5주차 1차시. 리버스엔지니어링 이해

【학습목표】

1. 리버스 엔지니어링에 대해 설명할 수 있다.

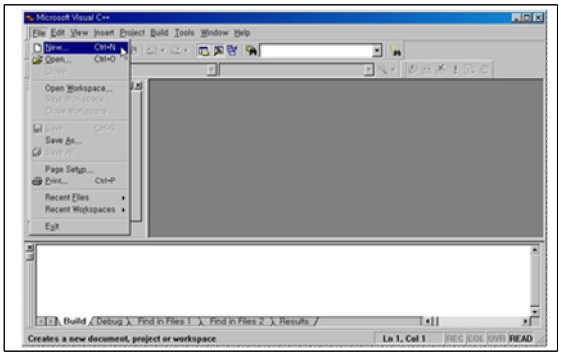
학습내용1 : 리버스 엔지니어링에 대한 이해

1. 리버스 엔지니어링(Reverse Engineering)

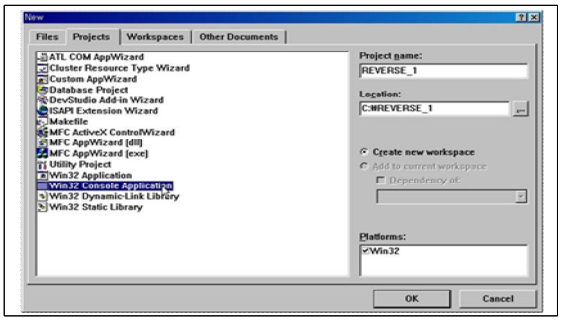
* 역공학(逆工學)
* 장치나 시스템의 구조를 분석하여원리를 발견하는 과정
* 이미 만들어진 프로그램 동작 원리 이해,유사 프로그램 제작
* 프로그램의 보안 문제, 동작 문제, 오류 등을이용, 제품 출시 전 문제점과 오류 제거 · 검토
* 바이러스(또는 웜), 백신 제작

2. 리버스 엔지니어링을 위한 준비

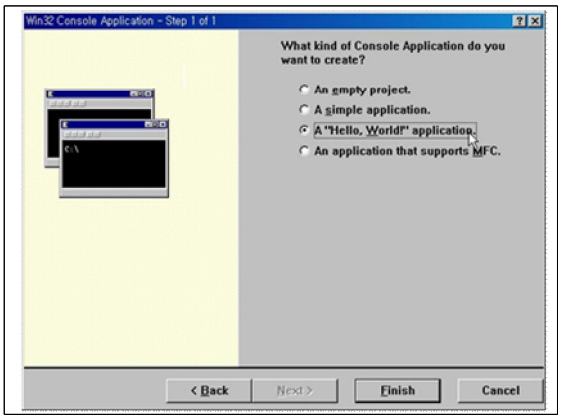
* 비주얼 C++ 6.0
* 새로운 파일 생성, [File]-[New]



