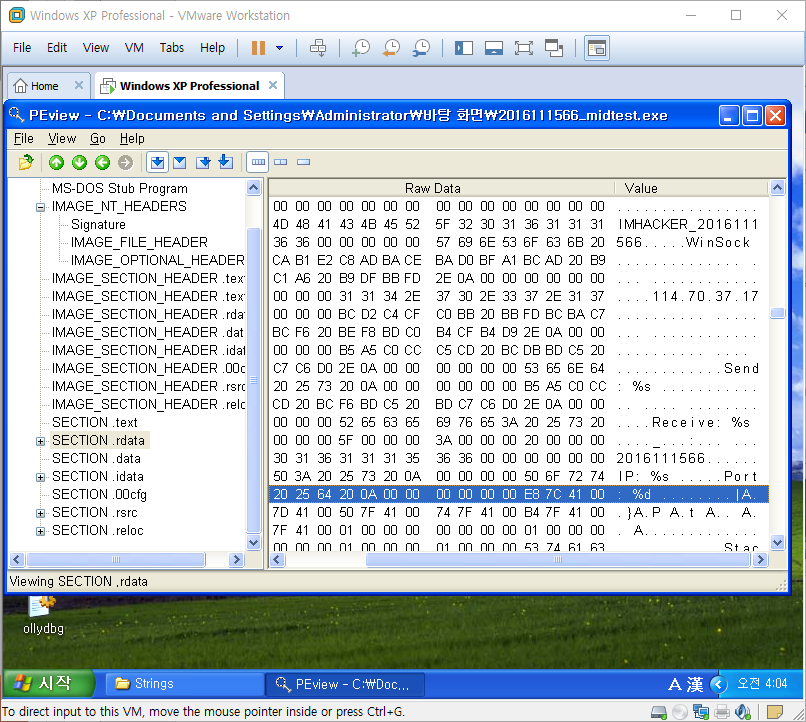
Subsystem 을 확인하면 Win32 Console로 콘솔 기반 프로그램인 것을 알 수 있다.



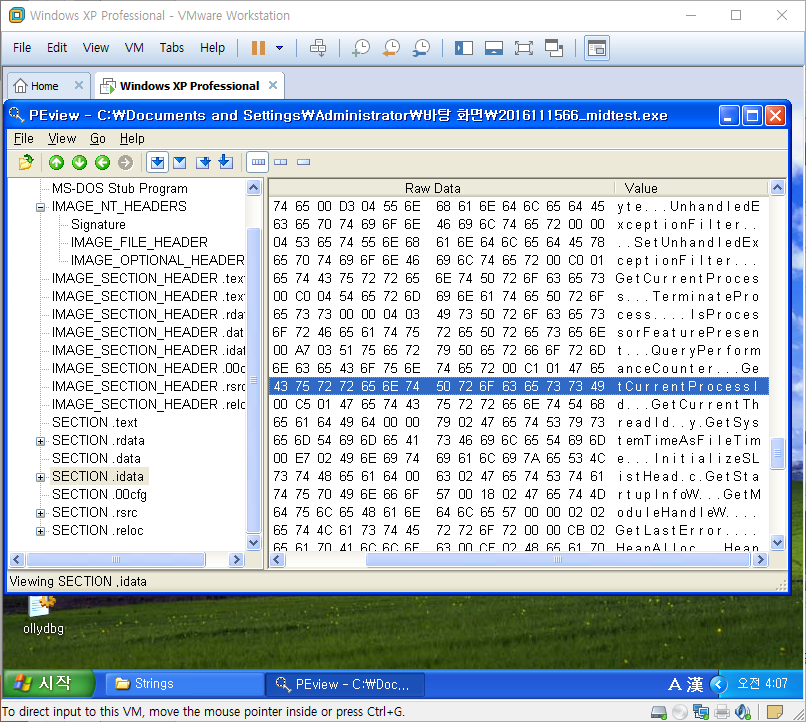
또한 실행하는 코드를 담고있는 .text 섹션에서 AttackBuffer, CnCServer\_Size, AttackServer, CnCServer 등의 문자가 확인되었다.



