**OFI Data Storage / Data Access Subteam Weekly telecom – 08/04/2015**

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

**Agenda**

* roll call, agenda bashing
* SC15 BoF
* kfabric intro slidedeck – Again!
* user mode I/O
* NVM usage models
* New API interface - getdevice

**Proposed BoF for SC15**

- Do we agree that there should be a BoF at all?

- If so, are we agreed that it should be distinct from the libfabric BoF (if one exists)?

- Christoph will be at SC15, Stan is a ‘no’, Paul is possible, NetApp will research.

- Everyone agrees to provide feedback on the mailing list on the proposed abstract.

**New interface – ‘getdevice’**

- need a Linux device structure. ‘int kfi\_get\_device(struct fid\_ep \*ep, struct device \*\*dev)’

- Bernard – would prefer a ‘put device’ instead of a ‘get device’. (put device ensures that the device doesn’t disappear unexpectedly. Essentially it defers cleanup of the device until the last user de-references it.)

**NVM usage models**

- Chet Douglas (Intel) may be a good candidate to do this. Paul will reach out to Chet offline. Chet works in the NVM programming model group in SNIA.

- Could include things like user level filesystems. Byte addressable memory is another candidate.

- Paul to connect Chet and Bernard to discuss further.

**User mode I/O**

- w.r.t. libfabrics, the persistency model for user mode I/O may be a clear differentiator.

- User mode file I/O – e.g. Ceph.

- The going in assumption is that we begin with libfabric and look to see if it needs to be enhanced. A key outcome from this group should be recommendations on expanding/enhancing libfabric.

- libfabric is I/O for IPC; we should look at it from a storage I/O (block and file I/O) perspective.

- Start with Key/Value.

- As part of the ‘introduction’ slide deck, NetApp had included a slide ‘Why kfi for NVM’:

- both use cases shown at present are for storage, which makes sense because the current slide deck is about kfi.

- More generally, we (DS/DA) need to cover both user mode (Ceph, byte addressable memory) and kernel model (block, file I/O).

- Doug O. raised the question of how much of RDMA should be exposed to the user.

- For example, libfabric today works in terms of a key and a virtual address. For storage purposes do we need a key and an offset instead? This is one example of an area where libfabric may need to be extended to accommodate storage operations.

**slide deck – kfabric-framework\_2015\_0714.pptx**

- We did not get to this topic today (except a brief discussion of the ‘Why kfi for NVM’ slide).

- Reminders:

- need to fill in the GitHub Repo Directory Structure slide - Frank

- suggestion from last week was to recast Slide 7 (Why kfi for NVM) into ‘block storage’ vs ‘memory access’.

- Slide 9 – needs a little wordsmithing on the last paragraph, last sentence.

- Do we want to assume this goes ‘in-kernel’? Does kfi belong to the network subsystem, or does it belong as part of drivers?

**Agenda for next meeting**

- Complete the slide deck

- Develop use cases – kernel vs user mode, I/O vs memory accesses

**Webex Recording:** [**Play recording**](https://cisco.webex.com/ciscosales/ldr.php?RCID=12668a6e4c1b6fbc566d8c85a24334dc)

**Next regular telecom:**

Next meeting: Tuesday, 08/11/15

8am-9am Pacific daylight time

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

<https://cisco.webex.com/ciscosales/j.php?MTID=m68f7fe26d65ee019c5870bc424875838>

**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: 205 894 276