####The Livelock paper has a hack to handle the problem of dropping packets on the “screend” queue. What limit does this solution have for multiple applications getting network data?

Since “screend” queue runs in user mode, the kernel must queue packets for delivery to “screend”. When the system is overloaded, this queue fills up and packets are dropped. “screend” never gets a chance to run to drain this queue, because the system devotes its cycles to handling input packets. To address this problem, the paper detects when the “screening” queue becomes full and inhibit further input processing (and input interrupts) until more queue space is available.

For multiple applications getting network data, the feedback policies for these queues would be more complex. It might be difficult to determine if input processing load was actually preventing progress at these queues. It would be hard to tell if there are too many packets arriving.

####In Livelock: Figure 6–3: Why does “Polling (no quota)” drop almost immediately to zero rather than gradually decreasing similar to “unmodified” Be very concrete.

Because there is no quota, packets are discarded for lack of space on the output queue. However, for “unmodified” kernel, as the interrupts increase, the CPU can become monopolized by the interrupt handler, and the response to the user-level process can be delayed. The more input interrupts, the more delay time. This is why the throughput of “unmodified” decreases gradually.