In my initial code I did not keep track of the offset so while running the attack cases agains my reference monitor, my code would fail every time it faced any test cases addressing the issue. To fix that I maintain an **eof_offset** variable that tracks the current end of the file to make sure no writes go beyond it. This will ensure that subsequent writes are not writing past the EOF.

And of course added the following lines of code to make sure my security layer handles it:

```
if offset > self.eof_offset:

    raise SeekPastEndOfFileError("Trying to write beyond EOF!")
```

Also to resolve some of the errors I added the following method:

```
def commit_pending(self):
   """Commit the pending write to the file."""
   if self.pending_data is not None:
      # Update eof_offset if writing beyond current EOF
      self.eof_offset = max(self.eof_offset, self.pending_offset +
len(self.pending_data))

      self.LPfile.writeat(self.pending_data, self.pending_offset)
      self.last_written_data = self.pending_data
      self.last_written_offset = self.pending_offset
      self.pending_data = None
      self.pending_offset = None
```

With this method in place, the writeat method will save a write operation as pending and commit_pending will actually execute it. This way, the eof_offset will be updated correctly, and the checks in writeat should be effective.