IMPORT Address Table는 프로그램에서 사용되는 라이브러리에서 어떤 함수들을 사용하고 있는지,함수명, 함수시작 주소 등에 대한 정보를 기술한 테이블이다. 이를 확인하면 WS2\_32.dll이 임포트되어있는 것을 알 수 있다. WS2\_32.dll은 Winsock(윈도우 소켓) 프로그램을 만들기 위해 필요한 라이브러리이다. 따라서 윈도우 소켓 기반 네트워크 프로그래밍 되어있는 프로그램으로 추정된다.



프로그램 내의 전역에서 접근 가능한 읽기 전용 데이터를 담고 있는 .rdata 섹션에서는 IMHACKER\_2016111566, WinSock,114.70.37.17,Send,Receive 등의 문자열을 확인할 수 있었다.

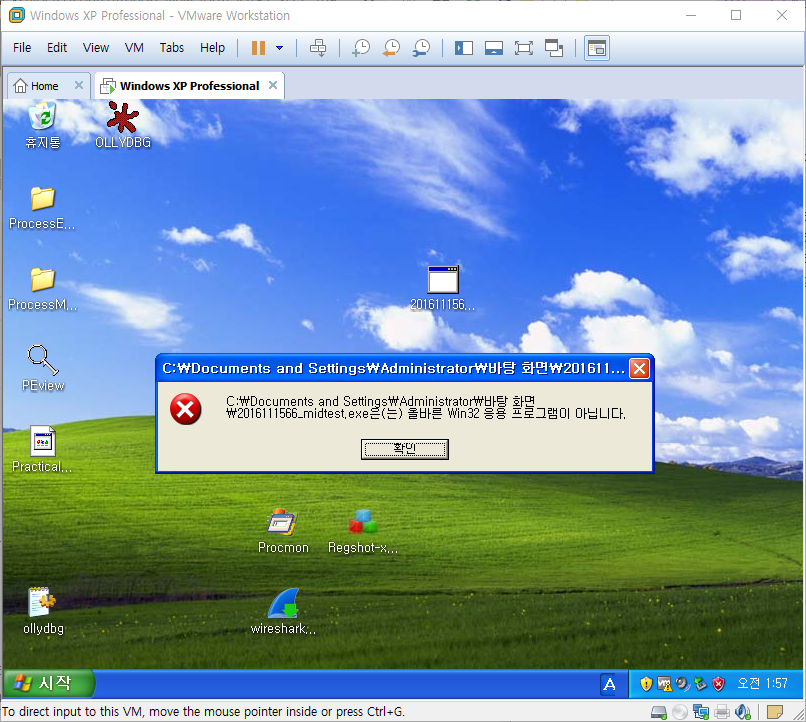


임포트 함수 정보를 저장하는 .idata에서 GetCurrentProcess, GetCurrentThread,

GetSystemFileAsFileTime, HeapAlloc, HeapFree 등등의 임포트된 함수를 찾아볼 수 있다.

