**OFI Data Storage / Data Access Subteam Weekly telecom – 06/07/2016**

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

**Agenda**

* roll call, agenda bashing

Kernel mode stuff

* Brief update on the status of the GitHub repo, and progress toward merging the two kfabric proposals.

Persistent memory stuff

* Update from Chet Douglas on Intel proposal for extending RDMA support for persistent memory access.

**Progress on merging kfabric frameworks on Github**

* NetApp posted its version of the framework, including the verbs provider. This was posted at Paul’s request in an attempt to leverage the spirit of open source to accelerate the merge of the frameworks. Some viewed this as a violation of open source process.
* There are presently two implementations of the framework – one from Intel and one from NetApp. The NetApp version includes a verbs provider and some test cases. It is the intention to find a way to merge these into one implementation.
* The question becomes, how do we proceed in merging them?
* Here are the forks in the road:
* 1. Toss existing Intel work and fix the NetApp version so it conforms to kernel coding style,
* 2. Toss the NetApp version and update the Intel version, (including adding a provider?),
* 3. Find a way to leverage the work that has been done on the two of them for the benefit of all.
* NetApp’s stated intention at the time they posted their version was to drive toward a converged version. Rather than try to converge the two in private, it felt like the best approach was to attempt to do this in public, hence my request to use the GitHub repo to expose both implementations. The clear intent is to end up with a single implementation.
* Would like to see the NetApp implementation cleaned up w.r.t. Check Patch
* The existing NetApp version should be seen as a ‘preview’, not as hardened, production ready code.
* Stan, Frank and Chen plan to continue discussing this, preferably using the OFIWG mailing list.

**Intel Proposal for Extending RDMA support for PM access – Chet Douglas**

**See the document “RDMA Extensions-Proposed libfabric API.DOCX**

* This is a follow-up to a presentation Chet gave to this group probably about a year ago on Intel’s thinking about using software methods to make persistent writes durable.
* Intel is looking at a number of s/w and h/w changes for its next platform. The previous mechanisms required a number of extra software steps to make it work.
* This is a libfabric proposal for a ring 3 API, and what extensions to it might look like, driven by an internal Intel hardware assessment of data paths and so on.
* One key objective: reduce h/w complexity. The proposal recognizes that the h/w design pipeline can be long.
* Chet will publish a new version of the doc following this meeting.
* Q: are these enhancements specific to OFI, or can they be extended to verbs? A: Should mirror nicely into verbs.
* Proposing changes to three libfabric APIs: fi\_getinfo, fi\_mr\_reg, fi\_writemsg
* **fi\_getinfo** – changes to the info flags to indicate the existence of persistent memory, i.e. this device supports accesses on something other than a block granularity.
* **fi\_mr\_reg** – proposing three new flags: fi\_pmem to indicate that the memory region being registered is persistent. fi\_uncached is a hint to indicate to the provider that this region should not be cached. Helps the NIC decide how to handle caching for this memory region. This hint applies to the target side; it is an open question as to whether it should be made available on the initiator side. fi\_non\_standard\_memory\_device, mainly for use if the PM is not attached to a memory bus. Allows kernel driver to supply whatever resources the NIC may need. Q: Current libfabric doesn’t distinguish between L\_Key and R\_Key, today you get a descriptor (equiv to an L\_Key) and a key (equivalent to an R\_Key). The group seemed to agree that this flag should not be exposed across the API to the consumer, but is useful to the provider implementation.
* **fi\_writemsg** – asking for new op codes for fi\_write\_commit, and fi\_write\_commit\_immediate. For the moment, include these as flags to the existing fi\_writemsg API. The provider will probably use these flags to create a new op code. New flags: FI\_COMMIT basically gives the completion semantics of an RDMA READ, i.e., you get a completion when data in scope has reached the global visibility point.
* **Ran out of time, to be continued at the next meeting**

NOTE: an updated version of Chet’s document has been posted to the OFI documents area:

**Next Agenda – 6/21/16**

* Continue walking through the Intel proposal for changes to the RDMA to support PM

**Webex Recording:**

|  |  |
| --- | --- |
| [**Play recording**](https://cisco.webex.com/ciscosales/ldr.php?RCID=0bc189796990ac787956a91fd28dab38) (56 min) | |
| Recording password: CmDPwHg7 |  |

**Next regular telecom:**

Next meeting: Tuesday, 6/21/16

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