**Agenda**

* Review/approve prior meeting minutes
* “Interest groups” update
* Initial feedback on MPI requirements – review past presentations

**OFWG Download Site:** [www.openfabrics.org](http://www.openfabrics.org) 🡪OFED/OFA Resources 🡪 OpenFramework WG

**Review/approve prior meeting minutes**

Catching up on meeting minutes. Minutes from the following meetings are approved by acclamation.

10/3/13, 10/22/13, 11/4/13, 11/12/13, 11/26/13, 12/3/13, 12/10/13, 12/17/13, 01/07/14

All are available for download on the OFA website https://www.openfabrics.org/downloads/OFWG/

**Interest Groups**

Four “interest groups” are being formed based on the taxonomy developed during the 12/17/13 meeting. These are: Distributed Computing, Storage and Data Access, Data Analysis, IP and Legacy Applications. The Distributed Computing interest group is off to a good start with Jeff Squyres leading the MPI portion and Howard Pritchard leading the (P)GAS portion. The Storage and Data Access interest group is in the formative stages. Yet to be addressed are the Data Analysis and IP/Legacy Applications groups.

**MPI Update – Jeff Squyres Cisco (see slide deck libfabric-old-mpi-presentations-2014-01-14.pptx)**

Interested parties should reach out to Jeff. Call has been put out to the MPI community for feedback, with a meeting planned for tomorrow to gather and coalesce inputs from an informal community (i.e. not the MPI Forum), which should generate some new feedback. Today’s review is somewhat old and possibly dated.

Today’s slides are an unfiltered recap from Sonoma 2009 and 2010. See slide deck titled

Sonoma 2009

* Feedback from major commercial verbs-based MPI implementations, how to make verbs better.
* Feedback from Open MPI, HP (no longer exists) Intel, Platform (also no longer exists).
* Eight requests were made, seven of them were made by two or more requesters.
* List of eight items.
* 1. Memory registration – MPI doesn’t control the buffer; it is handed to it by the user. This creates a need for an awkward caching mechanism. In verbs, MR is synchronous – can we do something asynchronously, or avoid memory registration completely, or a bulletproof way to notify?
* 2. Fork support inadequate – child should be able to access memory registered by a parent, device file descriptors should be set to “close on exec”. Child processes don’t do network accesses.
* 3. Connection setup scalability – all-to-all connections using RC doesn’t scale; SM can’t handle N\*\*2 path record lookups. All-to-all is not super important in MPI, but important nevertheless the same. (Note that the current framework proposal is that CM is integrated into the framework as a basic service provided by the framework.) Should think of this as RC connections between all MPI processes. Note that there is a distinction between “connection management” and communication, i.e., for all-to-all, a reliable datagram service (which doesn’t necessarily have all-to-all connections) could be useful.
* 4. Relaxed Ordering API support – only asked for by one company to support relaxed ordering over PCI. May or may not still be useful.
* 5. Stack / API portability – windows API very different from Linux. Probably not relevant to this forum, since we are focused on Linux. Is there a requirement for interoperability between platforms, e.g. AIX or other POSIX systems?
* 6. Reliable connectionless (different from RC connections at scale). XRC was there, but is sort of complex to use. Datagram support in 2009 was not great. Would really like something that is RD-like, i.e. RC-like performance, but without the scalability issues.
* 7. Send/Receive Registered Memory Utilization – current architecture requires pre-allocating memory, but the whole thing might never get used. This is opposed to other implementations that use “slabs” (a “slab” of memory is posted for receives).
* 8. CM is too complex – MPI requires a progress thread for incoming connections to avoid timeouts. One possibility is to push this down into the provider layer, provided that the cycle-stealing needed to do this doesn’t cause jitter. Q: isn’t this a problem mainly during connection establishment? A: yes, but MPI causes lazy connection creations.

 2010 Sonoma Panel

**Next meeting**

* Complete the re-cap of past thinking on MPI over verbs.
* Report on current feedback from the MPI community.
* Begin discussing Storage and Data Access (time permitting)

Logistics

Tuesday, 1/21/14

9am-10am Pacific time

<<https://meet.intel.com/sean.hefty/D14C7J10>>
<https://meet.intel.com/sean.hefty/D14C7J10>

Join by Phone
1-888-875-9370
+1(916)356-2663 (or your local bridge access #) Choose bridge 5.
Find a local number<<https://dial.intel.com>>

Conference ID: 746347966