ADVANCING OPEN FABRICS INTERFACES

Sean Hefty, OFIWG Co-Chair
Intel Corporation
March, 2017
MOVING LIBFABRIC FORWARD

Enhancements
• Deferred and new features

API adjustments
• Developer feedback

Performance optimizations
• Provider implementation

Many features require migration to API 1.5
Existing apps work unmodified
OFI DEVELOPER GUIDE

New!

Broad coverage of libfabric

What a novel concept!

Design motivations, architecture, usage guidance

Assumes minimal background
Reviews sockets API for context

Analyzes HPC requirements and their impact on an API
Memory copies, network buffering, asynchronous operations, interrupts, direct HW access, ...

Lower the barrier to adoption
Many concepts are generic
Examines API trade-offs
Call setup, branches, memory footprint, ...

Defines communication primitives
Command queues, memory buffers, completion notification, progress models, data and message ordering, ...

Then it goes into detail about OFI
https://github.com/ofiwg/ofi-guide
AUTHORIZATION KEYS

Isolate traffic between virtual sets of peers

Associated with endpoints and memory regions

Memory regions are only accessible through endpoints with matching keys

Only endpoints with the same key may communicate

Job Keys
MULTICAST

Datagram endpoint capability

New fi_join call to connect to group
- Supports send-only joins
- Returns fabric address (fi_addr_t) for group

Uses existing data transfer APIs
- Only message queue interface currently defined
- Multicast sends require use of FI_MULTICAST flag
  - Distinguish between unicast and multicast address
  - Received messages also marked with FI_MULTICAST
NEW ENDPOINT TYPES

Apps/middleware designed around socket semantics

Synchronous completions
No completion queue
Application owns buffers

Enable provider specific optimizations
RMA, circular receive buffers
Minimize data copies, use offload

Associated with file descriptor
Support for select/poll

Enterprise / Cloud

sockets rsockets ZeroMQ ... nanomsg

libfabric

OFI Provider

CM EQ EP:SOCK_STREAM fd
AV EP:SOCK_DGRAM fd

Beyond HPC
RMA key (not address) identifies the buffer

(RMA) Atomic

Tagged Atomic

FI_TAGGED op flag
- Replace key with tag

fi_atomic_query()
- Domain level call
- Also reports datatype size
DEFERRED WORK QUEUES

Generalizes triggered operations

Conceptual domain level work queue

Primitives to develop and evaluate collective APIs e.g. directed acyclic graphs

Allows (theoretical) associations between any domain level object Definition is limited

Experimental

work request

work queue

mem

ep

cq

3 5 7

counter

Conceptual model only
OPTIMIZING COMPLETIONS

Reduction of Overhead

**FI_RESTRICTED_COMP**
- Only similar endpoints will share CQs / counters
- *Eliminates post-processing*

**FI_NOTIFY_FLAGS_ONLY**
- Drops operation flag
- Retains notification flags:
  - Remote CQ data
  - Multi-receive buffer freed
- *Avoids op save/restore*

New mode bits
COMPLETION DATA

**Multi-Threading**

**App Request**

**Reporting errors**
- Provider specific data reported for debugging
  - Data is opaque to application
  - Storage managed by provider
  - Report maximum size of error data
  - Application provides buffer

*Improves multi-threaded completion processing*

**FI_SOURCE_ERR capability**
- Validate source address against local AV
- Report raw fabric address if not found
  
  EADDRNOTAVAIL completion error

*Pushes ‘connect’ address exchange to provider*
## FEATURE GRAB BAG

### New attributes

- Fabric: api version
- Domain: counter limits, memory region iov limits

### Communication scope

- Local: within the local node
  - Shared memory, most NICs
- Remote: with external nodes
  - Any NIC
  - Evading more refined definitions

### String based addressing

- FI_ADDR_STR address format
- Generic string format for any address
  - Application does not *need* to interpret
  - Well-known format for sockaddr
- Pass directly to `fi_getinfo()` or AV insertion
Simplify Development

PROVIDER UPDATES

Endpoints
- DGRAM
- MSG
- RDM

Capabilities

Provider: “udp;ofi-rxd”

Utility Providers

OFI

OFI

Sockets (TCP)  UDP  Cisco* usNIC  Verbs  Intel® OPA PSM  Intel® OPA PSM2  Cray GNI  IBM* Blue Gene  Mellanox Mlx

Core Providers

Exported X Core EP type

* Other names and brands may be claimed as the property of others
MEMORY REGISTRATION

Exposing the Pain

FI_MR_BASIC
Traditional RDMA fabrics

FI_MR_SCALABLE
Application driven capability

FI_MR_NEITHER
What other fabrics support
Redefine domain attribute mr_mode
- enum → integer flags

Divide FI_MR_BASIC into specific restrictions
- FI_MR_SCALABLE implied if mr_mode = 0

FI_MR_BASIC, FI_MR_SCALABLE kept for compatibility
- Behavior dependent on selected API version
**‘BASIC’ MR MODE BITS**

- **FI_MR_VIRT_ADDR**
  - Target offset is virtual address

- **FI_MR_ALLOCATED**
  - Backed by physical pages

- **FI_MR_LOCAL**
  - Locally accessed buffers must be registered

- **FI_MR_PROV_KEY**
  - Provider sets key

Exposing the Pain
FI_MR_RAW
More generic version of ‘basic’ registration

Provider selects base address

base address: ?

> 64-bit keys

FI_MR_RAW_ATTR()
Retrieve MR attributes

fi_mr_map_raw()
Peer sets up MR

It’s only a flesh wound
**FI_MR_MMU_NOTIFY**
- App notifies provider when physical pages backing MR change
- I.e. almost scalable, but NIC not linked into MMU
- Only those addresses accessed need to be backed

**FI_MR_RMA_EVENT**
- App indicates if MR will be used with RMA event counting
- MRs must be enabled after resource binding
LOOKING BACK

- Balance application needs, developer requests, and provider capabilities
- New features for expanding use cases
- Developer assistance
- Sensible performance optimizations
THANK YOU
Sean Hefty, OFIWG Co-Chair
Intel