ziti

UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

Extoll Lustre Network Driver

Overview and Preliminary Results

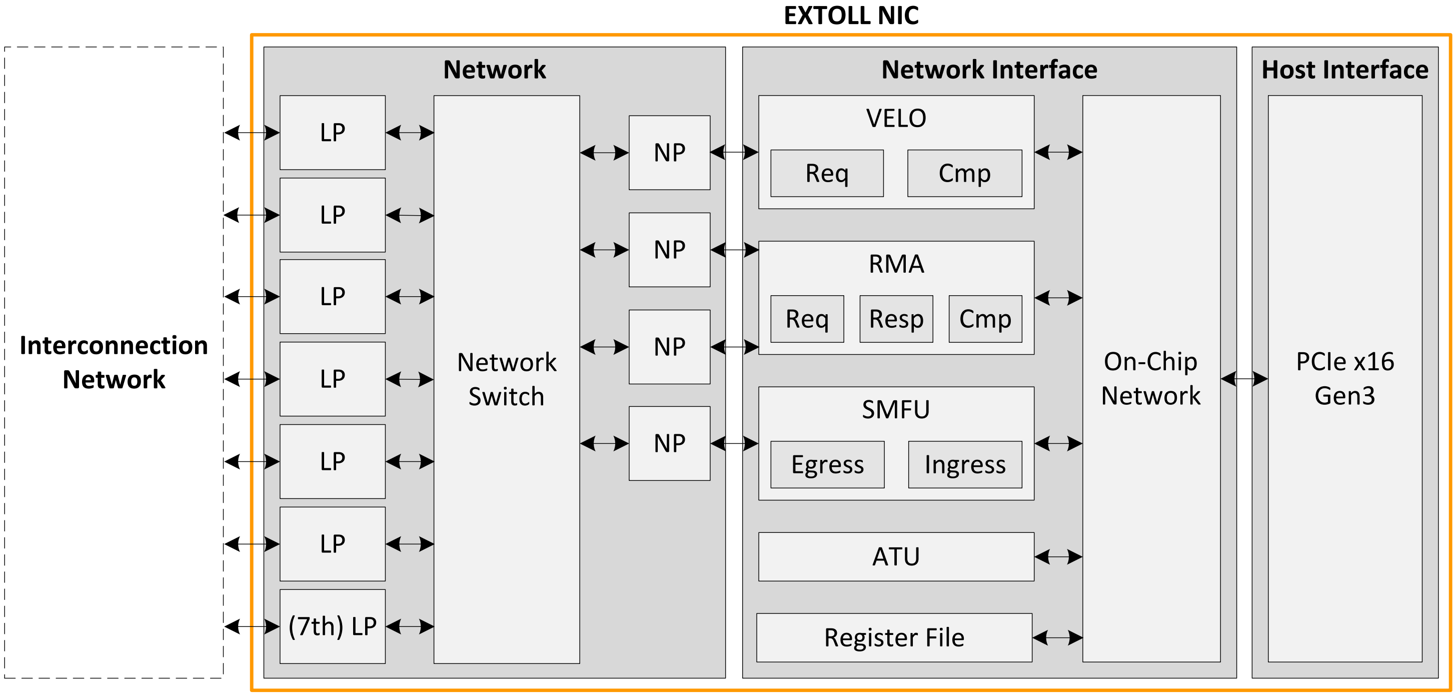
Sarah M. Neuwirth
University of Heidelberg, Germany

