**OFI WG Bi-Weekly telecom – 07/14/2015**

**Agenda:**

* Roll call, agenda bashing
* fabtests for different versions of libfabric
* completion flags #1142
* request to OFA MWG
* other topics

**Fabtests for different version of libfabric - Sean**

Should fabtests simply remain in sync with the latest version of libfabric?

+: Reduces maintenance costs.

-: Testing that libfabric is backwards compatible requires using older versions of fabtests.

Should the latest version of fabtests support all previous versions of libfabric?

+: Makes it easier to test for backwards compatibility

-: Greatly increases maintenance costs.

+: Provides an example of how an app can support multiple libfabric versions

Maintaining fabtests becomes more difficult over time.

No strong consensus either way…probably lean slightly toward keeping fabtests and libfabric in sync

**Completion Flags – Sean GitHub issue #1142**

When a data operation completes, we can report information about the type of request that it was for. E.g. was it a transmit or receive operation? This can be extended, for example, was the transmit a send or RMA write? Did it have remote CQ data (immediate data)? Etc. up to capture the full amount of information about the request. The question is what amount of information are apps actually needing?

I've currently defined completion flags which can map well to something like an opcode. The more this can be relaxed, the more optimization possibilities there are.

Basic question: How much information should be presented to the consumer via completions? Providing added information may cause branches/extra time to gather and present.

One concern – should there be some mechanism that reflects to the software (consumer) that providing additional information is not free?

Consumers (e.g. UPC compiler writers) don’t want to have to include multiple ifdefs, one per provider. One the other hand, we do want some sort of mechanism to reflect to the consumer that this information is not free by forcing it to explicitly request certain completion fields.

Having raised awareness of the issue, probably appropriate to take this back to the GitHub issue.

Open question – are all providers required to support all completion types?

If not, how do you allow an application to learn about unsupported formats at fi\_get\_info time, instead of at create\_completion time?

**FI\_MSG\_PREFIX issue – Ben Turrubiates**

Is this issue closed?

**-** usNIC is the only user of this today

- weird inconsistency between send and receive

- want to resolve this for this release since it is currently being used and want to limit the damage of backward compatibility

- suggestions: add a new mode bit, or use the (planned) libfabric versioning scheme to signify that after a certain version the behavior of FI\_MSG\_PREFIX can be expected to behave differently.

- Currently, two pull requests are outstanding which contain the two different proposals.

- Deprecate (rename) FI\_MSG\_PREFIX

- Sean points out that as of today there is no requirement that an FI\_INFO structure be allocated via the API.

- Consider adding the (long planned) version field to fi\_info

- Not clear today what the version field of FI\_INFO means. In any case this may or may not be part of resolving this particular issue.

- Ancilliary question: What does it mean if the libfabric package release number is different from the API (fi\_version) version number? Can we get into a situation where the package is 1.1 and the API version number is 2.0

- Need a plan for:

- Do fi\_info structs only come from the library? If so, we need a runtime check to assure this.

- Add a version number to fi\_info itself, and if so, does that solve the msg prefix problem?

- Addressing these two will help us understand if we need to slip the release date, and if so, by how much.

This discussion brought us to a level of understanding of the issues and the possible solutions.

Next step – settle on a resolution either via email on the reflector or via discussion of the two Pull Requests.

Webex link: <https://cisco.webex.com/ciscosales/j.php?MTID=m9389b0513c9ae643d57e2381e254dcf5>  
Webex password: ofi

**Future Agenda Topics:**

- Sean’s rsockets topic

- porting libfabric to other platforms (e.g. Windows)

**OFIWG Download Site:** [www.openfabrics.org/downloads/OFIWG](http://www.openfabrics.org/downloads/OFIWG)

**Github:** <https://github.com/ofiwg/libfabric>

**OFI Software Download Site:** [www.openfabrics.org/downloads/OFI](http://www.openfabrics.org/downloads/OFIWG)

**Link to WebEx Recording** - [**Play recording**](https://cisco.webex.com/ciscosales/lsr.php?RCID=b8c86ac30be249fa8aa5701638717ff2)

**Next regular telecon**

Next meeting: Tuesday, 7/14/15

9am-10am Pacific daylight time