**LAB5 Part3**

**Task1: Get Familiar with the Shellcode**

Disable Address Space Randomization

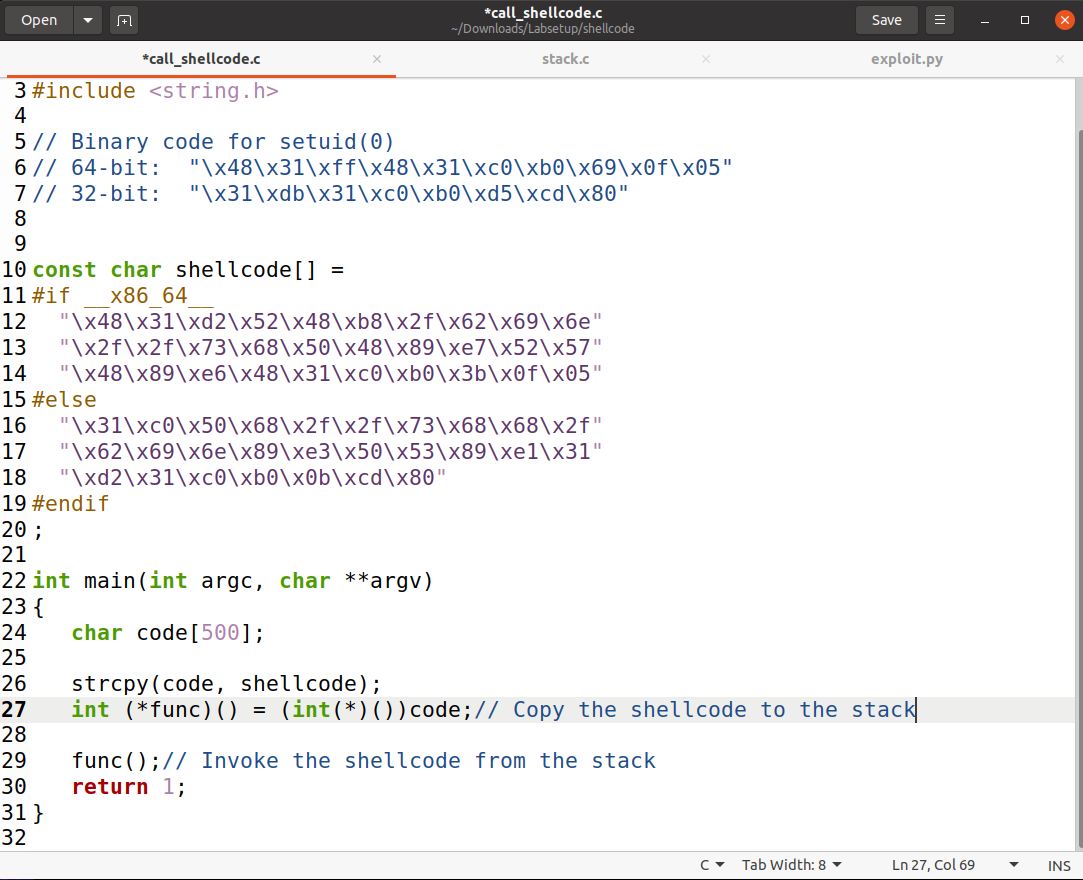
The system randomizes the starting addresses of the heap and stack. Guessing the addresses is one of the crucial steps in buffer overflow attacks. You can disable this feature using the following command:

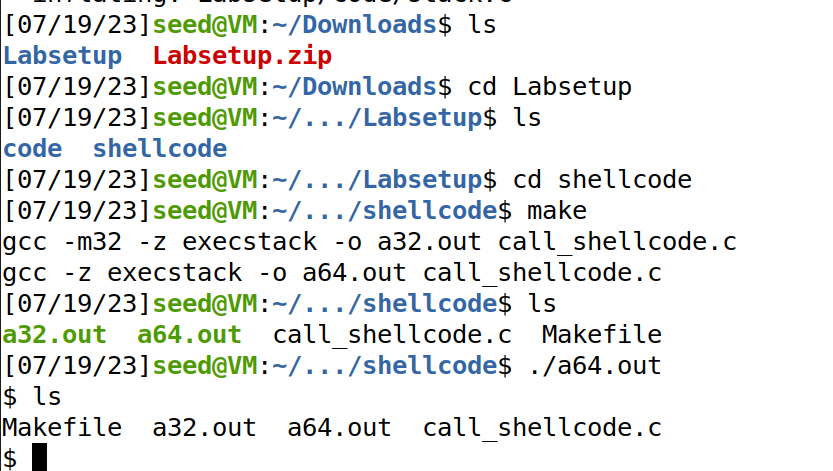
sudo sysctl -w kernel.randomize\_va\_space

