**OFI Data Storage / Data Access Subteam Weekly telecom – 01/24/2017**

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

**Agenda**

* roll call, agenda bashing
* OFA Workshop
* Lustre over kfabrics

**OFA Workshop**

* No real enthusiasm for a topic this year.

**Lustre over kfabric – Doug Oucharek**

* Can kfabric serve as a suitable network adaptation layer for Lustre?
* In some sense, existing LNDs serve a similar purpose as kfabric. If, by using kfabric, would we avoid the requirement to create a new LND for each new fabric?
* Slide 2 – complete set of function calls from LNET down to an LND (today). The thing to notice about this slide is that there are a very small number of calls, and they are quite abstract.
* Lnd\_send, lnd\_recv.
* Lnd\_eager\_recv only used by gnilnd today.
* Lnd\_notify, lnd\_accept, only used by socklnd today.
* For the o2iblnd, only six calls get registered today.
* Slide 3 – upward call backs. A similarly high level of abstraction.
* Lnet\_cpt\_...
* Slide 4, 5 - Kfabric API calls – test code
* Quite a few functions dedicated to managing objects and endpoints
* Existing LNET/LND i/f is fairly abstract, with much of the complexity buried in the LND.
* Areas for improving kfabric:
  + CPU/NUMA affinity – increasingly important for Lustre in an era of many cores/sockets.
  + Memory management becoming increasingly important, and one of areas that is the source of many errors.
  + Automatic connection…
* Slide 7 – three options:
  + ‘backport’ LND functions up into the LNET layer.
  + Maintain a simplified kfabric LND,
  + Add a simplified API layer to kfabric
* Curious as to whether a simplified kfabric API would be beneficial to other users such as NVMe.
* Raises the question as to whether kfabric, which to date follows the libfabric model, should be a simplified API. Libfabric today offers a fairly large number of any given type of operation (e.g. fi\_msg…) in an attempt to avoid branching in the code based on options.
* Given our commitment to being application-centric. Perhaps the existing kfabric API should be re-examined to see if it should be retro-fitted to be a better fit for the consumer (e.g. LNET).
* Willing to migrate the memory registration stuff upward to LNET.
* Less willing to expose the wide variety of calls…would prefer a simplified call into kfabrics for sends, completions, etc.
* A key point emerged in the discussion: perhaps what kfabrics really needs is a higher level, more abstract interface that does not expose queueing the way that e.g. a VERBS interface does.
* The question is whether we are on the right path with kfabric and simply need to augment it with a companion set of calls that are more abstract, or should we begin with a clean sheet of paper on which to gather requirements from, e.g. LNET and NVMe?

**Webex Recording:**

**Next regular telecom:**

Next meeting: Tuesday, 02/07/17

8am-9am Pacific daylight time

**Logistics:**

See the OFA’s central calendar (<https://openfabrics.org/index.php/ofa-calendar.html>) for current meeting logistics.