**Agenda**

|  |  |  |  |
| --- | --- | --- | --- |
| Wednesday Morning – 8/20 | | | |
| **Start** | **Duration** | **Topic** | **Notes** |
| 8:00 | :15 | Gathering |  |
| 8:15 | :45 | Test framework discussion | LANL – Howard Pritchard |
| 9:00 | :90 | Management and addressing assumptions | Sean |
| 10:30 | :90 | Models for implementation |  |
| 12:00 | :60 | lunch |  |
|  |  |  |  |

**Test Framework Discussion “Fabtests” – Howard Pritchard**

Agenda: Current state of fabtests, test suites for similar RDMA network protocols, HPC-style job launcher options, content ideas for fabtests

* Currently two tests: unit/provinfo.c, simple/pingpong.c
* OFED 3.1.2 tarball
  + Two source RPMs had tests in them: perftest-2.2-0.17, qperf-0.4.9
  + OFED unit testing is done by vendors
* PAMI – lots for collectives, p2p, internal functions, etc..
* Portals4
* GNI (Cray)
* Job launcher
  + Libfabrics doesn’t require a job launcher
* Looked at Hydra, ORTE
  + Hydra pretty straightforward
  + ORTE would require a wrapper to make it usable.
* Best options: Hydra, PMI
* Why use PMI? Why not just use MPI?
* May not want \*anything\* in the path to obscure where the problems are.
* Outcome: put fabtest somewhere (Github?) where the code can be branched.

**Management Interfaces and Assumptions – Sean**

* Local Mgmt (ibacm) uses a mgmt plug in to resolve addresses to L2 addresses
* Planning to allow a FI Provider to share memory with the Mgmt plug-in
* The MGMT plug-in runs in user space, and is specific to a given provider; it is an optional feature to offload mgmt workload from the provider.

**Requirements for 1.0 Release**

* What is release 1.0?
  + Man pages for the APIs?
  + Man pages + code for the APIs?
  + **Libfabrics including at least one provider?**
* Release 1.0 needs to be working
  + Include a generic socket provider (bonus points if it runs on MAC OS)
  + Librdmacm/libibverbs as a first provider?
  + What is the base set of functionality required?
  + Objective is to support application development, not necessarily to produce final, optimized applications and provider
  + What is the minimum test suite required?
  + Distinguish between compliance testing (does the API do what the man page said it would do), and functional testing (does it meet the needs of the user?)
* Acceptance for any given API
  + **Must include both a reference provider implementation and a test to validate it, and a comprehensive Man page to describe it.**
* Documenting an API
  + **Man pages should be comprehensive, and should include the author, the release number (major.minor) and date of the libfabric tarball, and the release number (major.minor) and date at the time of the last semantic change to the man page.**
  + **Man pages, where necessary, should reflect both the user and provider usage of the API**
  + Implementation examples may not be the same as test suites

**Minimum Required Functionality constituting Rel 1.0**

* **In libfabrics core:**
  + **Code is complete for all defined endpoint types and operations**
  + **The list of endpoints to be included is to be decided in upcoming meetings**
  + **There is at least one provider for each endpoint type**

**Atomic CM Msg RMA Tagged Trigger**

**11 EP-Dgrm [\*] [F] [1] [\*] [\*] [\*]**

**12 EP-Msg [p] [1] [1] [1] [?] [?]**

**13 EP-Pkt [\*] [\*] [F] [\*] [\*] [\*]**

**14 EP-Raw [\*] [\*] [F] [\*] [\*] [\*]**

**15 EP-RDM [1] [F] [1] [1] [1] [F]**

F- future

p-partial implementation

\* - n/a

1 – required for rel 1.0

? – not clear yet

Extensible Functionality – interface versioning

* How to add new function calls?
* Provider versioning
  + Currently, the provider specifies a libfabrics version number, if the version number is wrong, just don’t use it.
* Extending Interface structures
  + Size w/ query methods
* Extending data structures
  + Currently using a mask, is there a better way?
  + Fi\_get info(version,…) which will indicate the version of all structures throughout.

**GIT discussion**

* Git has now moved from openfabrics.org to GIThub
  + <https://github.com/ofiwg/libfabric>
  + Go see Doug Ledford’s ‘best practices’ on the GitHub.com/ofiwg/libfabric/wiki
* Two git repos:
  + ofiwg/libfabric – will have a small number of those qualified to write to this repo.
  + Ofiwg/fabtests