“Verbs” API (VAPI)

- IB/iWARP actions known as “verbs”
  - Send verb, receive verb, etc.
- First IB VAPI was Mellanox VAPI (mVAPI)
  - Now deprecated
- OpenFabrics has different VAPI
  - Similar concepts, but different API
No Unexpected Receives

- All messages *must* be “expected”
- Receiver must pre-allocate resources
  - Pool of buffers to receive messages
  - Pool of buffers as target for RDMA
- Unexpected message triggers an error
Virtual Lanes / Service Levels

• OpenFabrics traffic divided into virtual “lanes”
  ▪ Virtual separation of traffic
  ▪ Analogous to MPI communicators (!)
  ▪ Can be assigned QoS-like attributes
  ▪ Weighting, etc.

• Service levels maps to lanes
Some OpenFabrics Queues

• Queue Pair (QP)
  ▪ Unit of connection in OpenFabrics
  ▪ Think of as “sockets” for OpenFabrics
  ▪ Send queue + receive queue

• Completion queue
  ▪ Most OF verbs are non-blocking
  ▪ OF driver puts events on this queue to signal when a verb has completed
Registered Memory

• InfiniBand/iWARP are RDMA-based networks
  ▪ Directly sends / receives from RAM
  ▪ Without involvement from main CPU
• But…
  ▪ Operating system can change virtual ↔ physical RAM mapping at any time
1. MPI says “IB: send this buffer”
2. HCA obtains physical address
3. HCA starts sending
4. OS changes physical mapping
5. HCA now sending garbage!
Race Condition

1. MPI says “IB: send this buffer”
2. HCA obtains physical address
3. HCA starts sending
4. OS changes physical mapping
5. HCA now sending garbage!
Race Condition

1. MPI says “IB: send this buffer”
2. HCA obtains physical address
3. HCA starts sending
4. OS changes physical mapping
5. HCA now sending garbage!
1. MPI says “IB: send this buffer”
2. HCA obtains physical address
3. HCA starts sending
4. OS changes physical mapping
5. HCA now sending garbage!
“Registering” Memory

• Solution: tell OS not to change mapping
  ▪ “Pinning” (“locking”) memory
  ▪ Guarantees that the message will stay in the same physical location until HCA is done

• “Registering” memory does two things:
  1. Pinning virtual $\leftrightarrow$ physical mapping
  2. Notifying HCA of the mapping
Registered Memory Problems

- Registering and unregistering is slow
- OS can only support so much registered memory at a time
  - Pinned pages are unswappable
- Must be careful to set ulimits properly (OFED)
Registered Memory Footprint

- How much registered memory does Open MPI use?
  - A complicated answer
  - Requires some background information first…

- For reference:
  - Complete answer (for v1.2 and beyond):
    
    http://www.open-mpi.org/faq/?category=openfabrics#limiting-registered-memory-usage
Common MPI Trick

- **MPI_SEND**(buffer, ...)
  - Register the buffer
  - Do the send
  - Return (leaving the buffer registered)

- Rationale: next time you send from that buffer, do not pay registration cost again
  - Great for benchmarks!
  - Usually not great for real applications

-OMPI does not do this (…by default)
Problems of User Registration

• Can run out of registered memory
  ▪ MPI must implement eviction policies

• Application can free buffer
  ▪ MPI *must* intercept free() or sbrk() to unregister memory before given back to OS
  ▪ Extremely problematic

• So just say “No!”
  ▪ …except for benchmarks 😞
More Information

• Open MPI FAQ
  ▪ General tuning
    http://www.open-mpi.org/faq/?category=tuning
  ▪ InfiniBand / OpenFabrics tuning
    http://www.open-mpi.org/faq/?category=openfabrics