**<동적분석>**

해당 파일을 Windows XP에서 실행할 경우

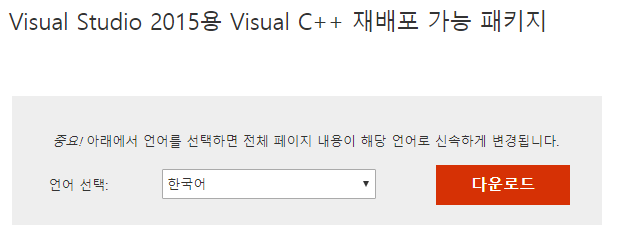


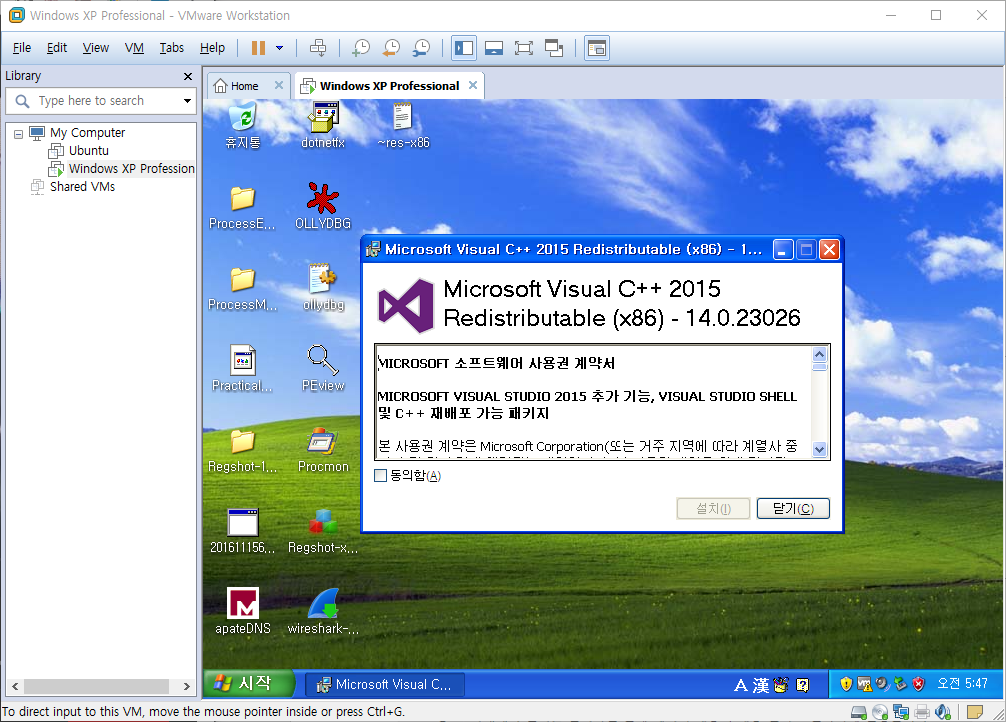
올바른 Win32 응용 프로그램이 아니라는 오류가 나타난다.

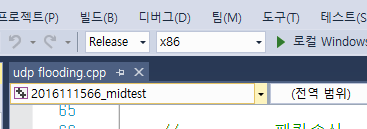
이는 .Net FrameWork 4.5이상으로 개발한 프로그램을 XP에서 실행할 경우 나타나는 오류로

Windows XP에서는 배포버전만 실행할 수 있으므로

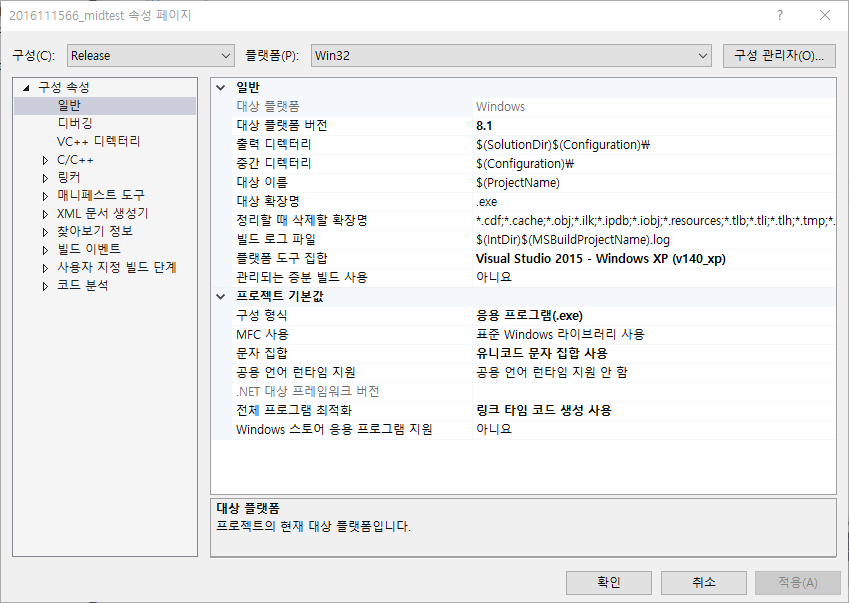
Windows XP 환경에 Visual Studio 2015용 Visual C++ 재배포 가능 패키지를 설치한다.







