The GNI Provider Layer for OFI libfabric

Sung-Eun Choi[*], Evan Harvey[†], **Howard Pritchard**[†], James Shimek[*], James Swaro[*], Zachary Tiffany[*]

[*]Cray Inc.
St. Paul, MN

[†]Los Alamos National Laboratory
Los Alamos, NM

Cray User Group Conference 2016
London
May 12, 2016

LA-UR-16-23186

Content

- Open Fabrics Alliance, OFI and all that
- libfabric
- GNI provider implementation
- Results
- What's Next?

Open Fabrics Interfaces WG

Open Fabrics Interfaces WG (OFI WG) is a subgroup of the Open Fabrics Alliance chartered to

> *Develop an extensible, open source framework and interfaces aligned with upper-layer protocols and application needs for high-performance fabric services*

Working Group participants include fabric vendors, universities, government labs, Linux distro vendors, etc.

Participation in the OFI WG is open to anyone.

Libfabric is being developed by the OFI WG

The goal of OFI, and libfabric specifically, is to define interfaces that enable a tight semantic map between applications and underlying fabric services. Specifically, libfabric software interfaces have been co-designed with fabric hardware providers and application developers. Libfabric supports multiple interface semantics, is fabric and hardware implementation agnostic, and leverages and expands the existing RDMA open source community.

The libfabric API is an outcome of an analysis of a broad range of application spaces - HPC, data center, storage, etc.

On github: https://github.com/ofiwg/libfabric

Mac OS X

Why Libfabric on Cray XC (Aries)

Cray XC(Aries) dragonfly network and associated software stack (GNI) provide features to support a high performance libfabric implementation

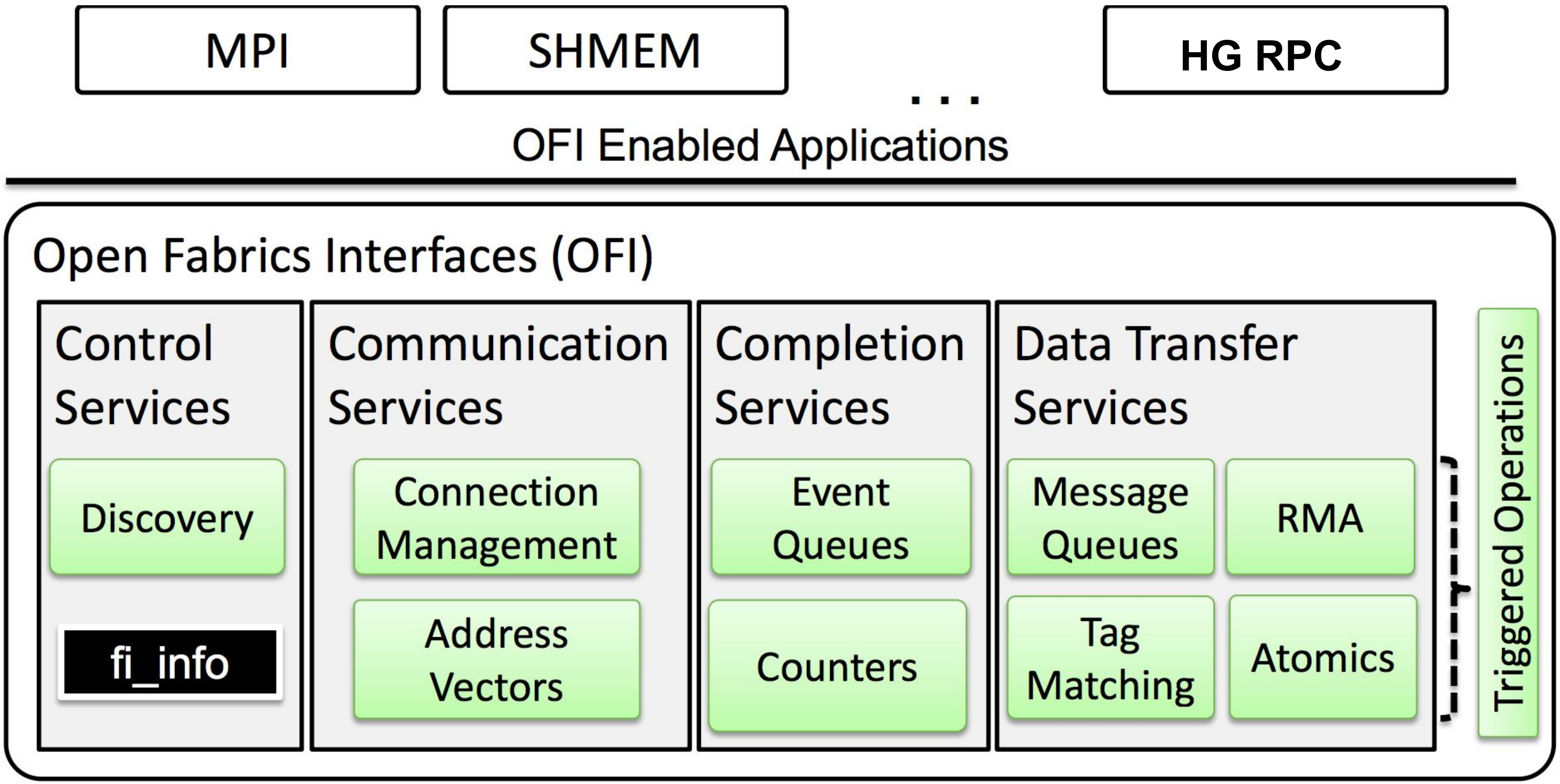
Such a libfabric implementation provides a forward-looking, portable eco-system to prepare program model communication middleware for future interconnects

