Emma Wilkins

CS3923 - Assignment 2.3

Vulnerability Write Up

My original reference model performed highly vulnerable to attack for the primary reason that the reference model itself could not compile. Not only did this render the monitor ineffective in protecting the system but made it non-functional as in general. Perhaps causing a failure of the entire system. This is obviously a huge problem and is completely unacceptable for a system and my top priority was to get the reference mode compiling and furthermore operating correctly.

What I found was stopping my file from running was the improper implementation of threading.

While I recognized that threading could be the appropriate way to handle file misuse within the system, in order to maintain a working implementation I decided to remove this aspect of the code. Another reason for this is among the attack cases that would be run on my reference model, I could not find any that correctly could implement running multiple threads. I realized, for this assignment specifically, it may not be critical to implement but still is worth noting for the system as a whole if I wanted it to be implemented entirely correctly.

Another problem I found with my previous code was that I was storing the most recent executable of the file in a dictionary. I failed to include important information about the file such as if the file was closed and the length of the file. I made a boolean value for if the file was closed or not as well as an “end” for the current as well as queued information.

From further inspection of the attack cases I realized I missed a vital part of the assignment in having a field for checking the end of file to maintain a full image of what the file looks like before and after a write. This led me to introduce the pending\_end and old\_end variables to maintain a record of the expensed end position of the file after a pending writeat and the last valid execution of the file.

I also simplified the function named commit change. I wrote a single line in the close function to issue a duplicate writeat for the information in the queue, in order to commit the queued information before the queue is reset to 0.

In the writeat function I implemented the bulk of the error handling based on discussions in class and the attack cases provided by other students. I added error checks to handle the following situations: if the file is already closed, if the offset is negative (indicating an attempt to read from a position before the file starts), or if the write operation exceeds the file's end.

With these modifications, my reference model is now functional and resilient to the majority of attacks, when run on my machine it only was surpassed by 2 of the accepted attack cases.