Assignment 2: Part 3

Enhancements to the Reference Monitor

Introduction

The security model of PART-1 presented several vulnerabilities in its functionalities. These issues stemmed from the lack of checks and balances on file operations, particularly on the sequence of write and undo operations. The vulnerabilities could allow a user to undo write operations erroneously or even to manipulate file contents in an unanticipated manner, compromising data integrity. PART-3 introduces several enhancements to address these flaws.

Vulnerabilities in PART-1

- **File State Checks:** The original code did not adequately verify whether a file was open or closed before performing operations, leading to potential inconsistencies and data loss especially during undo.
- **Error and Exception Handling:** PART-1 did not properly handle errors such as attempts to write or read past the file's end, use negative or invalid offsets, or modify closed files.
- **Locking Mechanisms:** The locking mechanism, particularly in the close API, was insufficient, raising the possibility of race conditions and deadlocks.

These gaps posed significant risks for secure file operations and required a comprehensive reevaluation.

Enhancements in PART-3

To address these shortcomings, PART-3 incorporates the following revised enhancements:

- **Enhanced State Tracking:** PART-3 tracks the open or closed state of the file rigorously. Operations are contingent on the file's state, preventing any read, write, or undo actions on a closed file.
- **Robust Error Handling:**
  - **Bounds Checking:** It implements thorough checks for offsets, ensuring they are within the valid range before proceeding with read or write operations.
  - **End-of-File Handling:** Attempts to operate beyond the file's current length raise a *SeekPastEndOfFileError*, preventing unauthorised access to the file system.
  - **Negative Offset Handling:** Negative offsets are explicitly checked, and operations with such offsets are blocked by raising a *RepyArgumentError*.
- **Improved Locking Mechanism:**
  - **Atomic Close Operations:** The locking mechanism during the *close()* method is refined to ensure atomicity, preventing any other operations from interfering during the close process.
  - **Locking Scope:** Locks are now used more judiciously, reducing the scope where they are held to minimise the chance of deadlocks.
  - **Condition-Dependent Lock Release:** The lock is released only if it was acquired by the same function call, preventing release by an unrelated operation and avoiding race conditions.
- **Immutable Closed Files:** Once a file is closed, its state is immutable. Any attempts to undo changes after the file has been closed are ignored, ensuring that closed files remain unaltered.
- **Undo Operation Restriction:** The undo functionality has been refined to reset the file to the previous state only if the file is open. This change ensures that undo operations do not introduce any inconsistency.

Additional interesting points during experimentations:

There were several test cases that failed in spite of the correct error handling; however, I realised that the order of precedence in error handling were crucial. Another vital realisation was to account for where the write call was coming from; if the close function invoked a write then a lock must be held by close and not write, whereas in case of a user’s call, a lock must be held by the writer. These two important realisations helped defend against numerous attacks enhancing the reference monitor’s robustness and reliability. After deleting the invalid test cases, I am now able to defend against all the test attacks successfully.

Conclusion

PART-3 presents a more robust and secure approach to file operations, particularly the undo functionality. By addressing the drawbacks of PART-1, PART-3 ensures that data integrity is maintained, and operations are predictable and secure. The rationale behind these changes is rooted in the principles of secure programming, which dictate that operations on files should be predictable, logged, and reversible only under correct conditions. The new security layer encapsulated in PART-3 provides a significant enhancement over its predecessor, offering a more secure and reliable reference monitor for file operations.

Makesh Srinivasan (ms15138)