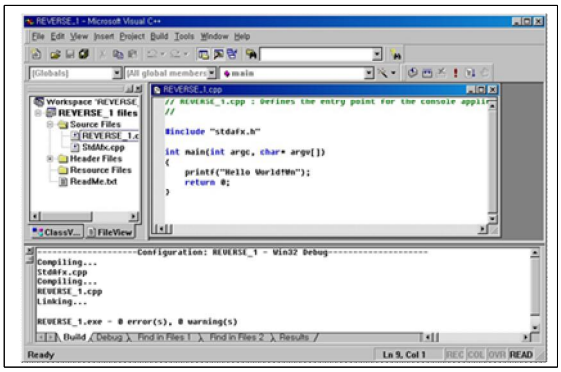
* Win32 Console Application



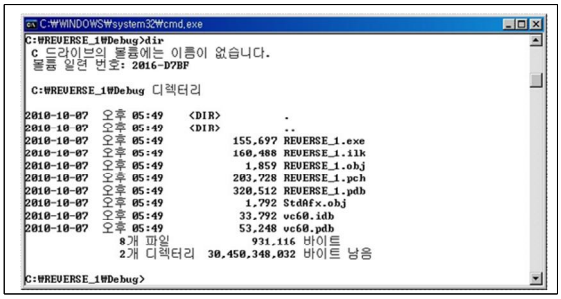
* Hello World가 출력되는 기본 프로그램 선택



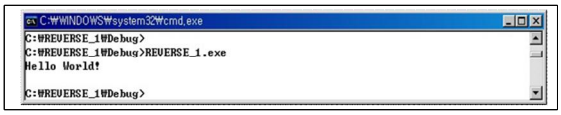
* 프로그램 생성, [Build]-[Build REVERSE_1.exe]



* REVERSE_1.exe 파일 확인



* REVERSE_1.exe 파일 실행

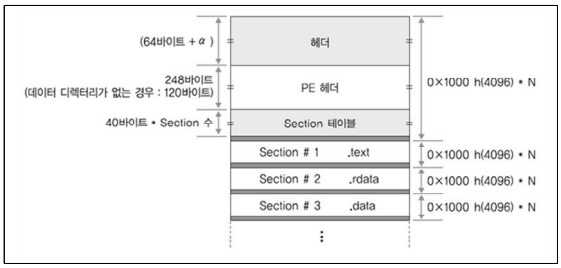


학습내용2 : PE 파일에 대한 이해

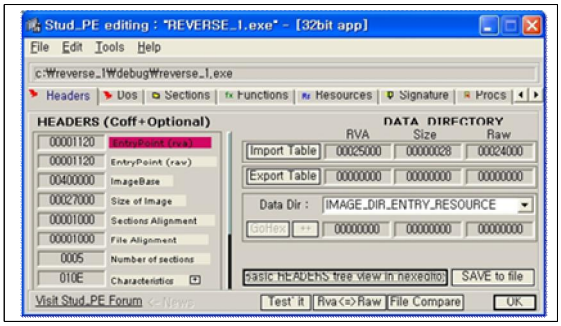
1. PE 파일 형식(Portable Executable File Format)

* 1993년 MS에서 표준안 작성
* 메모리에서도 디스크에 저장된 파일 형태로바로 실행될 수 있도록 설계
* Win32의 기본 파일 형식
* EXE 파일, 동적 링크 라이브러리(DLL, Dynamic Linking) 파일도 PE 파일 형식

* [그림 5-6] PE 파일 구조



* 파일 구조를 확인을 위해STUD_PE 툴과 XVI32 HEX 에디터를 사용

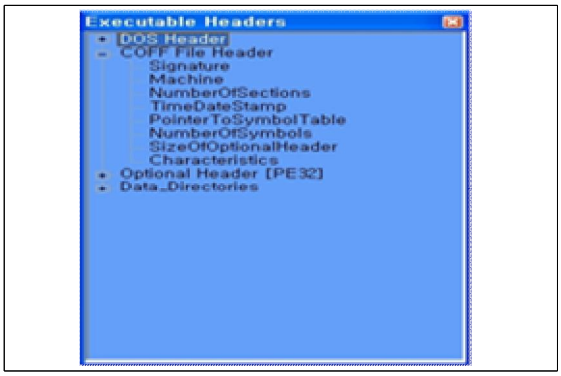


[그림 5-7] Stud_PE 실행

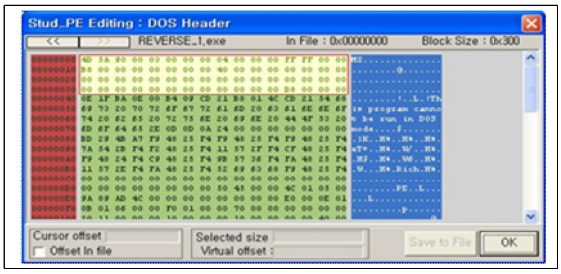
2. DOS 헤더

* DOS와 호환하기 위해 사용, 64바이트

<Basic HEADERS tree view in hexeditor> 버튼 클릭- (a) 창에서 선택한 항목이 (b) 창에서 붉은 색 박스 안에 표시



