**OFI WG Bi-Weekly telecom – 07/26/2016**

**Agenda:**

* Roll call, agenda bashing
* GitHub Issues – cont’d

**News**

**Quick review of current GitHub issues**

* #1975 – How to expose loopback only devices? Some devices, e.g. USNic, don’t support loopback. Would be good if there is a way for such devices to report that they don’t support loopback. Same issue for devices that support only loopback.
	+ Non-loopback devices – a Cisco issue
	+ Loopback devices – Intel, for shared memory provider
* 7/12/16 update –
	+ Ben has opened two PRs, and updated the issue itself. See #2195, #2196. When an app needs some sort of local communication, there isn’t a way to expose that in the current FI\_GETINFO. In addition, so devices don’t support loopback at all. Want a way for providers to indicate what types of communications they support (e.g. local), and a way for an application to indicate what types of providers should be exposed to it.
	+ Sayantan to update 1975 to add some flavor to it.
	+ Question comes down to how the Intel folks feel about how to express XEON Phi locality. We could solve the problem now, or attempt to forecast a broader solution.
	+ Resolution: defer incorporating this change (that breaks the ABI), until we have a chance to get a better look at the shared memory provider. Plan to incorporate a change, one way or the other, in the end of year release.
* #1890 – Resizing FI\_CONTEXT structure. Today, defined as exactly four pointers. Should we define a larger context size, or allow the provider to specify its size? The question is if there is a reasonably sized context that would be useful, but without being so large that you might just as well allocate from the stack. For PSM, for example, the existing context size is useful, but for something like Atomics, a fairly large context space would be needed anyway. Would be helpful if for USNic could come up with a reasonable estimate. At present, it is pretty large. Consensus trend is that we don’t want dynamic allocation, which seems like it is likely to complicate things on both sides of the API.
	+ Intel - Sayatan Sur
* 7/12/16 update –
	+ Variable sized context is not very popular;
	+ GNI provider at the moment doesn’t use this because it’s too small; expanding to 64 bits might make it large enough to be usable. No currently known providers that would benefit from expanding to 64 bits. Howard will look at this, decide if it’s worthwhile for the GNI provider and update the provider appropriately.
	+ Resolution – table this for now because providers out there now would need a lot more context. Alternative is to expand context to 64 bits, which would require an ABI update.
	+ AR Howard – explore if there is benefit to the GNI provider to expanding the context space to 64 bits.
* #1394 – Deals with FI\_CONTEXT, but from the perspective of completions. No way at present for the consumer to know how the CQ is going to be used. Need to be able to specify when it is created, how a CQ is going to be used. Could be done with a capability field. A bit more difficult is to be able to specify the attributes an endpoint will attach to. Likely to impact any provider.
	+ All provider vendors
* 7/12/16 update –
	+ Doesn’t matter to the GNI provider (no help).
	+ PSM provider doesn’t care
	+ USNic not sure yet – the original issue doesn’t really impact USNic, but the proposed solution might address a related problem.
	+ Consider the following (next meeting):
		- Inverting the sense of the defaults from what Sean proposed on the issue,
		- Discuss the set of restrictions described on the issue
		- Discuss FI\_PRIVATE\_COMP
* 7/26/16 update
	+ Flipping from the use of mode bits to capability bits is intended to prioritize performance.
	+ Some are concerned that Sean would take the alternative point of view.
	+ The whole concept behind fi\_get\_info is that the provider should report the appropriate modes for best performance.
	+ Still no real resolution (on mode bits vs capability bits) here, but also no pressing need to resolve this today.
	+ Re FI\_PRIVATE\_COMP – this is the second of two mode bits that Sean had proposed.
	+ Nobody seems to see any powerful argument in favor of FI\_PRIVATE\_COMP, although FI\_RESTRICTED\_COMP is agreed to be useful (subject to the above discussion).
	+ Resolution – does not sound like anybody’s provider could take advantage of FI\_PRIVATE\_COMP.
	+ Re; what are the restrictions related to FI\_RESTRICTED\_COMP? An application might create e.g., two EP RDM endpoints, but with one configured for say, atomics and the other for RMA, but under some definitions of FI\_RESTRICTED\_COMP this may be prohibited. Some providers may be able to handle this, but others may not. Do we need to split FI\_RESTRICTED\_COMP into two separate modes (capabilities?).
	+ Downside – complicates the API, Upside – allows for maximum performance. May limit whether certain use cases can take advantage of this mode (or capability).
	+ Or, could add a restriction that CQs can’t be shared between different endpoint types. If needed, this restriction could be relaxed at a later date. This has the advantage of being simple and easier to understand. Nobody seems able to imagine a use case that demands the ability to share CQs between EPs of different types, nor does it seem like much of a burden to require the use of two CQs. Needs a few lines added to the man pages.
	+ For two EPs of the same type that support different operations, use FI\_RESTRICTED\_COMP as a mode bit.
* #1218 – FI\_MORE – is there a way to chain together multiple requests? As above, this one needs more focus. Probably less critical since it’s a potential API impact, not an ABI impact, but nevertheless…
	+ Cray - Sung
* Would be highly desirable to address these changes for the possible ABI change release coming toward the end of the year. Suggest adding these to upcoming meeting agendas.
* #2182 – Still being discussed on the issues list. Grappling with how to handle this on OpenMPI.
* **New topic** – Cisco’s intern has submitted a PR to add a fi\_pingpong in the main package, just like fi\_info is, rather than require the use of fabtest; the thinking being that a pingpong test is a very useful go-nogo. Strong agreement that this is a great idea, the call is for the community to go read the code.

**Next Agenda:**

* Take up discussion of issue 1218.

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

**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=1958bf4870474ca7937422c37cb7cd9c) (43 min 14 sec) |
| Recording password: BvntXQ6N |  |

**Next regular telecon**

Next meeting: Tuesday, 8/9/16

9am – 10am Pacific daylight time