Too many worklogs! Mystery Jira slowdowns due to index contention

I've spent the last few weeks dealing with a slow Jira, caused by a single issue having 59k worklogs on it. See my writeup and worklog-moving PL/pgSQL script on Github for details on the fix.

The interesting part for me what discovering just how single threaded and bottlenecked Jira actually is, due to certain plugins:

- While many threads can read a Lucene index concurrently, *writing* to a Lucene index is done by only one thread.
- Calling reIndex() for an issue reindexes its worklogs too (JRASERVER-72469).
- There are a number of plugins that call reIndex() on issues for no good reason:
    - The PagerDuty plugin has a listener that reindexes the issue before anything else.
    - The CSAT plugin (used to) reindex the issue before deciding whether to show its panel.

Perhaps these plugin authors don't trust Jira to keep its indexes up-to-date, and toss in a reIndex() to be sure.

The net result is reIndex() calls into everyday operations like viewing an issue. These operations all queue up for the global Lucene write lock, blocking all reads too. **Jira becomes effectively single-threaded,** but you don't notice this until someone views the "Meetings" issue, causing a giant 59k-worklog hairball to block the Lucene write index.

If you've ever had users complaining of Jira slowness, but found the server underutilized (1 core active, minimal disk use), and been unable to replicate the slowness: it's probably index lock contention. I suggest getting familiar with the wonderful new JSON index statistics emitted in Jira 8.13.x+.