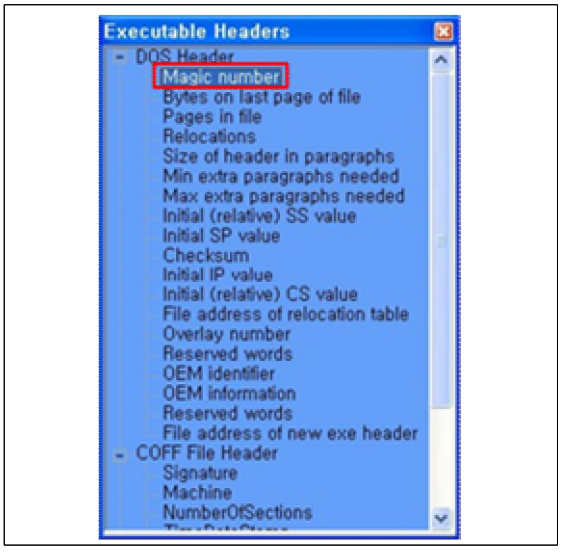
(a) PE 파일 구조

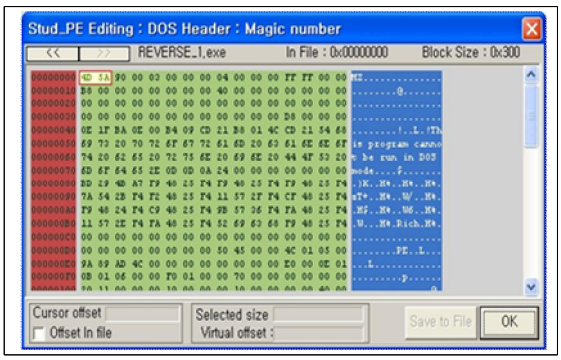


(b) HEX 에디터에서 연REVERSE_1.EXE 파일

3. Magic Number

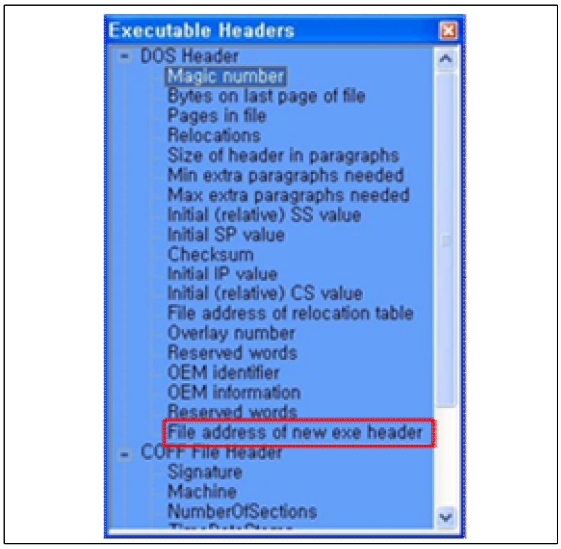
* DOS 헤더 항목을 확장한 Magic Number 부분은 HEX 에디터의 오른쪽 블록에서 MZ 문자열 확인
* MZ는 DOS 헤더의 시그니처로 사용
* 운영체제가MZ 문자열을 통해해당 파일을 PE 파일 인식

