The Common Communication Interface

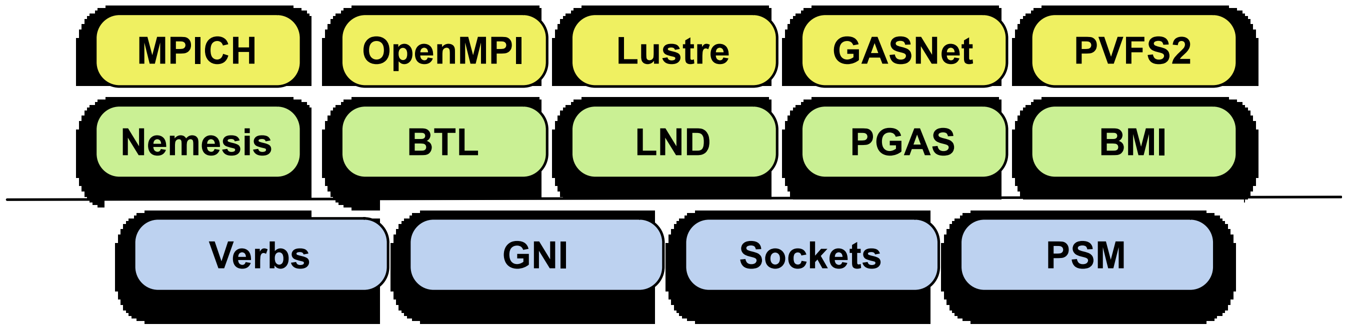
Patrick Geoffray – Myricom

Galen Shipman – Oak Ridge National Laboratory

Collaborators: Scott Atchley, George Bosilca, Peter Braam, David Dillow, Brice Goglin, Ken Matney, Ron Minnich, Jeff Squyres, Geoffroy Vallee

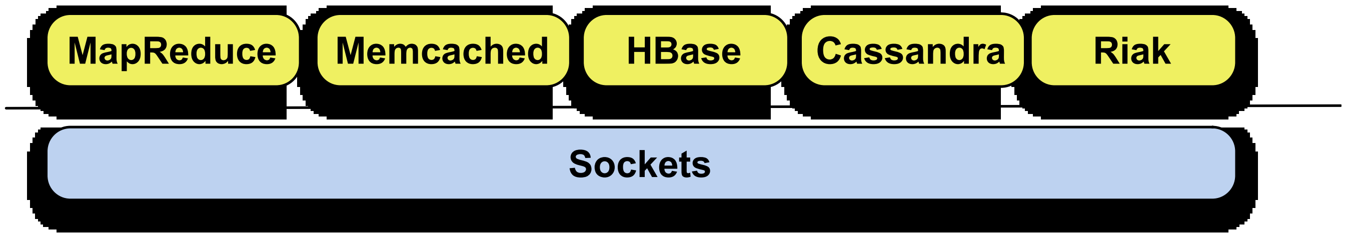
State of the Art in Middleware

- Most HPC middleware solutions have developed complex software stacks to enable portability
  - HPC networking is one area where maintaining portability is a core requirement
  - Significant effort is spent in optimizing and maintaining these portable software stacks



The Data Center Relies on Sockets

- Currently the only viable alternative to maintaining a portability layer is to use Sockets
- Sockets provides the portability needed by modern data intensive workloads
  - Performance and scalability can remain elusive



Sockets in Data Centers

- Sockets is the de-facto standard Application Programming Interface (API) in networking
  - Portable, robust, simple
- Commonly uses TCP or UDP on the wire
- Designed in the 1980s
  - Relatively slow and lossy networks
  - Limited host concurrency

The Sockets API Has Problems

- Difficult to leverage networking innovations:
  - Semantics incompatible with zero-copy techniques
  - No portable support for asynchronous operations
  - Poor scalability with per-peer buffering and polling

