Design related Issues in the Asgn2 driver

What the Driver Does

This driver works for a character device file that lets programs read data sent from dummy hardware. Only one user can open the device at a given time. The device driver takes care of moving the data from hardware -> through a buffer -> to the program that reads from `/dev/asgn2`.

Data Race Prevention

- **Locks and Semaphores:**
  The driver uses "spinlocks" to avoid problems when the hardware and the program might try to use the buffer at the same time. There's also a "semaphore" that only lets one program open the device at once, so no two programs corrupt each other's data.
- **Problems with Global Session State and Multi-User Support:**
  If more than one user was allowed, there could be trouble. The driver keeps some info (like `session_read_done`) as global variables. That means if two users come in at the same time, their sessions could get mixed up and the driver wouldn't know which session belongs to whom.
- **Cleaning Up:**
  When a program closes the device soon, the driver tries to delete leftover data, but if it misses something (because of bugs or fast changes), another user may see old data.

Misuse of Deferred Work Mechanism (tasklet)

The driver uses a tasklet to handle data processing from the device. Because tasklets run in a high-priority interrupt context, they must execute almost instantaneously to avoid blocking the system. Using a tasklet for a potentially long-running operation like processing a large file monopolizes a CPU core, starving other critical processes and causing the entire system to crash.We can instead use workqueue which is safe for long running task.

Security Concerns

- **Big Memory Use:**
  If programs open the device and don't read anything, the buffers will keep getting bigger. This might fill up kernel memory and make the system slow or crash.

- **Leftover Data:**
  If the driver doesn't clear out data properly after a user closes the device, the next program might get someone else's old bytes. This could be a problem for privacy.

- **Blocking (Denial-of-Service):**
  Because only one program can open the device at a time, a buggy program could open it and never close, blocking everyone else. The code doesn't have a timeout or force-release if someone holds on too long.

How to Make It Better

- To avoid issues with global state, we can enable multiple session management i.e session per user. But this can lead to complexity in user state management. Having a producer/subscriber model is theoratically possible but that again will introduce complexity in terms of memory usage.
- The cleaning up can be improved by having stronger Invariants/checks and a fall back mechanism to clean the leftovers. We can log and monitor such failures to cleanup.
- Put a limit on buffer size so memory can't run out and add timeout or force close if someone holds the device for too long.