Because now the /bin/sh symbolic link points to /bin/dash, and in the first experiment, we found out that dash has a security policy for set-UID programs, which drops privileges. Therefore, we need to change the symbolic link to /bin/zsh using the following command:

|  |  |
| --- | --- |
| 1 | sudo ln -sf /bin/zsh /bin/sh |

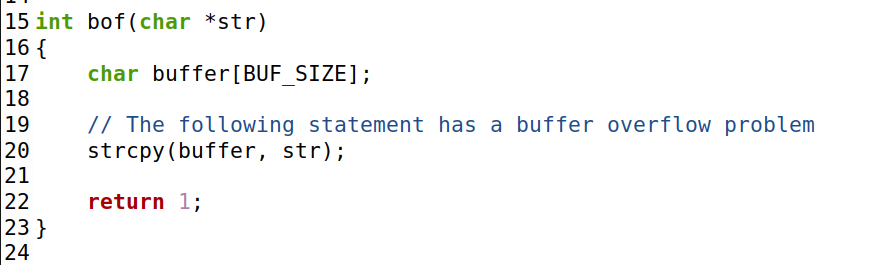
Task: Invoking the Shellcode

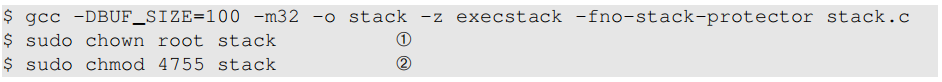
****

****

**Task 2: Understanding the Vulnerable Program**

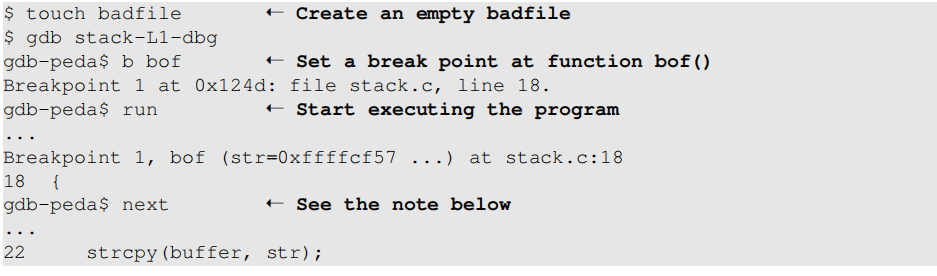
Audit from the source code reveals that the vulnerability lies in the strcpy(buffer, str) function within the bof function.

