Brandon Pinos

Professor Cappos

CS3923 / 6813: Computer Security

November 2nd, 2023

Assignment 2.3

My original reference monitor, although it performed the undo function as directed, did so inadequately with many attacks capable of bypassing it or causing it to otherwise perform unwanted behavior. I had overlooked some vulnerabilities such as EOF errors, attempting to perform a writeat operation at a negative offset, attempting to close a file when it was already closed, or trying to writeat when the file was already closed. Although these simple oversights may not seem such a big deal at first, a malicious user could abuse them in their entirety. These vulnerabilities were fixed by adding checks to track what the max possible offset was, whether the attempted offset was an invalid value, or whether the file was already opened or closed.

However, the biggest vulnerability was the lack of accounting for multithreading errors. By creating a primitive lock object, access to shared objects can be controlled. This is necessary to prevent against the corruption of data when guarding against simultaneous access to data. By utilizing the acquire and release functions of a locked object before and after each function call, I can prevent race conditions and prevent changes from occurring on multiple threads.