**Assignment 3 Report**

**Nishant Chaudhary(1001568781)**

To execute the code type:

gcc assg\_3.c -o assg\_3

Exploit:

The program have two character arrays, *first\_string* having size 20 and *second\_string* having size 10.

Now we are using *strcpy* to copy *first\_string* to *second\_string*.

**Case 1: If the size of the string entered in *first\_string* is not greater than *second\_string*:**

In this case, there will be no complications.

e.g. first\_string = “Hello”

second\_string = “Hello”

**Case 2: If the size of the string entered in *first\_string* is greater than *second\_string*:**

In this case, the *first\_string* will still be copied to *second\_string* even though it’s of a greater size. The string from *first\_string* will be copied until there is free space available right next to the *second\_string* memory space. It will stop when there’s no more free space available.

e.g. first\_string = “Hello World, How are you?”

second\_string = “Hello World, How are you?”

The size of string entered in first\_string was greater than the second\_string but it’s still copied because there was memory available.

first\_string = “Hello World, How are you? What are you doing?”

second\_string = “Hello World, How are you? What are yo”

In this example the size of string entered in first\_string was greater than the array size but it still reads it because there was memory available but for the second\_string there wasn’t enough space available to store the entire string so it just stores as much as it can.

So this way we can exploit this program because over here we are using memory space which we aren’t supposed to use. A buffer overflow, or buffer overrun, will occur because more data is put into a fixed-length buffer than the buffer can handle. The extra information, which should go somewhere, can overflow into adjacent memory space, corrupting or overwriting the data held in that space.