Critical Thinking 5

Gregory Norris

CSC 450

Colorado State University Global Campus

Critical Thinking 5

The goal of Critical Thinking 5 is to work with manipulating text files in C++. The first goal is to accept a user input in the form of a string and append it to a file given by the instructor titled “CSC450\_CT5\_mod5.txt”. The program then copies the contents of the appended file to another file called “CSC450\_CT5\_mod5\_2.txt”. After this second file is created, the program will take the information created in “CSC450\_CT5\_mod5.txt” and save a reversed copy of the information in “CSC450\_CT5\_mod5\_reverse.txt”. The end result is the first file is appended to, the second file is overwritten with the contents of the first file, and the third file contains the reversed content of the second file.

Any time a program is accepting input from a user and-or a file, there is the risk that information which is inserted into the program can cause unwanted operations. A user can attempt to overflow the buffer, or to insert commands into the program which may be executed as code or allow random code executions within the program. The same can be said when a program accepts input from a file. In this case, there is an even bigger threat of allowing the user to write whatever they want to into the file. This increases the attack surface of the program since the user has multiple opportunities for their input to pass through the program.

My chosen means of combatting this vulnerability is in ensuring that the user input is handled as a string before being processed by the program. There are try catch statements to handle buffer overflow events, and the built-in strings class in C++ handles most of the input correctly. I used getline() to take the entire user input and copy it into a string variable, which seems like the most secure way to take the input at this point. There is no additional checks from this point, since the program then copies each character individually to perform the reversal, and the file-copy is performed by loading the file contents into a buffer stream which is written to the reversed file. The program handled all of the attack types I inserted into the console, although I suspect that it might be possible to alter any of the files from the OS to insert malicious code into this program.