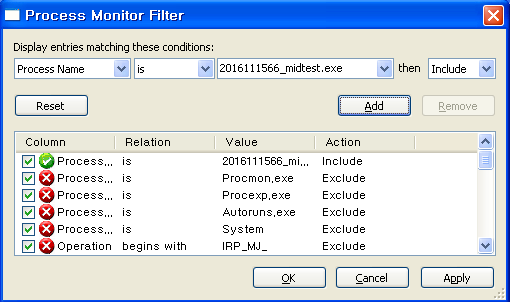
Visual studio에서 구성을 Debug에서 Release로, 플랫폼은 x86(Win32)버전으로 설정한 후



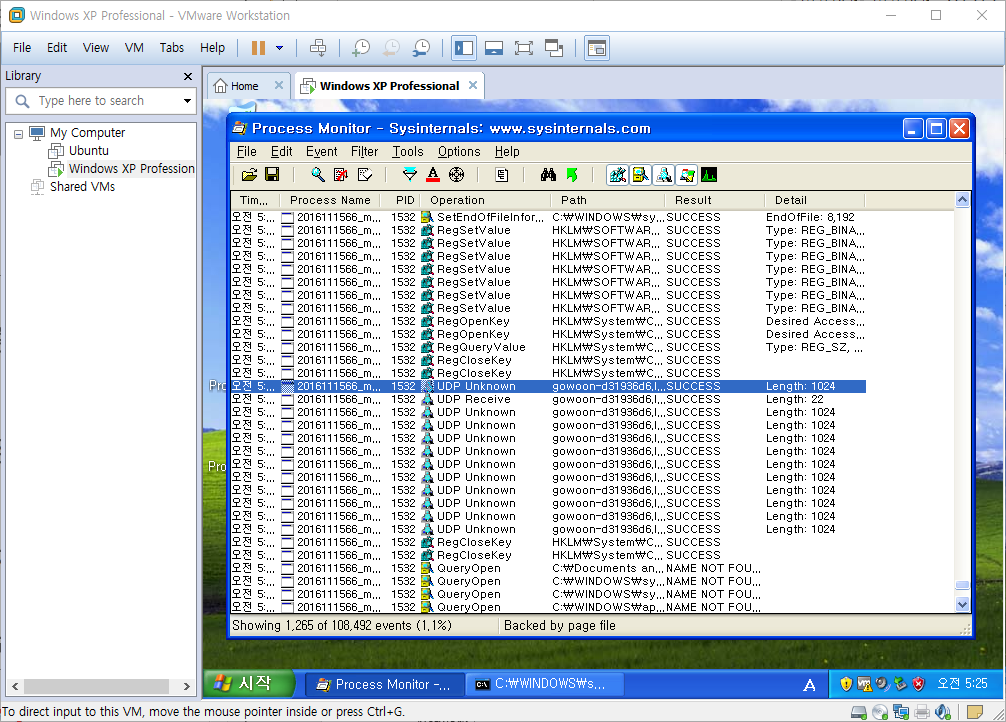
프로젝트 속성>일반>플랫폼 도구 집합을 Winsows XP로 설정하고 빌드하면 된다.

프로그램을 실행해보았으나 실행 후 바로 종료되었다.

따라서 프로그램을 분석하기 위해 프로그램을 실행시키고 Process Monitor로 확인하였다.



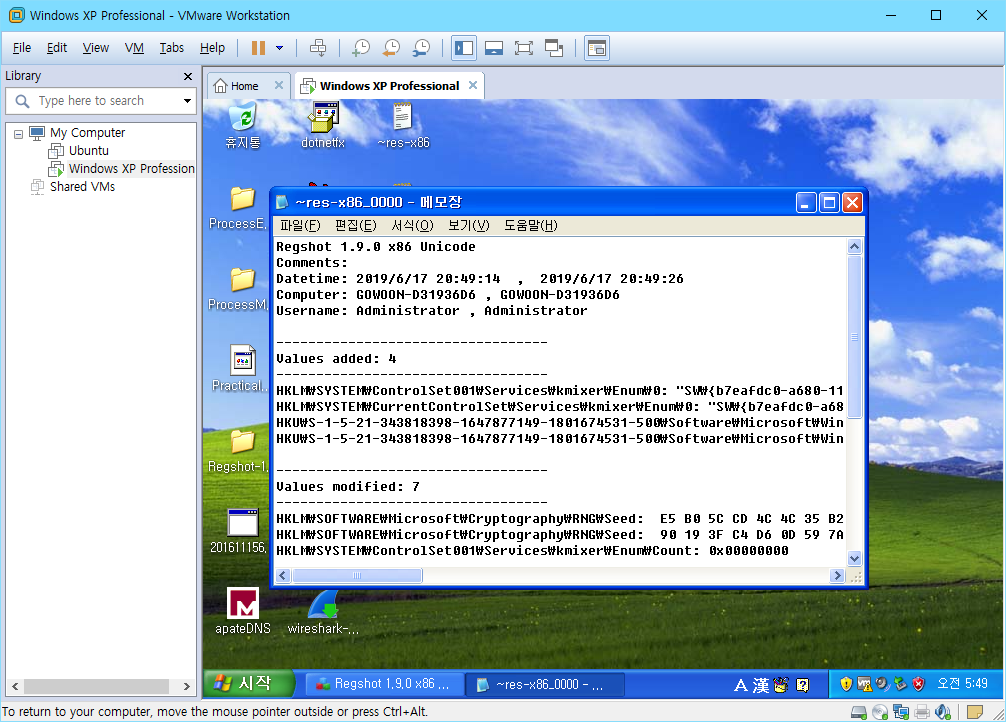
다른 프로그램과 구분하기 위해 Process Name을 설정하여 필터링하였다.



