Larry Arndt

CS-405

January 14, 2023

Buffer Overflow

With character arrays and buffer overflow in C++ you have to ensure that too many characters are not inserted into your array. There are several ways to do this. My method here is to create a much larger array and test its size. If it is over the limit of the lower array size it would have overflowed it. I just chose a very large array size of 25000 characters, (since in C++an error is given if you try to declare a dynamic char array with unknown size) which is the size at least 50 pages of text, and highly unlikely to be overloaded. Even if it was overloaded it is a temp variable and would crash the program does not show the passwords. Another way is to use the malloc command and assign a memory size, same concept really it can also be overloaded with too much data.

In this project the array size is modified to 21 because of end characters. It does not affect the functionality. This is to accommodate the copying that takes place, where a null terminator is inserted at the end of the copy function of the strncpy\_s function and must be accounted for in an insert of 20 characters to not overload the string as the size of minus one is inserted to ensure room for the null terminator. The length of the Buffer overflow has already been determined before this point; no overflow ever occurs.

One other method would be to read one character at a time and input them into the array one at a time as you count them and throw the overflow message if a 21st first character input attempt is made. You would have to append characters to the array as you went, while you were reading from the input stream with a loop into a temp character. I wanted to do this implementation; however, the large array method is very simple to put in to place Why re-invent the wheel, this second method would not be able to be overloaded, nonetheless.

Text

Description automatically generated