Programming to libfabric (as opposed to proprietary or network architecture specific APIs) future-proofs middleware applications

LANL and Cray collaborating to develop a GNI provider

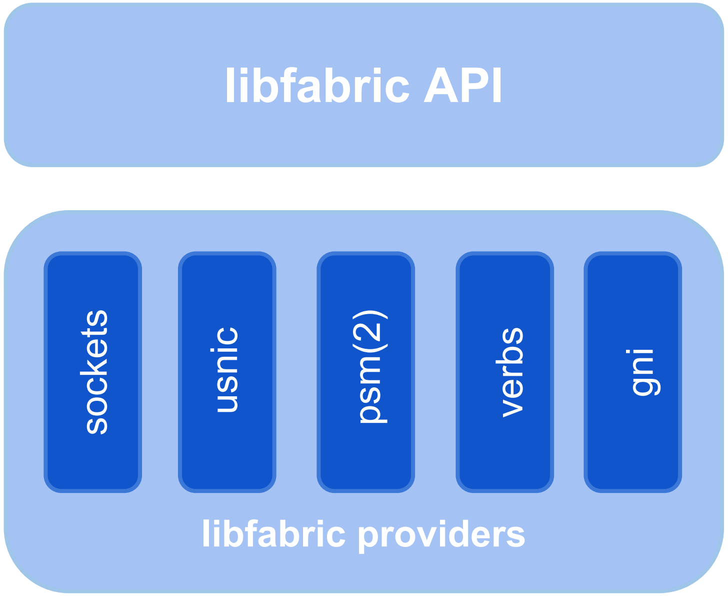
libfabric in a small nutshell

Libfabric (OFI) - elements of the API



Libfabric (OFI) - provider plugins

```
checking for CRAY_UGNI... yes
checking for CRAY_GNI_HEADERS... yes
checking for CRAY_ALPS_LLI... yes
checking for CRAY_ALPS_UTIL... yes
checking criterion path... yes
checking for CRAY_PMI... yes
configure: gni provider: include in libfabric
checking that generated files are newer than configure...
configure: creating ./config.status
config.status: creating Makefile
config.status: creating libfabric.spec
config.status: creating config.h
config.status: executing depfiles commands
config.status: executing libtool commands
***
*** Built-in providers: gni sockets
*** DSO providers:
***
hpp@edison06:~/libfabric-cray> (master)
```