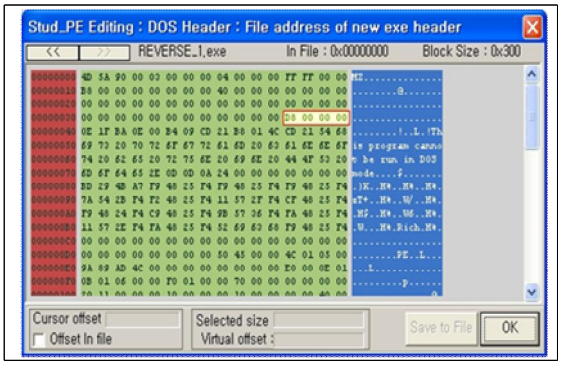




4. File address of new exe header

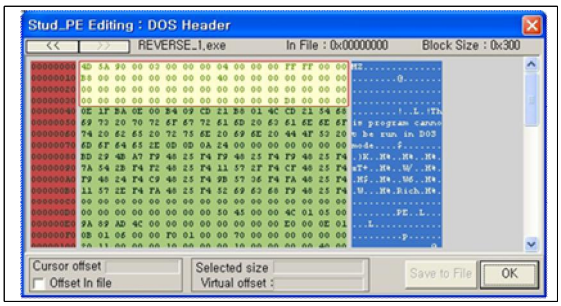
* e_lfanew 값이라고도 함
* PE 헤더의 주소 값 기록
* REVERSE_1.EXE의'File address of new exe header'값은 000000D8





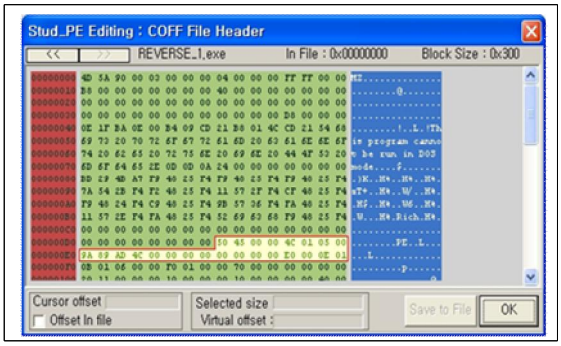
5. DOS Stub 코드

File address of new exe header 값이 기록된 부분(00000040)과 000000D8 주소 값 사이에 표시한 부분
This program cannot be run in DOS mode 와같이 적절하지 못한 상황에서 PE 파일이 실행될 때 경고문 등에 대한 데이터 포함



