OPEN MPI AND RECENT TRENDS IN NETWORK APIs

#OFADevWorkshop

HOWARD PRITCHARD (HOWARDP@LANL.GOV)
LOS ALAMOS NATIONAL LAB
LA-UR-16-22559
OUTLINE

- Open MPI background and release timeline
- Open MPI Internals and Network APIs
- Lessons learned
- Advertisement
OPEN MPI BACKGROUND

- Open source implementation of MPI on github: https://github.com/open-mpi/ompi
- Developer community includes industry partners, government labs, and universities
- Lots of users. Widely used on IB clusters.
- Check the [github wiki page](https://github.com/open-mpi/ompi) for info on contributing to the project
OPEN MPI STRUCTURE

MPI/SHMEM Application

MPI API

PML (p2p messaging layer) base

OSC (MPI RMA) base

OB1 (message matching)

R2 (BML)

BTL Base

network

Vendor (including OFED)

Community property but a specific owner/maintainer

Community property

Vendor supported/developed part of Open MPI

Vendor (including OFED)

External projects
LIBFABRIC IN BOXES

Libfabric Enabled Middleware

- **libfabric**
  - Control Services: Discovery
  - Communication Services: Connection Management, Address Vectors
  - Completion Services: Event Queues, Counters
  - Data Transfer Services: Message Queues, RMA, Tag Matching, Atomics

- Supported or in active development
  - Sockets: TCP, UDP
  - Verbs: IB, RoCE, iWarp
  - Cisco usNIC
  - Intel Omni-Path
  - Cray GNI
  - Mellanox MXM
  - IBM Blue Gene
  - A3Cube RONNIE

OpenFabric Interface (OFI) is a framework focused on enabling fabric communication services to...
OPEN UCX IN BOXES

Accelerated verbs for MLX5
LESSONS LEARNED
Great for these folks – lots more tests, even if the API is only exercised through certain paths

Helps a lot if the developer can toggle between different underlying network transports – helps discriminate between bugs in a provider (libfabric speak) vs bugs in the Open MPI usage of the API
There have been some issues which needed solving (memory hooks interfering), there probably will be others. This may become a potential issue if the interference is due to a vendor proprietary component.

Useful for providing feedback to Network APIs – new features, etc. that would be great to have
FOR END USERS HOWEVER...

- Unlike previous network APIs like PSM, verbs, etc. multi-network software like libfabric and Open UCX can be confusing to users and administrators
- Which network APIs got built in to my Open MPI install?
- Which lower level API(s) (provider in case of libfabric, TL in case of Open UCX) am I using?
MAKE CONFIGURAY OUTPUT CLEARER

Open MPI configuration:
-----------------------
Version: 3.0.0a1
Build MPI C bindings: yes
Build MPI C++ bindings (deprecated): no
Build MPI Fortran bindings: mpif.h, use mpi
MPI Build Java bindings (experimental): no
Build Open SHMEM support: yes
Debug build: no
Platform file: (none)

Miscellaneous
--------------
CUDA support: no

Transports
-----------
Cray uGNI (Gemini/Aries): no
Intel Omnipath (PSM2): no
Intel SCIF: no
Intel TrueScale (PSM): yes
Mellanox MXM: no
Open UCX: no
OpenFabrics Libfabric: yes
OpenFabrics Verbs: yes
Portals4: no
Shared memory/copy in+copy out: yes
Shared memory/Linux CMA: no
Shared memory/Linux KNEM: no
Shared memory/XPMEM: no
TCP: yes

Resource Managers
------------------
Cray Alps: no
Grid Engine: no
LSF: no
Slurm: yes
ssh/rsh: yes
Torque: no

This feature is not in 1.10.2 and 2.0.0 releases.
WHICH NETWORK API OPTION IS BEST FOR MY CLUSTER?

- Not easy to tell from project descriptions
- Try some simple MPI benchmarks with your install.
BASIC BENCHMARK RESULTS

- Used two clusters at LANL
  - An AMD Opteron/Mellanox Connect X2 (pci-e gen2)
  - An Intel Sandy-Bridge/Truescale (pci-e gen2) system

- Open MPI (master, aka 3.0.0a1), no special config options (if possible), except those needed to pick up libfabric and Open UCX installs

- Considerable –mca parameter specification was required to pick up the right PML and/or MTL. This is not bad.

- Used libfabric master@13f841c and Open UCX master@4103c00
UCX needs work to support max inline data 0.

Performance of libfabric verbs provider not good for large messages – not unexpected though.
The diagram shows latency (usec) on the y-axis and message size (bytes) on the x-axis for different configurations: ucx-rc, ofi/verbs(rc), mxm-rc, yalla-rc, and ob1. The y-axis ranges from 0 to 16, and the x-axis ranges from 1 to 4,096 bytes.

Lower is better in terms of latency. The graph indicates that ofi/verbs required fixes to work, as seen by the higher latency compared to other configurations.

The diagram is labeled as 'osu_latency'.
Currently can’t coax UCX to use a zcopy path

ofi/verbs not able to use application’s memory registrations.
A FEW CONCLUSIONS

- Both libfabric and Open UCX are works in progress.
- In areas with vendor focus, things look promising.
- For older interconnects, or those that may not be main focus of vendors, not so clear.
- Open source solutions help reduce risk of interoperability issues with other components of Open MPI.
LANL/NMC LOOKING FOR STUDENTS AND POST DOCS

https://newmexicoconsortium.org/home/jobs
THANK YOU