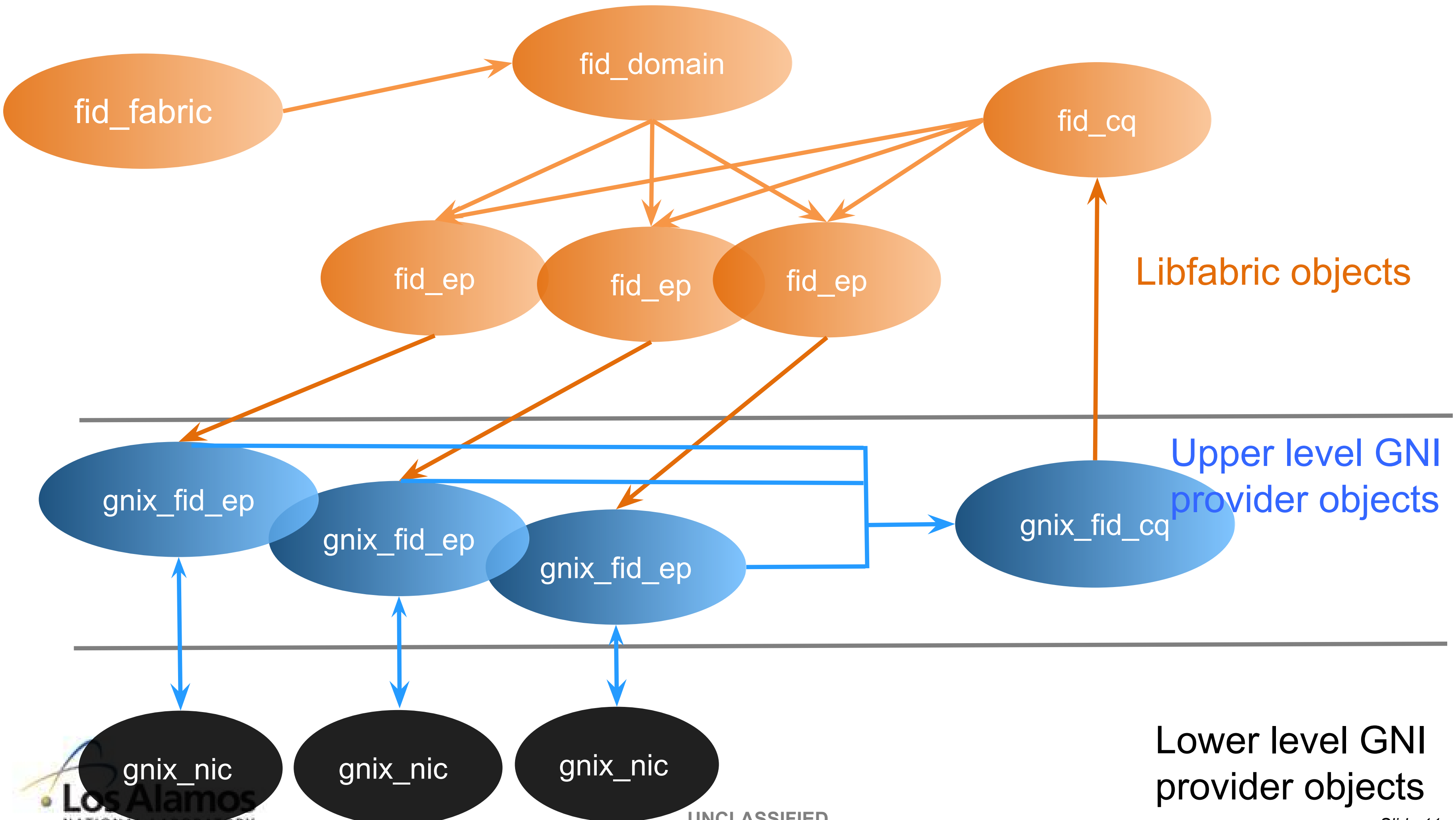


GNI provider implementation

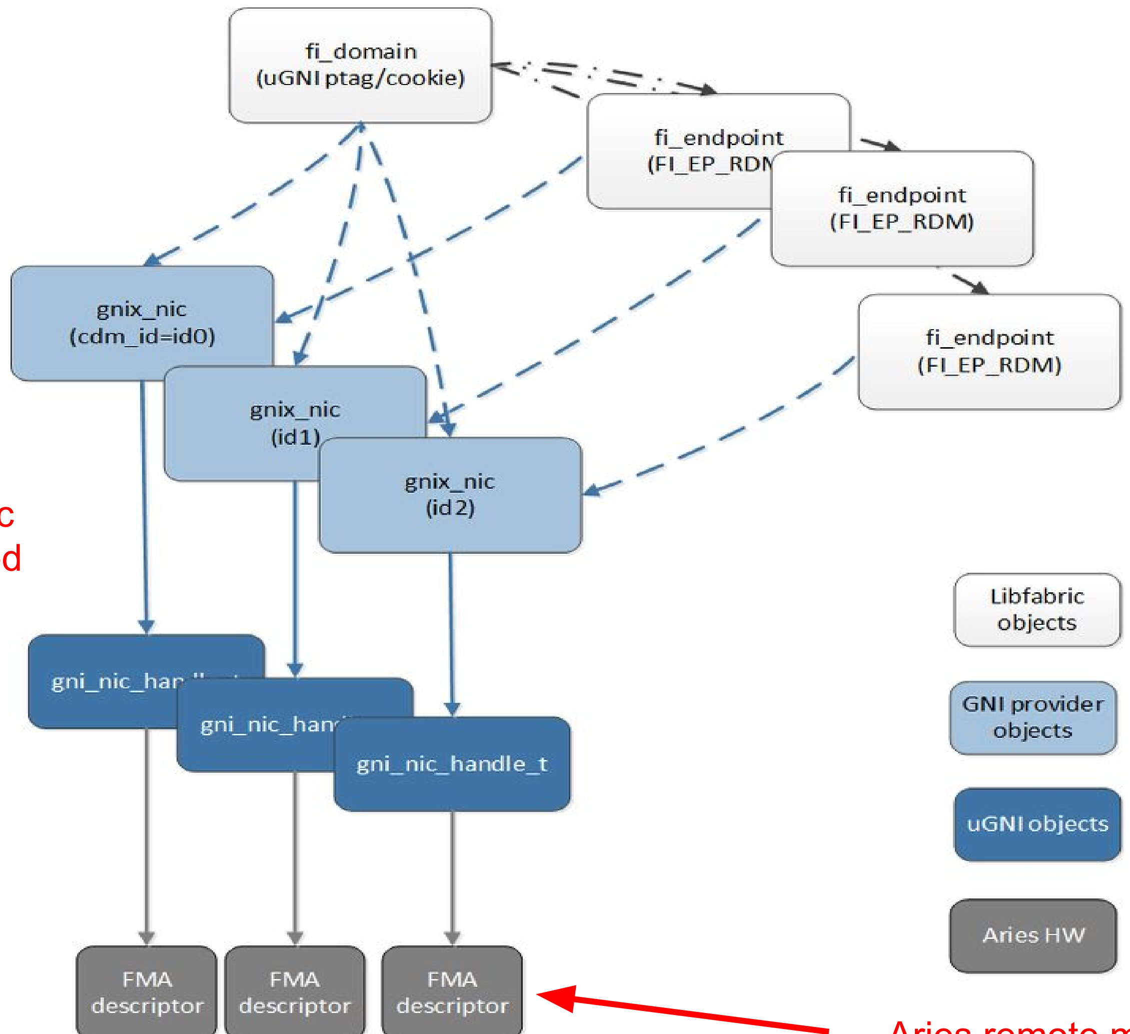
GNI Provider

- uses the uGNI API to access network resources
- supported on Cray XC Aries (not Gemini)
- requires Cray Linux Environment (CLE) 5.2 UP04 or higher
- requires GCC 4.9.1 or higher, can be built with Intel compiler
- thread safe by default and supports thread hot
- Supports FI_EP_RDM and FI_EP_DGRAM endpoint types
- downstream repo - https://github.com/ofi-cray/libfabric-cray (see wiki pages for help)

GNI provider: Upper level/Lower level class structure



GNI provider: Mapping GNI provider objects to uGNI objects/Aries hardware



access to a gnix_nic instance is serialized

Aries remote memory load/store units

Los Alamos NATIONAL LABORATORY EST.1943
Operated by Los Alamos National Security, LLC for NNSA