그 결과 네트워크 연결을 통한 활동을 하는 것이 검출되었는데

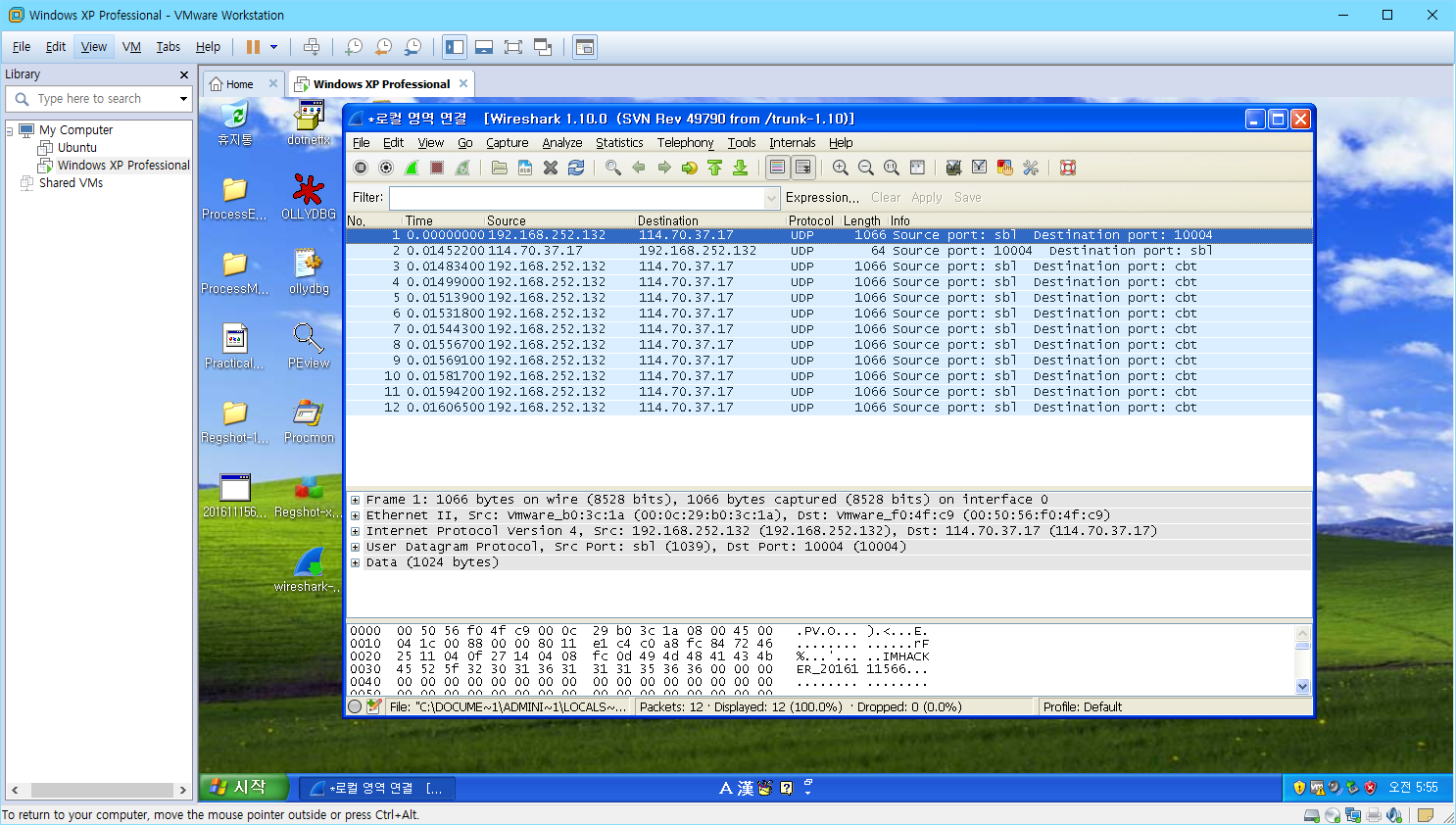
UDP 형식의 데이터를 전송하고 수신한 후 다시 10회 전송하는 활동이 발견되었다.