6. PE 헤더

* PE 파일의 구성 정보 보관



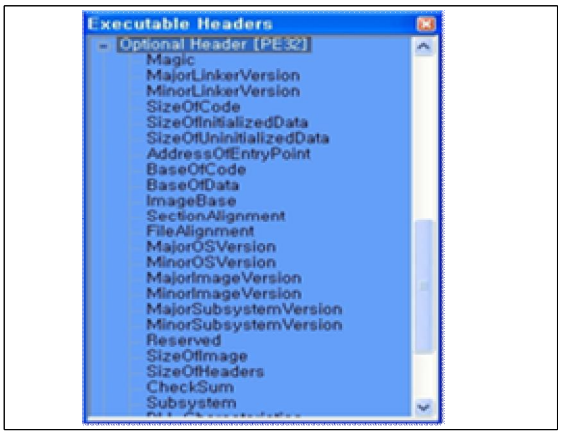
[그림 5-11] PE 파일 헤더 영역 확인

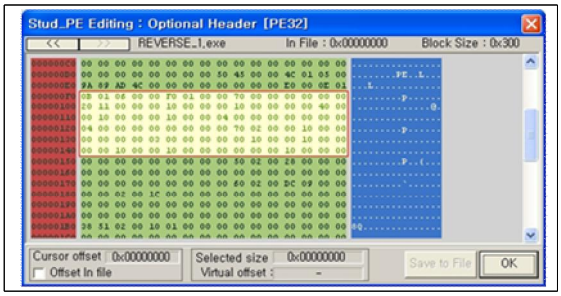
7. PE 헤더 필드

① Machine(0x014C, 2바이트)
CPU ID
IA32(Intel Architecture 32)의 경우 : 0x014C
IA64일 경우 : 0x0200
② NumberOfSections(0x0005, 2바이트)
파일에 존재하는 섹션의 개수
③ TimeDateStamp(0x4CAD899A, 4바이트)
파일이 생성된 시간과 날짜
④ SizeOfOptionalHeader(0x000E, 2바이트)
PE 파일 헤더 뒤에 오는옵션 헤더(optionalheader)크기를 16으로 나눈 값
⑤ Characteristics(0x010E, 2바이트)
파일의 속성
일반 실행 파일의 경우 값 : 0x010F
이 속성을 통해 파일이 exe인지 dll 인지 구분

8. 섹션헤더

* Optional 헤더라고도 불림
* PE 파일에서 프로그램과 관련한필수 정보를 담고 있는 부분





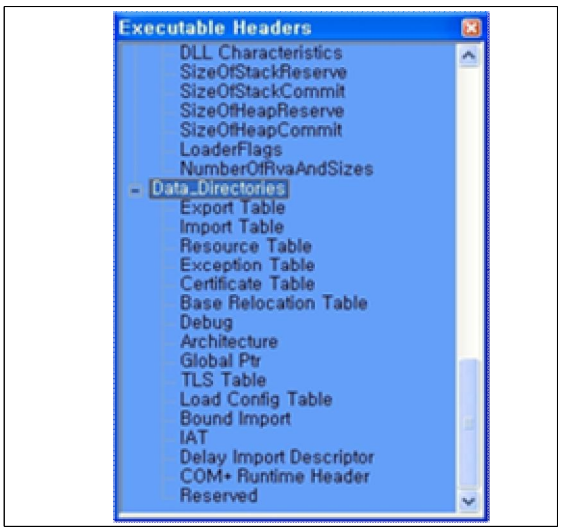
9. 섹션헤더 필드

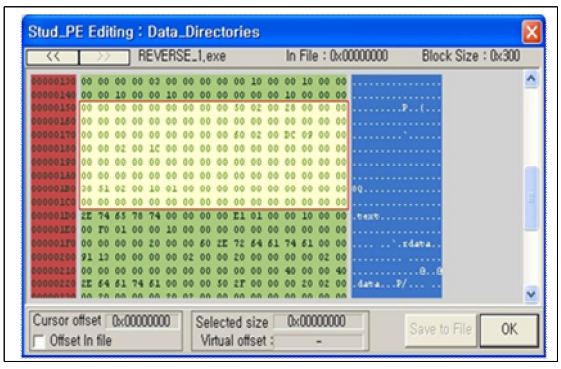
① Magic(0x010B, 2바이트)
시작 위치의 필드
Optional 헤더 구분 시그니처(값 0x010B)

② AddressOfEntryPoint(0x00001120, 4바이트)
PE 파일이 메모리에 로드된 후처음 실행 되는 코드의 주소
가상 주소의 절대 값이 아닌이미지 베이스부터의 오프셋 값인상대 가상 주소(RVA, Relative Virtual Address)
엔트리 포인트는 .text 섹션(실행 코드를 담고 있는 메모리 영역)의시작점인 경우가 대부분
.text 섹션 헤더의 가상 주소 값과 일치하는 경우가 많음

10. 섹션 헤더 - Data Directory

* 다른 DLL에서 함수를 가져오거나본 파일의 함수를 내보낼 때 해당 정보를 기록
Export Table(16바이트)
Export 함수들에 대한 Export Table의 시작 위치와 크기
Import Table(16바이트)
Import 함수들에 대한 Import Table의 시작 위치와 크기
* 다른 DLL에서 함수를 가져오거나본 파일의 함수를 내보낼 때 해당 정보를 기록