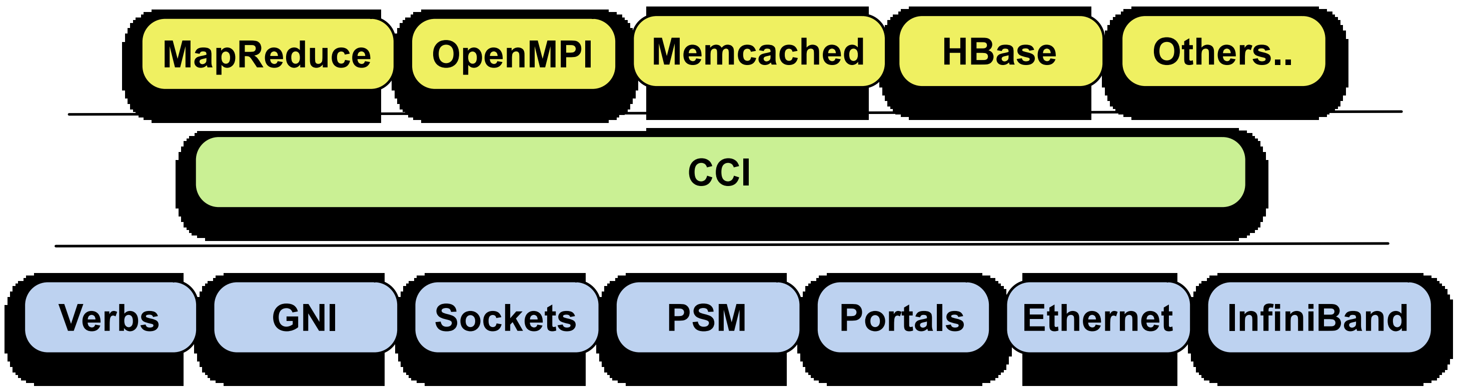
- A bottleneck on application performance
  - Bad at 10GbE, worse at 40GbE or 100GbE
    - Rsockets shows improvements in narrowing the gap at the cost of more CPU overhead (without zero-copy)

What about Verbs?

- High degree of complexity
  - See Sean Hefty's talk on Rsockets
- Need for broader portability
  - Christopher Lameter's talk on OFED Use in the Financial Industry

Where does CCI fit?

- Our vision: A common API with support for all major HPC and big-data interconnects
- But.. CCI is NOT a replacement for other APIs
  - For some use-cases going straight to the "native" API would be preferred



Breaking the Bottleneck

- Need an alternative programing interface to reap the benefits of high-performance networks
  - While keeping things simple !
- Experiences from high performance interconnects:
  - Techniques: OS-bypass, zero-copy, scalability
  - Vendor-neutral ecosystem through an open API

A Modern RDMA Network API

- Common Communication Interface (CCI)
  - Performance: low latency, high throughput, low CPU overhead, efficient multi-thread and NUMA
  - Scalability: no per-peer resources
  - Robustness: connection-oriented model
  - Portability: network and vendor neutral
  - Simplicity: compact API, event-driven

- ***A modern paradigm for RDMA networks***
  - *A simple, flexible and logical API.*

View from the top, today

	Sockets	MPI	Specialized APIs
Performance	✗	✔	✔
Scalability	✗	✔	Varies
Portability	✔	✔	✗
Robustness	✔	✗	Varies
Simplicity	✔	✗	Varies

State of the Specialized Art

- IB Verbs
- iWarp Verbs
- RoCE Verbs
- PSM
- MX
- Portals
- GNI
- Gasnet
- RDS
- QsNet
- DAPL
- VIA
- GM
- DCMF
- LAPI
- AM
- And many more…

