**OFI WG telecon – 11/06/2018**

**Agenda:**

1. Opens
2. Support for PCI peer-to-peer transfers (e.g. GPUDirect, Direct GMA)

**Opens**

Still planning for a 1.7 release before EoY. May need to start release process a little early to beat the Christmas rush. RC1 probably before the end of November, or possibly even before Supercomputing.

**Discussion – Support for PCI peer-to-peer transfers**

Primary focus is to support p-2-p transfers in the API. Not trying to be the interface for the accelerator.

Proposal: Support for two models:

1. P2P hooking provider
   1. Likely only works with CUDA buffers
   2. Hooks are enabled via env variables, may add the capability to select hooks programmatically.
   3. OFI extensions:
      1. New capability bit FI\_P2P
      2. mr\_mode bit: FI\_MR\_P2P
2. Direct API support

**Discussion (from 10-09-18 meeting) – Introduction on accelerators**

No slides available today.

Possible overlap between accelerators and Smart NICs/FPGAs

Only thing obvious so far is related to memory registration. Need a way to indicate that the target is not on the CPU.

CPU would not use libfabric to communicate with the GPU.

Libfabric is not a CUDA replacement.

Do you want to allow a GPU to launch a data transfer via libfabric?

Any overlap with the RPM stuff? Not clear at this point, such an overlap may emerge. Whatever is being done for the current HA Whitepaper should not preclude that from occurring, and a GPU should be able to access remote persistent memory.

One case: a GPU accessing remote persistent memory. The RPM API enhancements should allow a GPU to do this.

Second case: consider a GPU as an accelerator performing a higher-level function e.g. mirroring. The GPU should be able to use libfabric to execute its higher-level function just like any other ‘application’.

What is the scope of libfabric w.r.t. a GPU?

Three cases:

1. An operation launched by a host CPU targeting a GPU for data transfer, and
2. An operation launched by the GPU itself where GPU memory is the target, and
3. An operation launched by the GPU where host memory is the target.

Is there a requirement (limitation) that the target address space be memory-mapped into the process’ memory space?

**Next meeting**

Tuesday, November 20, 2018

9:00 – 10:00AM PST

**Recording:**

|  |
| --- |
| **Every 2 week OFIWG/libfabric meeting (2018)-20181009 1610-1** |
| Tuesday, October 9, 2018 |
| 12:10 pm  |  Eastern Daylight Time (New York, GMT-04:00) |

|  |
| --- |
|  |

|  |  |
| --- | --- |
| [**Play recording**](https://cisco.webex.com/cisco/lsr.php?RCID=dc060b8e7b7e4f2f8517fa70ff3d1094) (39 min 39 sec) | |
| Recording password: xEuC9dvS |  |

|  |
| --- |
|  |

**Webex link:** See the OFA central calendar for meeting logistics. <https://openfabrics.org/index.php/ofa-calendar.html>

**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)