Networks and Systems Security 2 - Winter 2025

Sambuddho Chakravarty

April 25, 2025

Exercise 4 (total points: 40) (Bonus: +45 points)

Due date: May 6, 2025. Time: 23:59 Hrs.

1 Basic buffer overflow (total points: 40)

The first part of the exercise involves launching a buffer overflow attack on the given vulnerable program (`echoserver-basic`). The program is a rudimentary echo server that listens on port 40000 and waits for incoming connections. Once a client connects to the echo server, he/she can type input that is eventually returned back by the server (*i.e.* echo-ed back).

Your first exercise would be launch a basic buffer overflow attack by crafting an appropriate payload bearing a reverse TCP shell code using `msfvenom` program that is part of `msfconsole` program you can find in Kali Linux or Blackarch repositories. The crafted shell code could be sent to the server through the connection established. To establish the connection to the server you can use the `netcat` program.

Before running the program, you MUST disable ASLR option (hint: the option is available through a flag in the `/proc` filesystem.)

1.1 What to submit/Grading rubric:

1. The shellcode generated from `msfvenom` program that can be used to launch the reverse TCP shell code[15 points].
2. Script to launch the shellcode that should be able to supply the shellcode to the target program [15 points]
3. Write-up showing the screenshot on how you figured out how to craft the payload and how to use that to generate the payload bearing the shell code, with appropriate explainations of the logic and commands used [10 points]

2 ROP buffer overflow (total points: 45) (Bonus)

The second part of the exercise bears the "non-networked" version of the echo server program that prints each input as it is supplied. The stack is not set executable, unlike the previous part. You need to launch a shell code using

appropriately crafted inputs for the stack that would lead to launching a ROP (Ret2Libc) based shell code.

Here again, you MUST disable ASLR and you could use the `ROPgadget` program to identfy gadget locations in the C library routines.

2.1 What to submit/Grading rubric:

1. Input bearing the address to the ROP gadgets that are launched when some function returns along with its detailed explaination how it was chosen [25 points].

2. Script to supply the input to the target program, that updates the stack with address to the appropriate ROP gadgets [10 points]

3. Write-up showing the screenshot on how you figured out how to craft the payload and appropriate explainations of the logic and commands used [10 points]