Regshot으로 코드 실행 전후 레지스트리 스냅샷을 찍어 비교하였다.

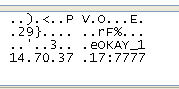


추가된 key는 없는 것으로 확인되었다.

Wireshark로 패킷을 관찰한 결과 UDP통신이 관찰되었고



처음 114.70.37.17 서버의 10004 포트에 ‘IMHACKER\_2016111566’이라는 데이터를 보내고



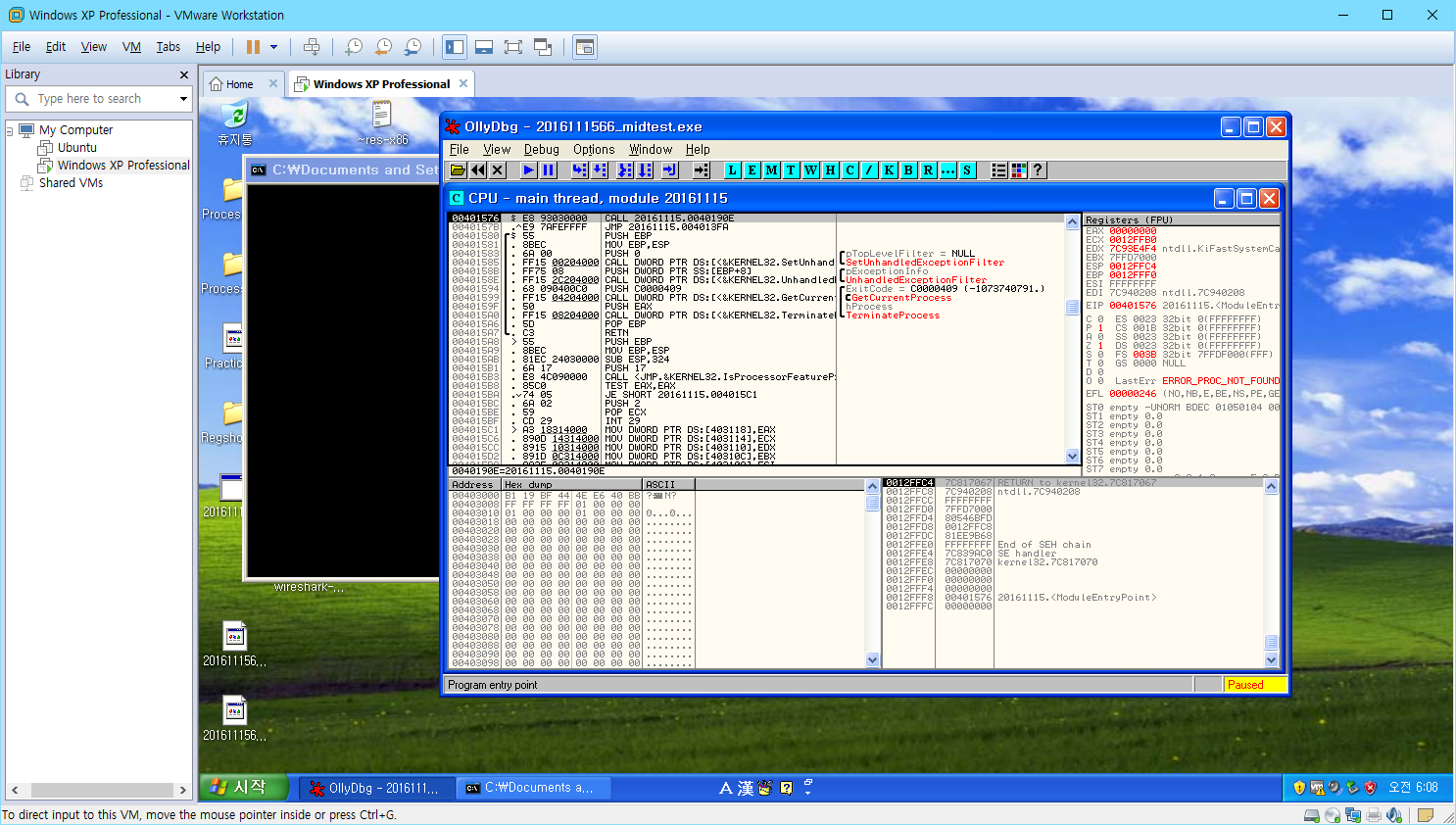
114.70.37.17로부터 ‘OKAY\_114.70.37.17:7777’이라는 데이터를 받는다.