Results

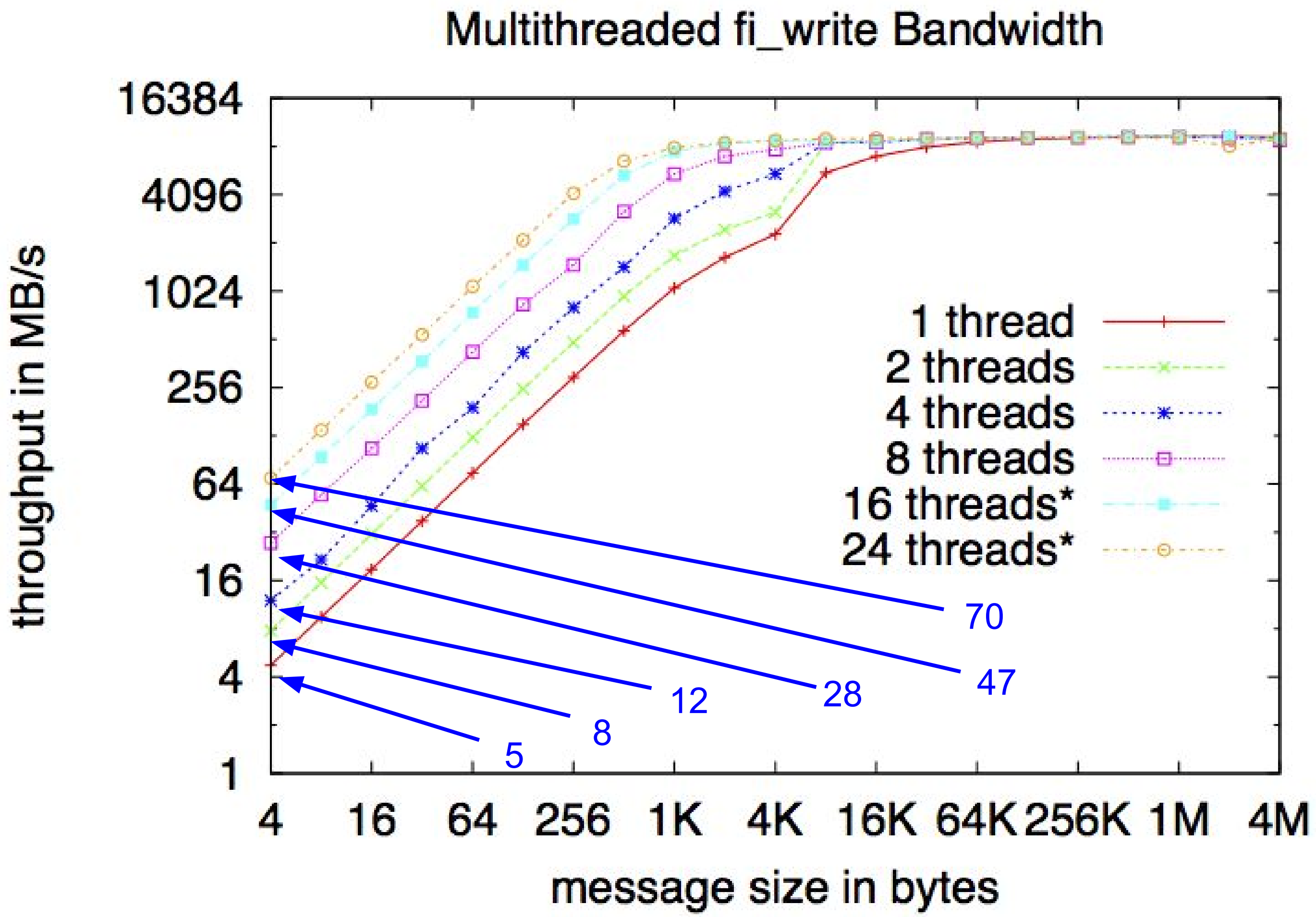
Test setup

- Cray XC 30
  - two Ivy Bridge (E5-2697 v2) processors/node
  - 12 cores (no HT)/processor
- Cray Linux Environment (CLE) 5.2 UP04
- libfabric 1.3 compiled with GCC 5.1.0 with -O2 optimization level
- libfabric tests at https://github.com/ofi-cray/fabtests-cray
- MVAPICH OSU 5.3 tests
  - modified to use `MPI_Init_thread`

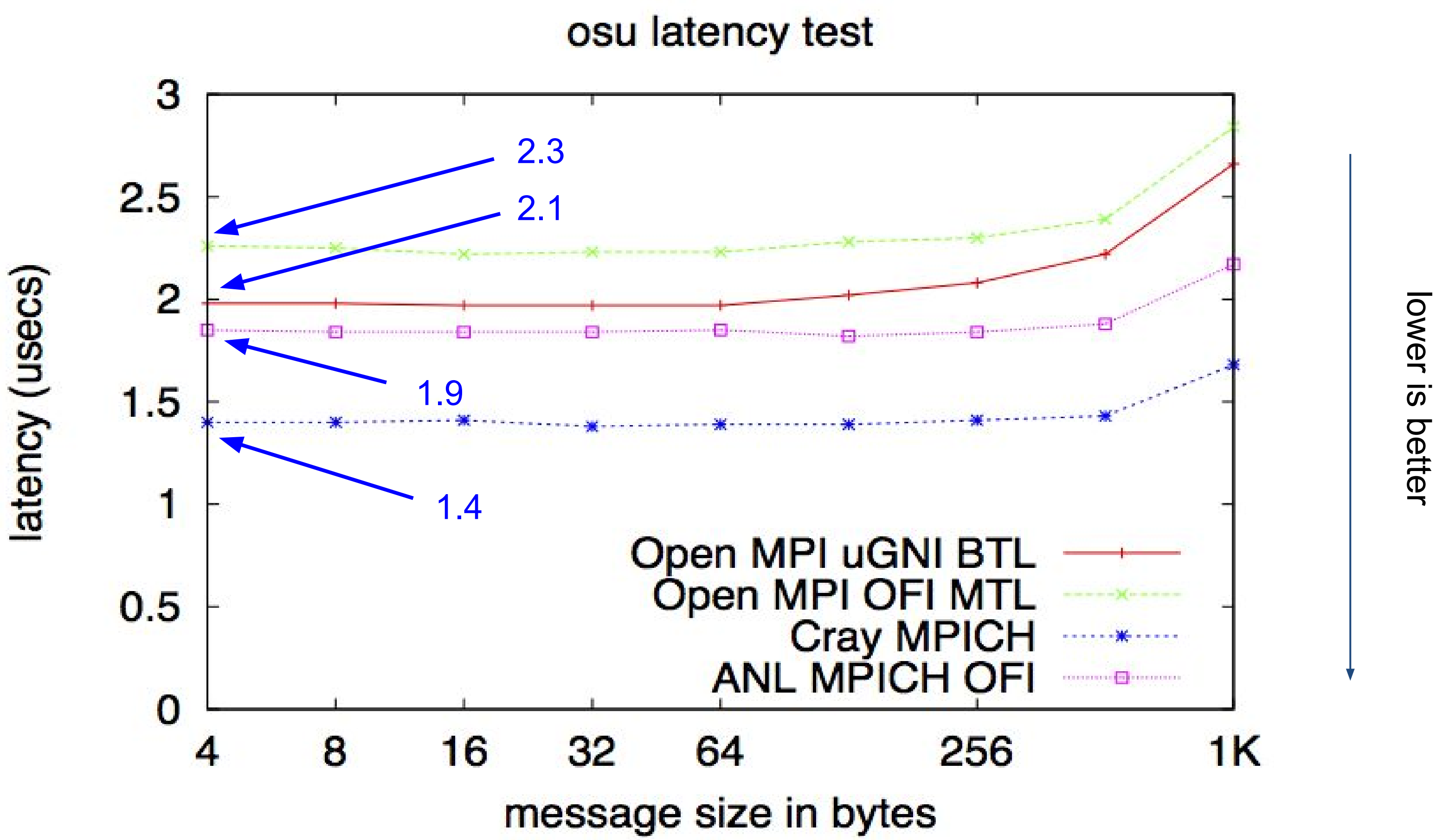
Test setup (continued)