LAD’18, Paris, 2018

Extoll Interconnect
Architectural Idea

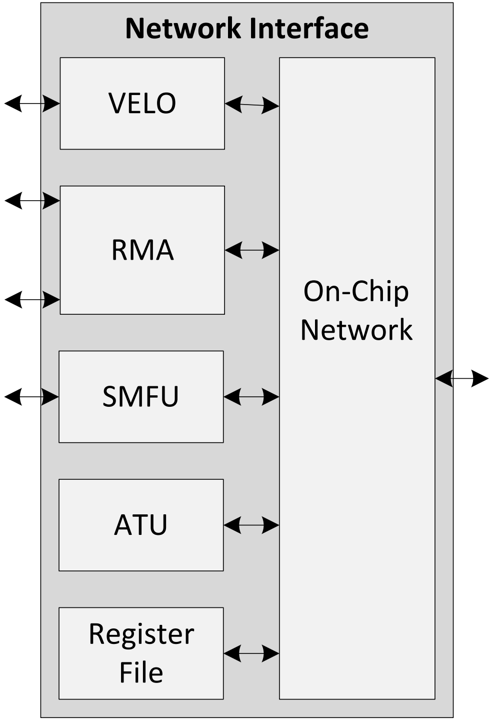
- Lean network interface
  - Low latency message exchange
  - High hardware message rate
  - Optimized memory footprint for scalability
- Switchless design
  - 3D Torus direct network
  - Reliable network
  - High Scalability
- Efficient, pipelined hardware architecture

Extoll Interconnect
Hardware Architecture – Overview



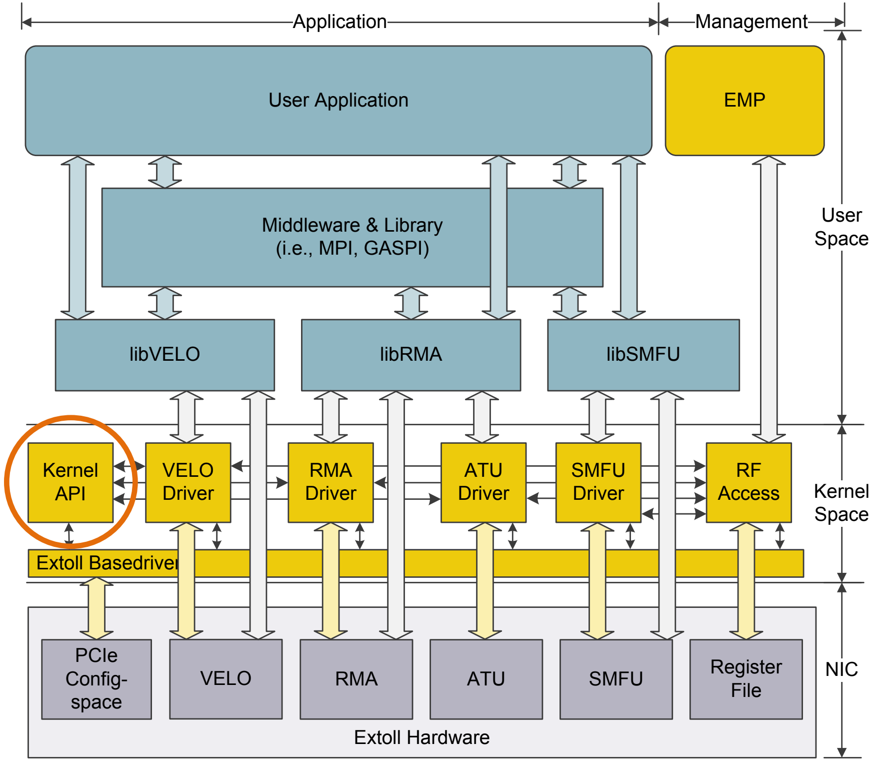
LP = Link Port NP = Network Port Req = Requester Resp = Responder Cmp = Completer

Extoll Interconnect
Functional Units



- VELO – Virtualized Engine for Low Overhead
  - Low latency two-sided messaging for small payloads
- RMA – Remote Memory Access
  - Supports *put* and *get* operations to transfers large amounts of data with one- and zero-copy methods
  - Local and remote completion notifications
- ATU – Address Translation Unit
  - Mapping of memory regions and page lists
- SMFU – Shared Memory Functional Unit
  - Distributed shared memory
  - Remote load/store operations