그 후 114.70.37.17서버의 7777포트로 2016111566이라는 데이터를 10회 전송한다.

**<리버싱>**

OllyDbg를 사용하여 리버싱을 수행한다.

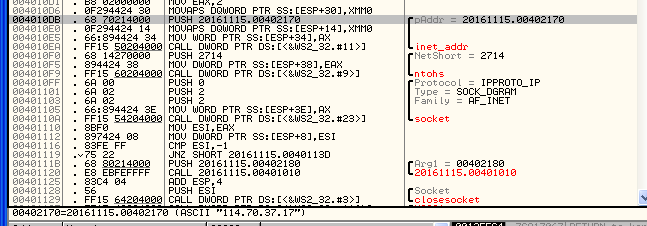


F7키과 F8키를 이용해 함수를 하나하나 확인하는 방법도 있지만 search 기능을 통해 함수를 찾아보도록 한다.



팝업->search for-> All referenced text strings로 모든 문자열을 조회한다.

첫번째 IP주소형식의 문자열을 클릭하면



PUSH로 어딘가에 삽입되어지는 것을 확인할 수 있다.

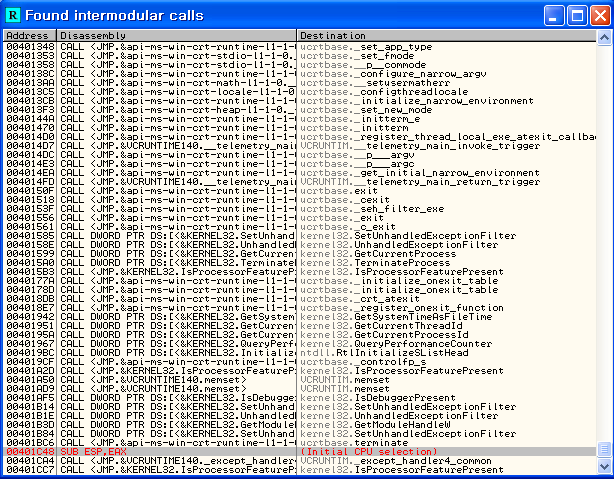
Go to[Ctrl+G] 명령어를 통해 00402170 주소로 이동하여 값을 확인하면



문자열을 확인할 수 있다.



그 외에도 00402138 주소에 IMHACKER\_2016111566 등의 문자열이 저장되어있는 것을 확인할 수 있다.



팝업->search for->All intermodular calls로 모든 호출함수를 조회한다.



Inet\_addr 함수로 주소를 설정하고 sendto로 데이터를 전송, recvfrom으로 데이터를 수신받은 후 다시 inet\_addr로 주소를 설정하고 sendto로 데이터를 전송하는 프로그램으로 추정된다.



그 중 첫번째 데이터를 보내는 것으로 추정되는 sendto함수 부분으로 이동한다.



WS2\_32.sendto 함수로 데이터를 전송하는 함수인 것을 알 수 있다.

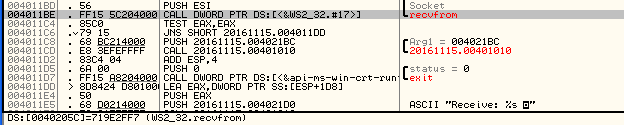


이 함수를 실제로 실행하는지 확인하기 위해 BP[F2]를 설정하고 실행[F9]해본다.