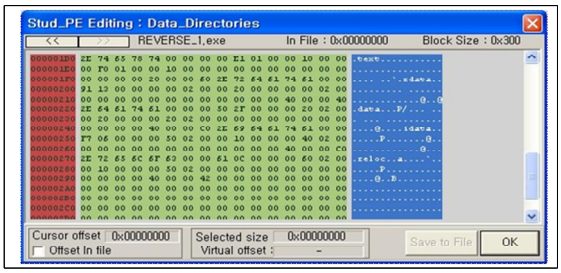


11. 섹션 테이블

* 메모리 보호를 위한 최소 단위가 페이지이므로, 페이지 단위로 속성 설정
* 속성에 위배되는 행동 시 접근 오류 발생하여 메모리 보호
* 페이지 단위 메모리 속성 설정

성질이 다른 데이터들은 하나의 페이지에 설정 불가
실행 코드, 읽고 쓰기가 가능한 데이터,읽기만 가능한 데이터는 별도 페이지에 설정

* [그림 5-14] 섹션 테이블



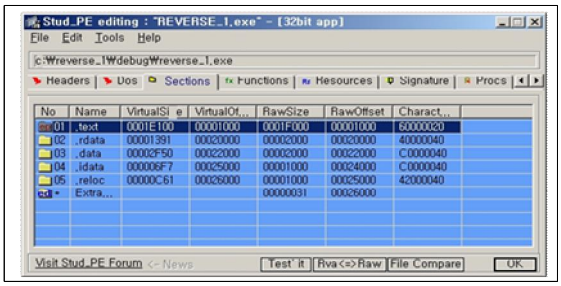
* .text 섹션

① Name(.text, 8바이트)
섹션 이름
② Virtual Size(0x0001E100, 8바이트)
섹션 크기
③ Virtual Offset(0x00001000, 8바이트)
섹션이 로드 될 가상 주소

PE 형식 내에서 이 필드도 상대 가상 주소(RVA)임
④ Size of Raw Data(0x000100F0, 8바이트)
File Alignment의 다음 배수 값까지반올림 한 섹션 데이터의 크기
⑤ Pointer To Raw Data(0x00001000, 8바이트)
섹션의 시작을 담고 있는 파일 오프셋
PE 로더는 파일 안의 섹션에서데이터 위치를 알기 위해 이 영역의 값 이용
⑥ Characteristics(0x60000020, 8바이트)
섹션의 실행 코드, 초기화된 데이터,비초기화된 데이터와 같은 Flag(플래그)를 가지고 있음
메모리에 대한실행(eXecute), 읽기(Read), 쓰기(Write) 속성 지정

11. 섹션 테이블

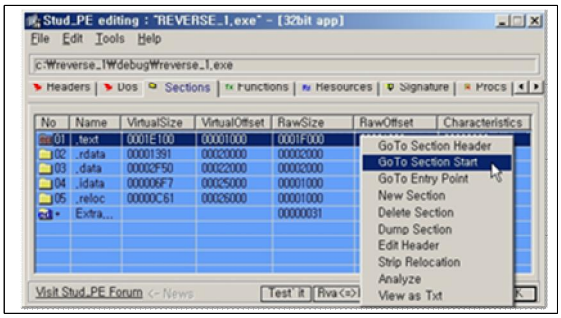
이 섹션 테이블의 내용을Stud_PE의 [Section] 탭에서 각 섹션별로 확인



[그림 5-15] 섹션 테이블 내용

12. 섹션

* 동일한 성질의 데이터가 저장되는 영역
* 각 섹션은 해당 섹션을 선택→ 마우스 오른쪽 버튼의 팝업 메뉴[Go To Section Start]로 해당 값 조회



【학습정리】

1. 리버스 엔지니어링은 역공학이라고 변역되고, 장치나 시스템의 구조를 분석하여 원리를 발견하는 과정이다.

2. PE 파일의 기본 구조는 헤더(Header), PE 헤더, 섹션 테이블, 섹션(Section)으로 이루어져 있다.