- MPICH versions used
  - Argonne MPICH (CH3 device) using OFI netmod (patched to work with aprun/srun and Cray PMI)
  - Cray proprietary version of MPICH using GNI netmod
- Open MPI
  - using GNI BTL
  - using OFI MTL

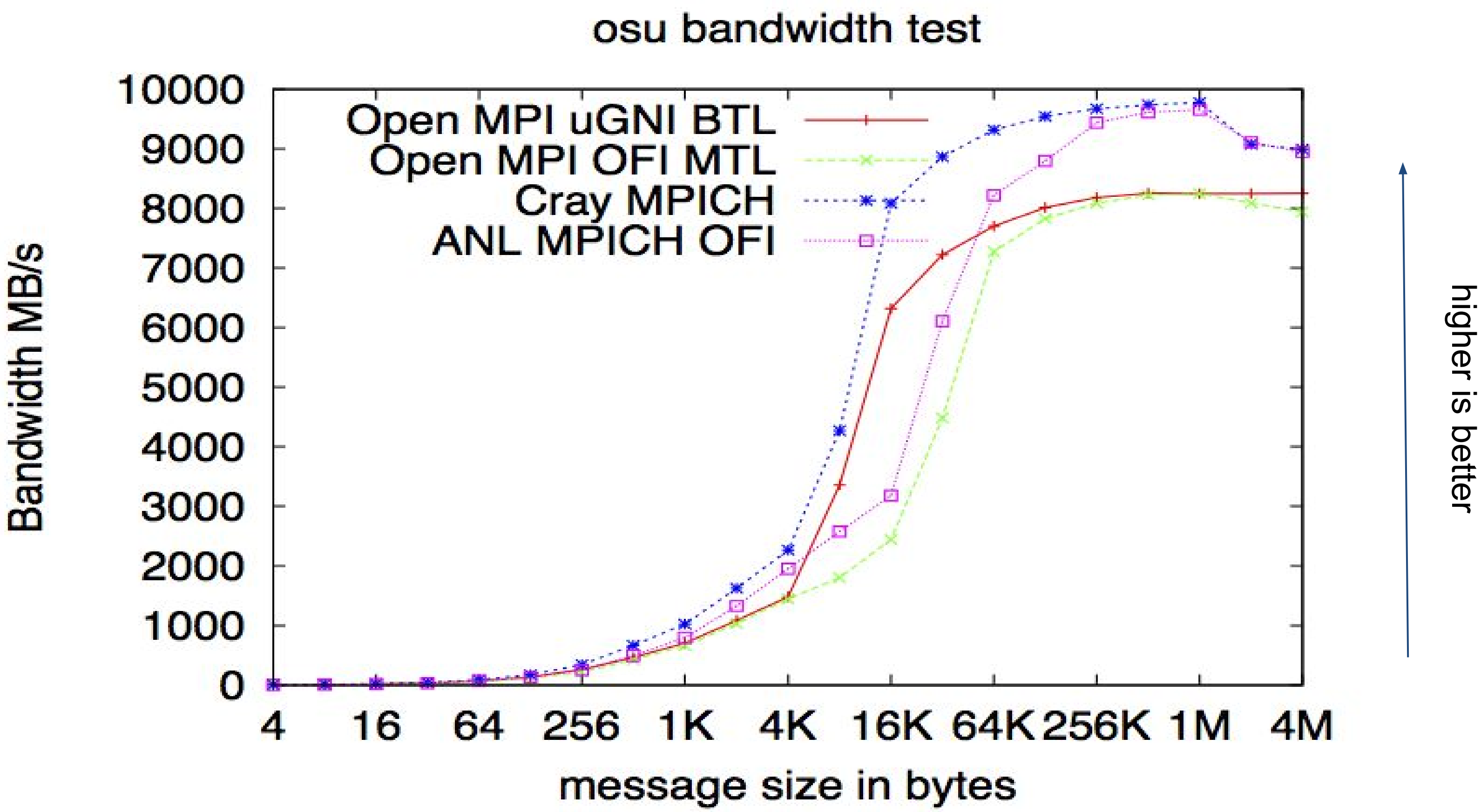
fi_write throughput using modified osu_bw