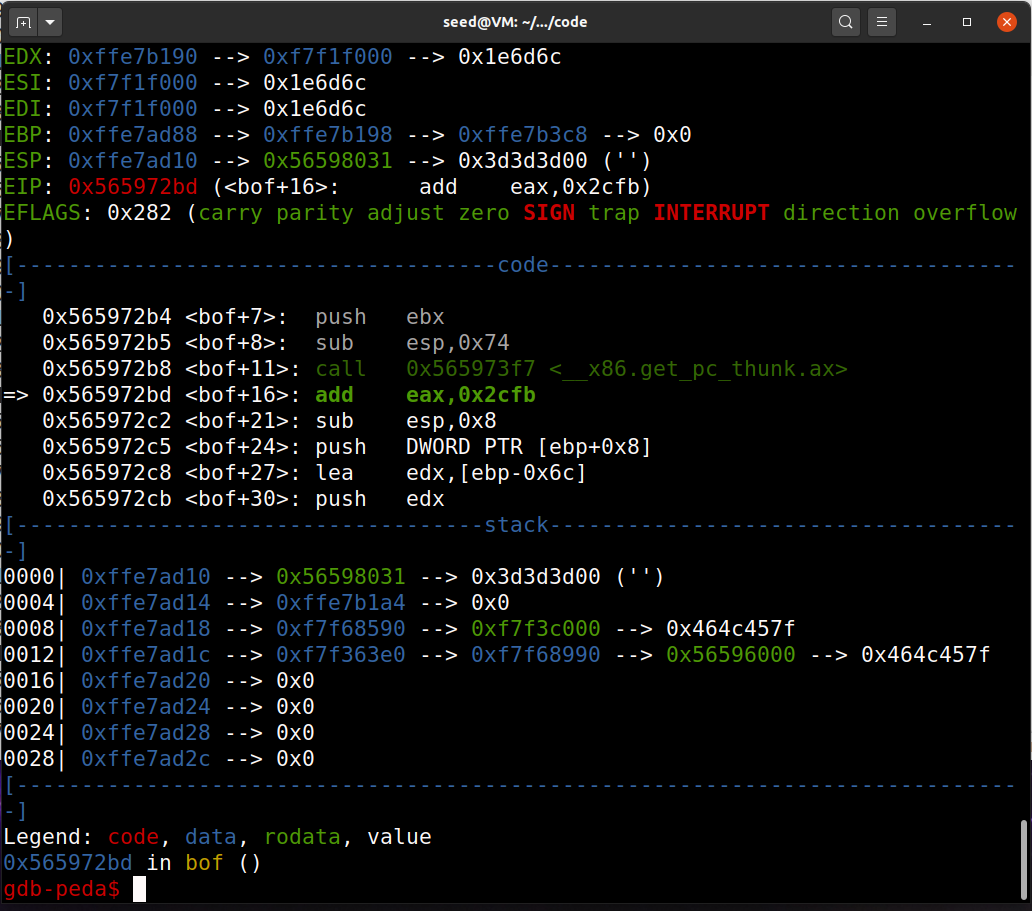


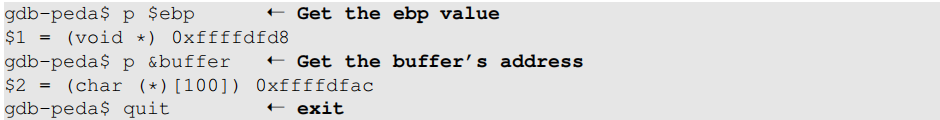
We need to fill the badfile file with the payload to exploit the buffer overflow vulnerability. The specific command-line instructions are provided in the official lab PDF file.

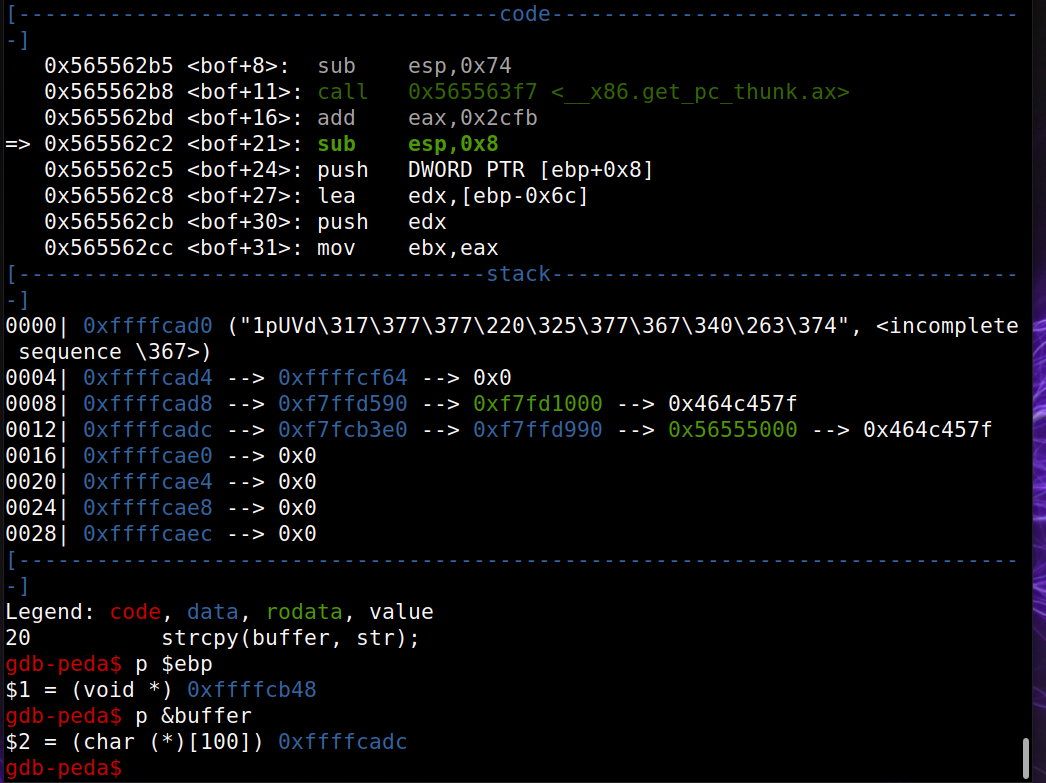
**Task 3: Launching Attack on 32-bit Program**

