CS6570 Secure Systems Engineering
Lab 5

Akshith Sriram CS19B005, Bhanu Shashank CS19B043

Q1. Read the code given in `team11.c` and run the exploit provided to you.

1. List all the bugs/vulnerabilities present in the program.
    - In the function `addplayer()` the instruction `gets(p)` allows the user to enter any length string as player name, but the size defined for buffer `p` is only 16 bytes (on the heap). This can be exploited to cause a heap overflow.
    - In the function `addplayer()` the instruction gets(type) also can be exploited to cause a buffer overflow on the stack. Also, `atoi()` might not work as intended for invalid inputs like '%' or '('

2. If there are vulnerabilities, then which of them can be used to overwrite the secret present in `main` with your team name? Explain the process in detail.
   The Heap Overflow vulnerability (explained in point 1 in the previous question) can be used to overwrite the `secret` message. The idea is explained as follows:
    - Add three players ("virat", "dhoni", "sehwag") in the program, which will cause three mallocs, and three chunks are allocated to the players.
    - Remove the players in reverse order, so that three chunks are added to fastbin. Removal in reverse order is important to make sure that chunk allocated to first player ("virat") is used in the next malloc. (Removal from fastbin follows LIFO)
    - Add another player, which uses the first chunk. As the player name, give a large input, which contains the address of the `secret` array at an offset of 32 bytes, so that the overflow in the heap

causes the linked list pointer of the second chunk to point to the `secret` array location.

- Add another player, so that second chunk is removed from the fastbin, and the forward pointer is now pointing to `secret` array location.
- Add another player with the name "Anonymous Hackers", so that the secret array content is replaced with the team name.

Note: ASLR should be turned off in the system, so that the address of secret array doesn't change during runtime.

3. Submit your exploit string as .exp
   The exploit string is present as CS19B043_CS19B005.exp in the submission zip.

Q2.

1. What protection is present in the binary to prevent an attacker from overwriting the secret?
The binary is protected by ASLR, making it difficult for the attackers to find addresses.

2. As an attacker, explain the process used to exploit this vulnerability by bypassing the protection and overwriting the `secret` with the secret provided to you.
In the code, there is a line to print the address of the `secret` array. Using this address, we were able to modify the content of `secret` array. The idea used is same as the idea explained in Q1.

3. Submit your code to exploit the binary hosted at the server as .py
The code is submitted as CS19B043_CS19B005.py