Lustre + Linux

The tale of two trees
Upstream client progress since last LUG

- From rage to acceptance
  - Merged in 2013. Left to decay for two years.
  - Almost deleted from upstream twice
  - Now patches merged in a few days

- Massive checkpatch cleanups have been done

- Moved from procfs to sysfs

- Updated tools to allow testing of upstream client

- Merged some of the OpenSFS clean ups for libcfs

- Synced upstream LNet layer to current master
  - Made LNet stand alone for Linux kernel
  - Can use default Linux kernel 4.6+ for lustre routers
  - Interest in moving LNet out of staging first.
Impact of a functional Upstream client

• Preparing automated testing of upstream client
  – Working out test failures
    1) Don’t test latest functionality (LU-7746)
    2) Skip real test fails for now
  (LU-4011)
• Submitting patches for LNet/libcfs to upstream as well as OpenSFS/Intel branch
  – Upstream one change rule.
    • Mostly impact new features
  – Consider kernel coding style
    • OpenSFS guide lines mostly the same.
    • Run kernel-source/scripts/checkpatch yourpatch.diff
Upstream coding style difference

• 80+ character line length more relaxed
  – Readability more important
  – Don’t break strings
  – Headers style doesn’t matter?

• Hate of spacing.
  – No “int rc;”
  – No “var = value1;”

• No generic naming, i.e goto failed2; …. failed2:

• No lots of returns in functions. Use goto instead.

• No return -1 or positive numbers

• Handle error handling not success handling

• http://wiki.lustre.org/Upstream_contributing
Upstream style versus OpenSFS style

• Resistance to some kernel styles
  – “if (rc != 0” or “if (rc == 0)”
  – “if (ptr != NULL)” or “if (ptr == NULL)”
  – “if ((a & b) == 0)” is bad style and should be changed

• Discuss at developers day
  – Use all kernel style makes porting easy
  – If not using all kernel style need script to push patches upstream
Upstream future work

• Sync libcfs for OpenSFS/upstream
• Push lustre client code changes to upstream
• Move to standard debugging
  – Tracepoint support
  – Memory leak detection
    • Enable kernel debugging options (kmemleaks)
    • KEDR framework
Syncing master with upstream client

• Participants:
  – Oleg Drokin, John Hammond, and Dmitry Eremin from Intel
  – Frank Zago, Ben Evans from Cray
  – James Simmons from ORNL

• Goals:
  – Remove technical debt (LU-7917)
  – Meet kernel coding styles (LU-6142)
  – Remove abstractions (LU-6245 / LU-6401)
  – Support newer kernels (LU-6215)
  – Back port fixes for bugs from upstream (LU-4423)
Master’s progress since last LUG

• Support up to 4.4.6 kernels (LU-6215)
• Tool support for upstream client (LU-5030)
• Kernel coding styles enforced (LU-6142)
• Flow of upstream fixes to master (LU-4423)
• Continue cleanup of lustre headers (LU-6401)
• Work on libcfs continues (LU-6245)
  – libcfs module free of user land code
  – libcfs headers are uapi compliant
Preparing master’s server code for upstream

• All code cleanup for clients are applied to servers

• LU-20 : Goal of no more patching the server side
  – Patching only needed for ldiskfs support
  – LU-3406 : raid5 mmp unplug patch
  – LU-684 : Use dm flakey to test fail over
  – Drop the rest of the patches since they are upstream
  – Only need to patch latest kernel with LU-3406 for testing. Hopefully proper upstream solution will be done

• LU-6220 : Push most ldiskfs patches upstream

• LU-7311 : Server, ldiskfs support for 3.18 kernels

• LU-3953 : Working lustre packaging system
What needs to be done for OpenSFS/Intel branch?

• Checkpatch audit

• Back port sysfs support to OpenSFS branch.

• LU-6245 : libcfs train wreck
  – No more using libcfs.h as master kernel header
  – Abstractions still left

• LU-6401 : lustre header train wreck
  – User land and kernel header entanglement
  – Use of internal kernel headers in user land

• Rework build system
  – Build only lustre utilities and use upstream kernel headers
Conclusion

• Very successful year

• Work left for master
  – Finish off libcfs syncing during 2.9 development cycle
  – Lustre user land / kernel header cleanup
  – Kernel code style updates will be big update

• Continue to bring upstream client up to date

• We can’t forget support for latest kernels

• Goal by next LUG is have sync all the upstream client code with OpenSFS branch