Extoll Interconnect
Software Environment



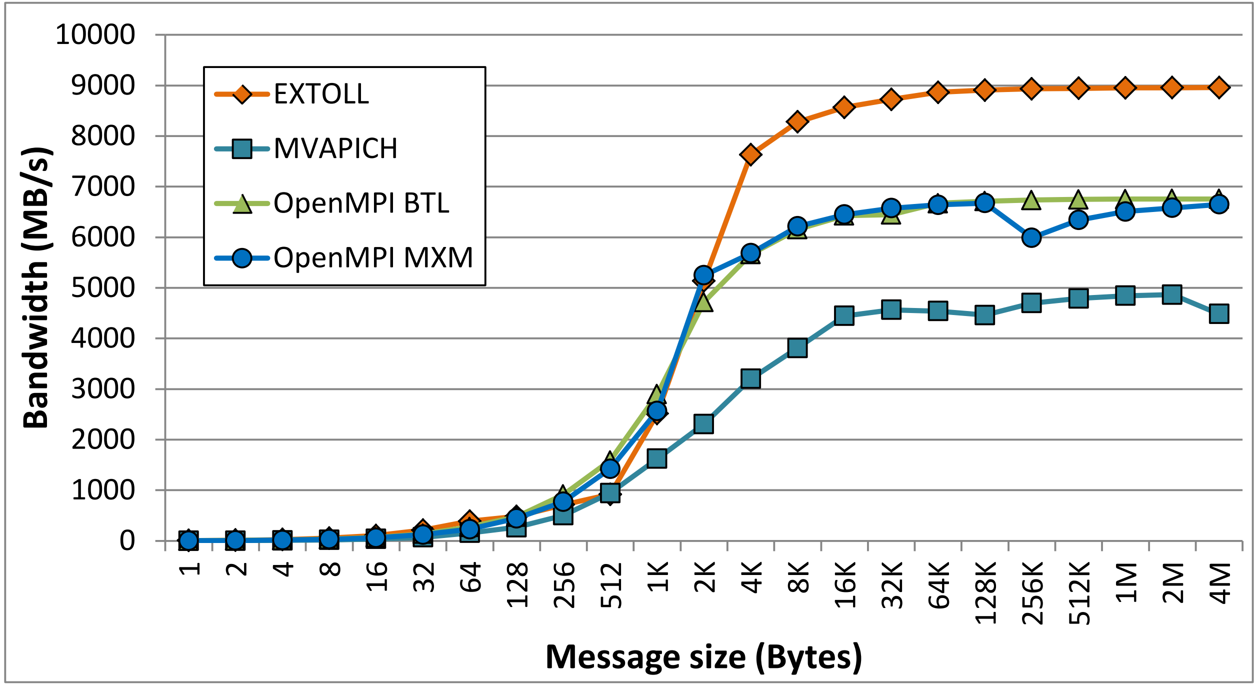
- Device driver and modules
  - Provide operating system bypass
  - Manage resources
- Kernel API provides interface for…
  - Network interface (EXN)
  - Direct sockets (AF_EXTL)
  - Network-attached accelerators (VPCI)
  - **Lustre Network Driver (EXLND)**
- Low-level user libraries
  - OpenMPI MTL for Extoll
- PGAS support (e.g., GASPI, GASNet)
- EMP management software