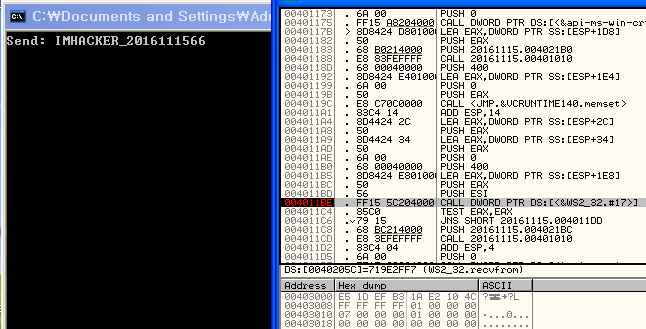


실제로 이 함수를 사용하는것으로 확인되었다.

그 외에도 WS2\_32.inet\_addr 함수로 IP주소를 설정하여 통신하는 것으로 확인된다.



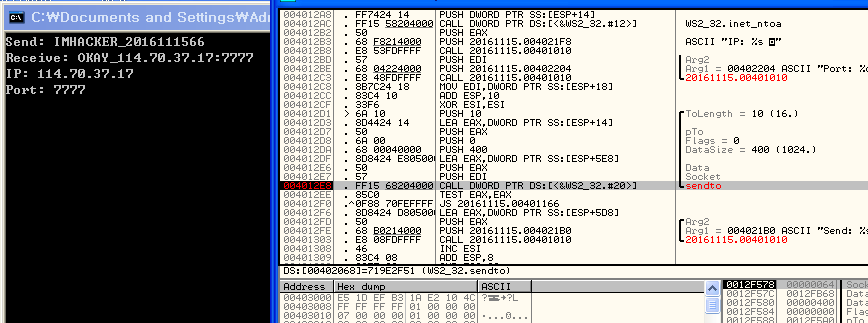
데이터를 수신하는 것으로 추정되는 recvfrom으로 이동하자 WS2\_32.recvfrom으로 데이터를 수신하는 함수가 발견되었고



BP를 설정한 후 실행해보았더니

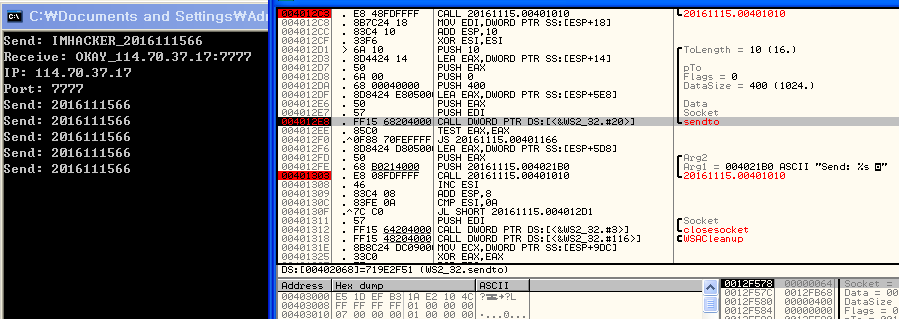
실행창에 Send: IMHACKER\_2016111566이 출력되며 멈추었다.

두번째 sendto 함수에 BP를 설정하고 실행하였더니



실행창에 Receive: OKAY\_114.70.37.17:7777이 출력되고 수신한 데이터로 보이는 문자열에서 IP주소와 포트번호를 분리하여 각각 IP와 Port로 설정하는 것으로 보인다.

그 후 F9로 실행하면 sendto함수와 출력이 반복되는 것을 알 수 있다.



10회 반복 후 종료되었다.

리버싱을 통해 코드를 분석한 결과

처음 어떤 주소(114.70.37.17)로 IMHACKER\_2016111566라는 데이터를 보내면 IP와 포트번호가 담긴 데이터를 수신한다. 그 후 그 주소로 sento함수를 10회 반복하여 데이터를 전송하는 것을 확인할 수 있다.

**<분석결과>**

1. 분석결과 2016111566\_midtest.exe은 지정된 서버(114.70.37.17)의 포트번호(10004)로‘IMHACKER\_2016111566’ 문자열을 전송한 후
2. 서버로부터 ‘OKAY\_114.70.37.17:7777’라는 응답을 받고
3. 받은 서버의 포트번호로 2016111566이라는 문자열을 10회 반복하여 전송하는

UDP Spoofing 프로그램이다.