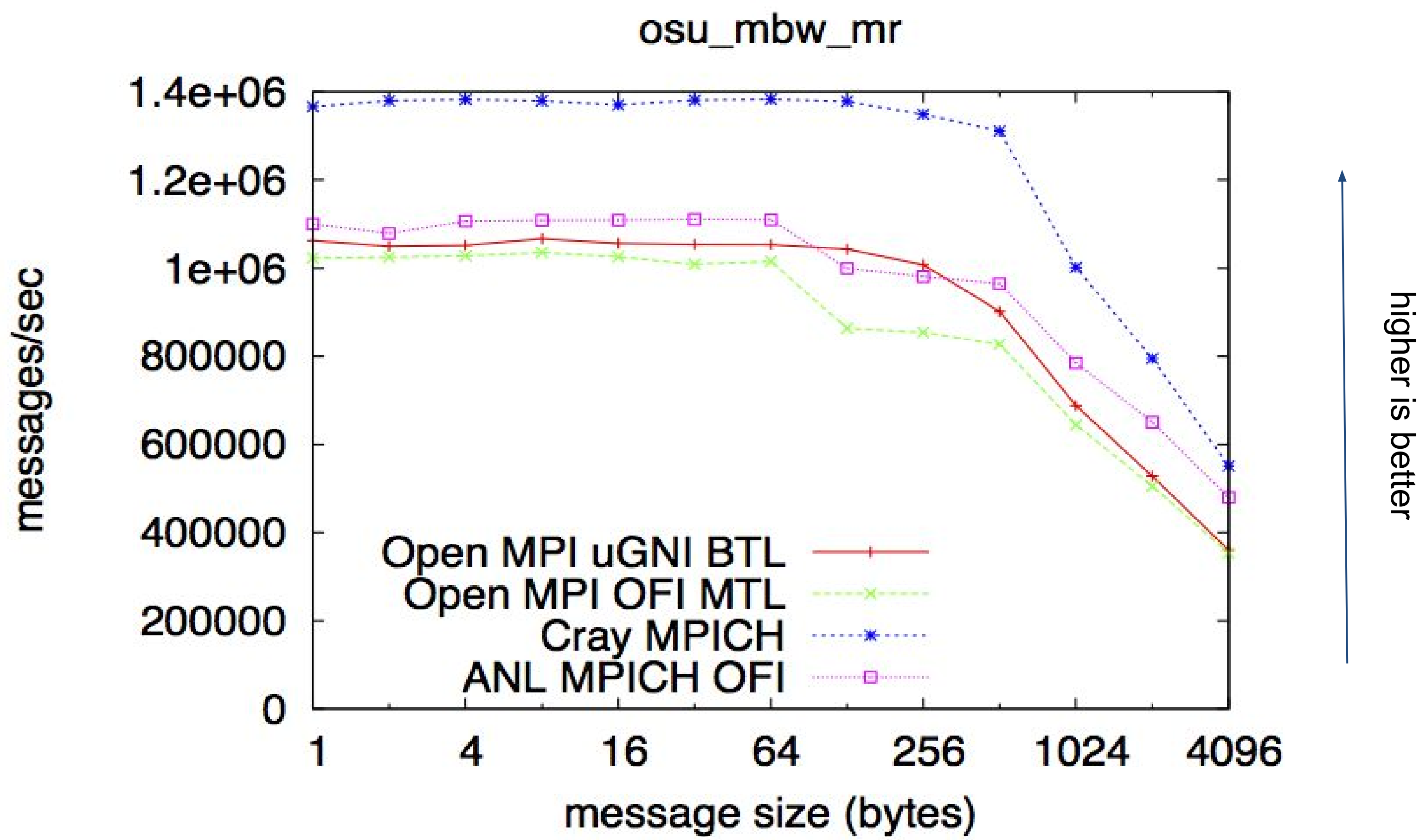
OSU Latency (MPI_THREAD_MULTIPLE)



