**OFI Data Storage / Data Access Subteam Weekly telecom – 02/23/2016**

**DS/DA Shared Documents:** <http://downloads.openfabrics.org/WorkGroups/ofiwg/>

**Agenda**

* roll call, agenda bashing
* review Frank’s slide
* discuss Doug’s proposed Lustre stack

**Lustre – Doug Oucharek’s slides on the Lustre stack**

* Current LNET stack, varies a little bit depending on the particular path
* o2iblnd talks directly to OFED, which then calls at least one, or more providers
* Doug’s drawing shows three ‘providers’ beneath OFED, and each one driving a ‘vendor hardware driver layer beneath that. That’s really the kfabric architecture. The concept of a ‘provider’ doesn’t really exist under OFED.
* Currently, OFED contains A LOT of code (~6,000 lines!), and its pretty fragile.
* Bernard made an offline observation that GPFS is primarily user space.
* Basic takeaway – the network interface today is too complex, would be really nice if the network exported a cleaner, simpler abstract interface. Struggling with queues, completions, etc. A cleaner interface, even at the cost of more complex memory registration is worthwhile.
* Need to add some of this thinking to the Kernel Maintainer slide deck, since Lustre is a kernel application.
* Some of the issues Doug raised earlier (assigning threads to cores, etc), are really more platform issues, as opposed to communications issues.
* Suppose you are using the network for both MPI and Lustre. Without a common point that handles the runtime issues (use of cores, use of NUMA, use of threads, etc), how do the two applications know how these issues are being handled commonly? Today, for Lustre, these issues are handled in the LND layer (and above). Wouldn’t it be nice if there was a traffic cop above that is able to handle these issues.
* Looking at Doug’s second diagram – kfabric serves as the ‘LND’ layer itself, since it hides the details of the underlying network.
* Current drawing shows all providers behind kfabrics (verbs provider, gni provider, sock provider…). This also centralizes the platform management issues (thread management, etc).
* This drawing implies that LNET is (re-)written to the kfabric API. Probably an intermediate step would be a thin “LND shim” layer. Doug thinks the Lustre community (and the kernel maintainers) would find this appealing.
* The drawing also shows sockets underneath kfabrics, but so far we have been positioning kfabric as parallel to and complementary to sockets, not supplanting it.
* Stan raises a question that the LNET layer seems to imply a desire to communicate with the network via e.g. reliable data gram. But that might be problematic if RDGM isn’t well-supported by the provider.
* Would be nice if credits could be handled by kfabrics, instead of LND. May not be necessary to do this in the kfabrics layer, but could be handled in the provider (reliability, end-to-end flow control), etc.
* Similarly, multi-rail should be handled by the provider.
* Because Lustre is RPC-based, ordering is not a requirement. (But small messages are very, very bad).
* Takeaway: today, LND Is owned and maintained by the Lustre community; in a kfabric world, much of the burden is shouldered by the provider vendors.
* Greg Cage is the current kernel maintainer for Lustre. Should we include him?
* **Need to weave this story into the Kernel Maintainer slide deck.**

**Kernel maintainer slide deck – next steps**

* Are we ready to expose this to the maintainers?
* Stan volunteers to show this privately to a maintainer with whom he is familiar. Just to give us quick feedback and prevent future disasters.
* AR – Paul to publish the current version.
* AR – Frank to propose a possible addition to cover the NVM for HA case.

**Where to next?**

* Paul believes that there are still outstanding issues related to NVM that may be best illuminated by looking at it from a user mode or use case perspective. For example, look at the use of NVM in a PGAS or SHMEM environment where processes are accessing shared, distributed NVM.

**Webex Recording:**

**Next regular telecom:**

Next meeting: Tuesday, 3/1/16

8am-9am Pacific daylight time

**NOTE:** We have switched over to using Webex (courtesy of Cisco). The URL for joining meetings is:

[Join WebEx meeting](https://cisco.webex.com/ciscosales/j.php?MTID=m221d8a20185d84b30daa0096aca0f182)

**Join by phone**

+1-866-432-9903 Call-in toll-free number (US/Canada)

+1-408-525-6800 Call-in toll number (US/Canada)

Access code: 201 212 241