Design choices

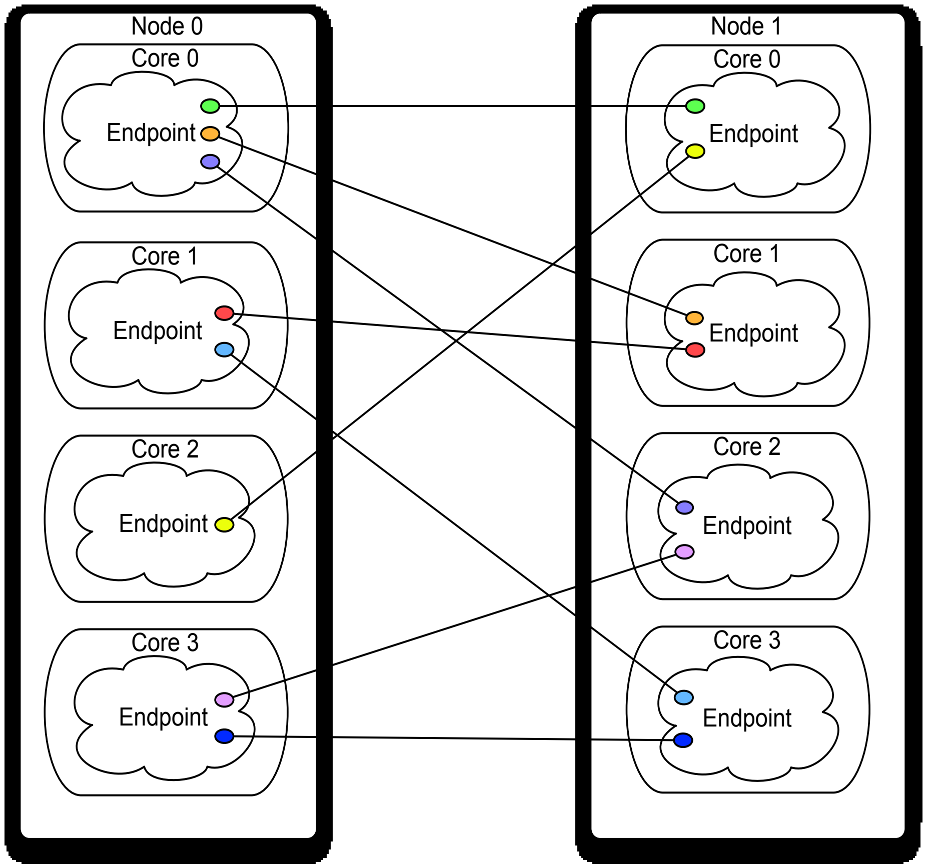
- Receive semantic: Ma[illegible]g vs FIFO ?
- Buffering management: Ap[illegible]on vs Library ?
- Notification: Events vs Ha[illegible]rs ?
- Connections: Explicit vs I[illegible]it ?
- Communications: Buffered vs Zero-copy ?

Design choices

- Efficient teaming/bonding, adaptive/dispersive routing
  - Relax order when possible
- Breaking bad habits
  - RMA (one-sided) operations are not the best choice for small message latency
  - No last-byte-written-last assumption (see order above)
- Simple is easier to learn, use, debug, maintain and tune

CCI Basics

- Endpoints
  - Virtualized instance of a device
- Connections
  - Allows granular control of reliability and ordering attributes
- Communication
  - Small Messages
  - Remote Memory Access