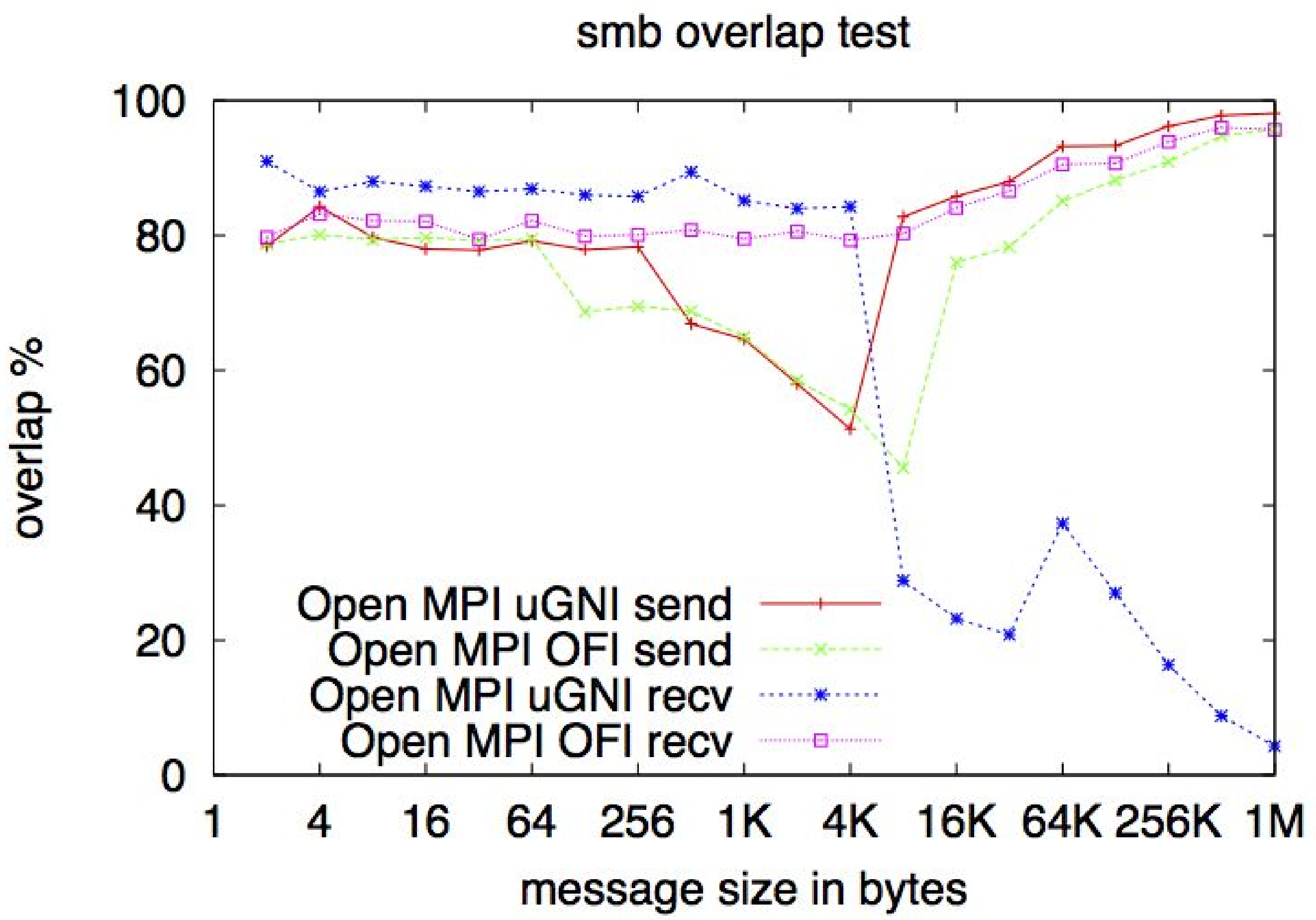
OSU bw (MPI_THREAD_MULTIPLE)



OSU mbw_mr (MPI_THREAD_MULTIPLE)



Sandia MPI overlap test



higher is better

Next Steps

- Features currently in upstream (git@github.com:ofiwg/libfabric) master not in 1.3
  - support for FI_THREAD_COMPLETION
  - triggered ops support
- Features planned for 1.4 release
  - performance improvements
  - scalable endpoints (SEPs)
  - support for non-native atomic memory op types (GUPS accelerator)
- Future releases
  - shared memory (xpmem) bypass for better intra-node transfers
  - support for FI_EP_MSG endpoint types

Questions