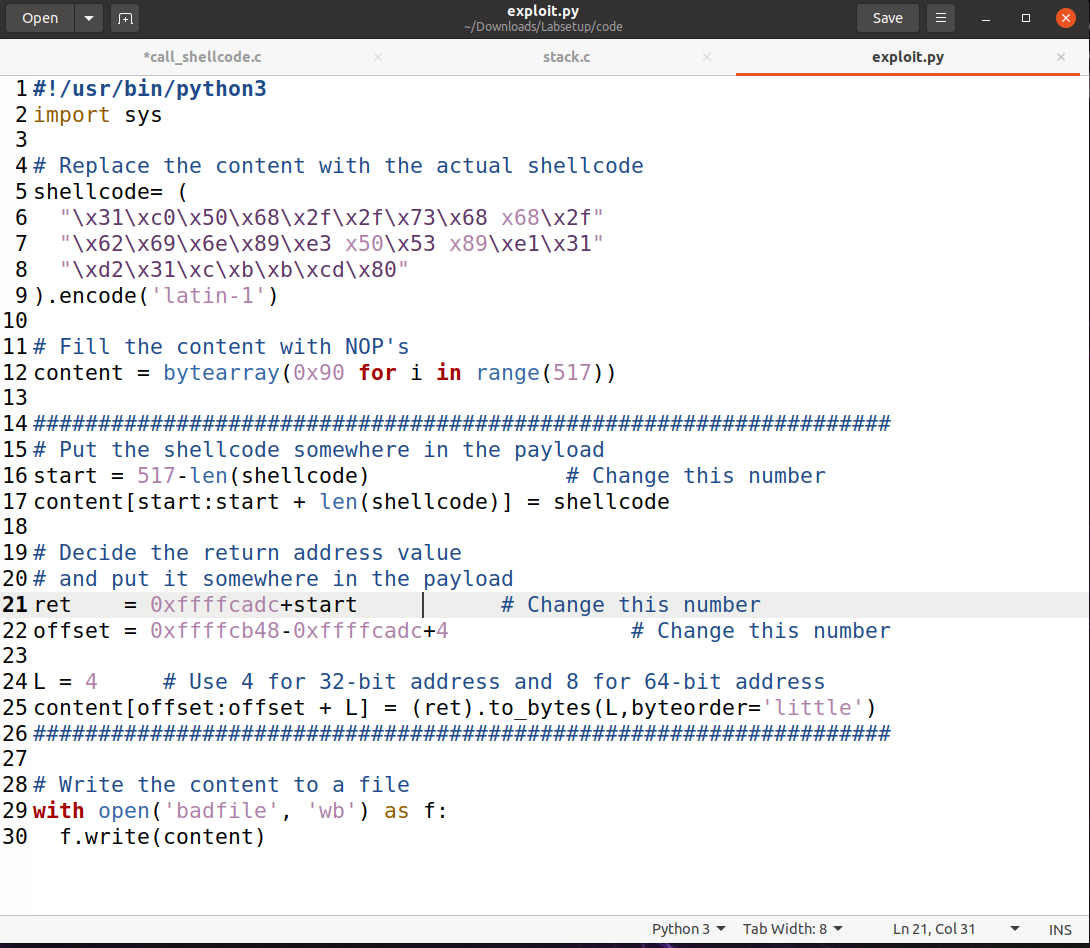
Follow the steps below to launch a buffer overflow attack:

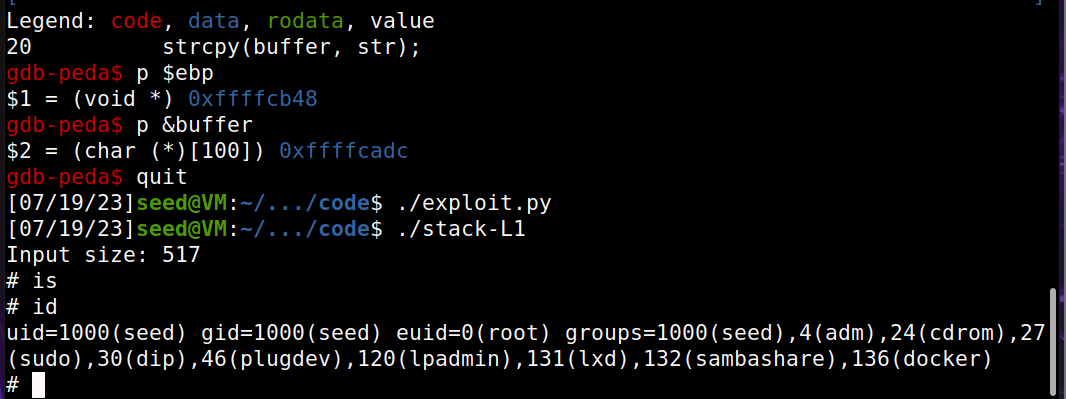








Next, modify the exploit.py file：

Generate the payload and launch the attack：

We can see that we have successfully gained root privileges!