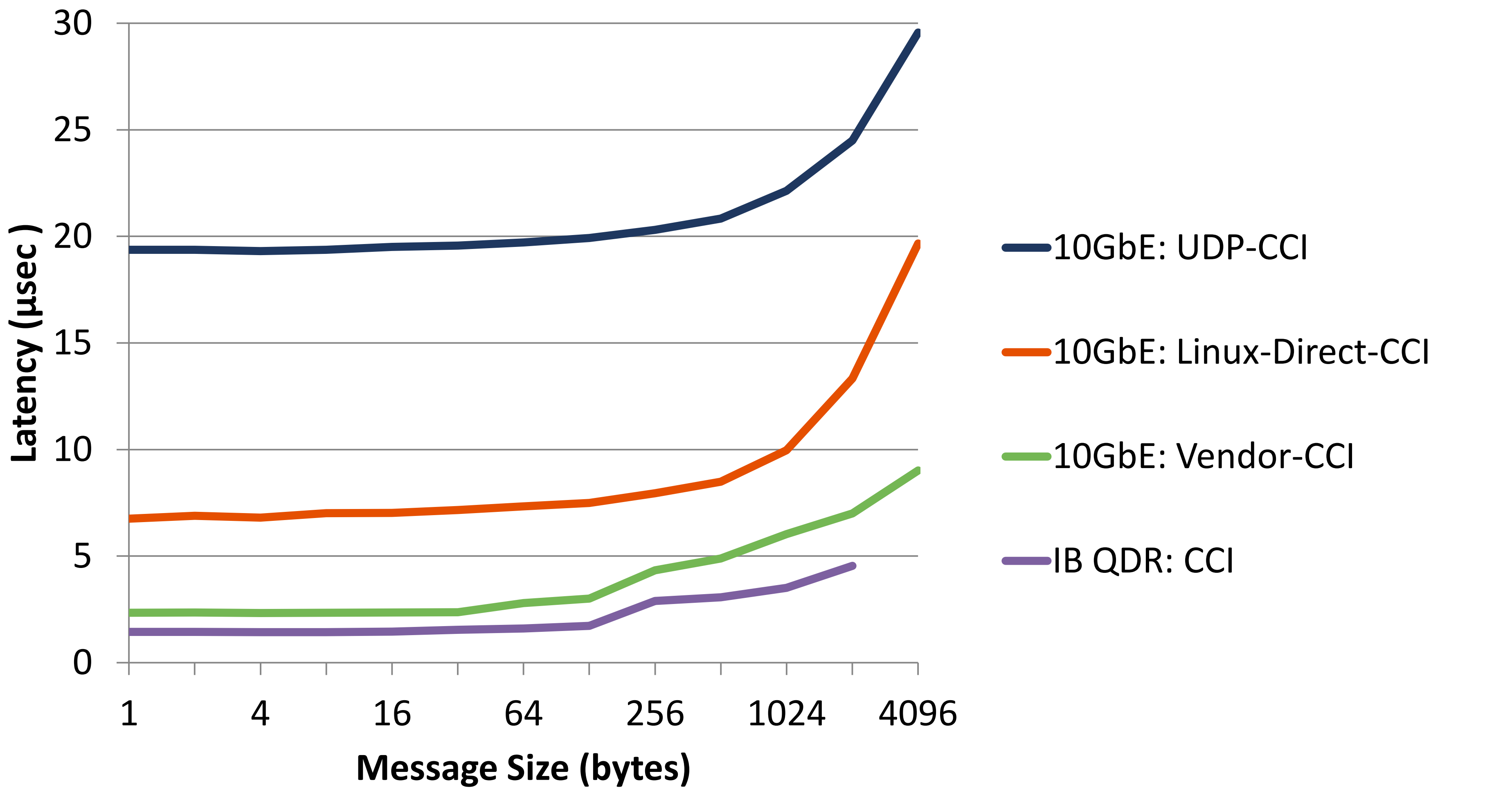
Endpoints and Connections

- Endpoints
  - Complete container of resources
  - An event driven model
    - Application may poll or block on single file descriptor
    - Events include send, recv, connection establishment, etc.
    - Events may contain resources (buffers for messages)
- Connections
  - Per peer – one endpoint can have many connections
  - Scalable, no per-peer resources (buffers or queues)
  - RO, RU, UU, MC_TX, MC_RX

Communication

- Small Messages
  - Always buffered on both send and receive side
  - Library manages buffers, not the application
  - Message may be processed in-place
  - Limited to transport-specific MTU
- Bulk Data
  - RMA communication for bulk-data transfer
  - Zero-copy when available
  - No implicit order for efficient link aggregation
    - explicit fence
  - May be combined with delivery of a remote Event

Modern Network Performance + Portability



Smooth Transition

- CCI will not replace Sockets overnight
  - Both are complementary in data centers
  - Migrate performance-sensitive, intra-application communication to CCI

CCI	**Sockets**
Application controls both sides of the communication	Application controls only one side of the communication
Performance gain worth the porting effort	Existing implementation is good enough
East-West traffic	***North-South traffic***

Our Approach

- CCI defines the API not the software stack
  - Free to innovate under a common API
- BSD-style license
  - Easy to commercialize your derivative work
  - Easy to leverage existing code base
  - Protects your IP
- Apache-style contributor agreement
  - Protects the entire CCI community

Current Partners

OPENFABRICS ALLIANCE

CISCO

Inria
INVENTORS FOR THE DIGITAL WORLD

Myricom®

OAK RIDGE
National Laboratory

Parallel Scientific

Sandia
National
Laboratories

THE UNIVERSITY of TENNESSEE UT
KNOXVILLE

Conclusion

- Sockets API cannot leverage modern NIC's capabilities
- We propose CCI, a novel communication interface built on over a decade of high performance networking experience
- CCI allows application to fully benefit from modern networks
- CCI enables an open, vendor-neutral high performance networking ecosystem

Questions?

Visit http://cci-forum.com

Patr[redacted]ffray

patric[redacted].com

Galen Shipman

gshipman@ornl.gov