Extoll Tourmalet

Throughput Performance – RDMA Capability

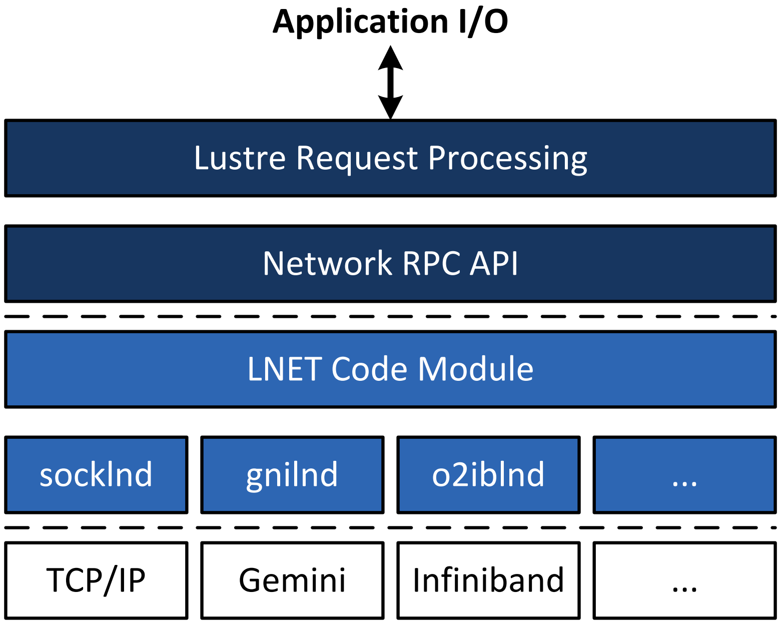
UNIVERSITÄT HEIDELBERG
ZUKUNFT SEIT 1386

- Two DL380 HP Servers each equipped with
  - EXTOLL Tourmalet Card
  - ConnectIB FDR Card
  - Intel(R) Xeon(R) CPU E5-2620 v3 @ 2.40GHz
  - 32 GB RAM
- Software:
  - CentOS 7.2
  - Mellanox OFED 3.0-1.01
  - EXTOLL SW Stack 1.3.1
  - OSU MPI Benchmark



LNET
Lustre Network Communication Protocol

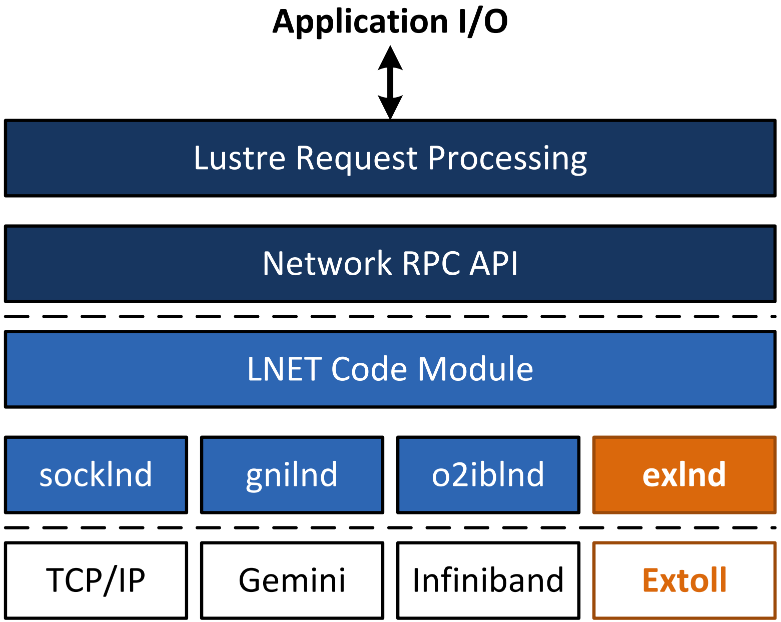
- LNET
  - Defines the communication infrastructure between Lustre clients and servers
  - Abstracts network details from Lustre
- Supports most network technologies
  - TCP/IP networks, Cray Gemini, Infiniband
  - RDMA for bulk data transfers
  - Provides routing between LNET networks
- Lustre Network Driver (LND)
  - Implements LNET-to-LND API
  - Provides support for a particular network type



EXLND Internals

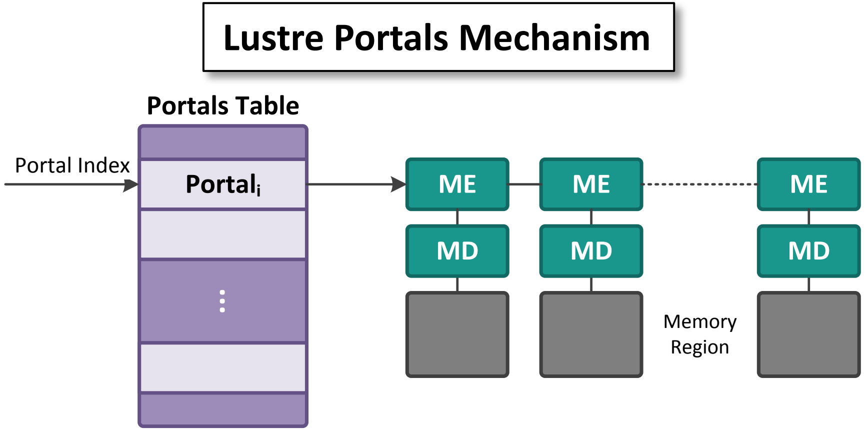
*EXLND – **Ex**toll **L**ustre **N**etwork **D**river*

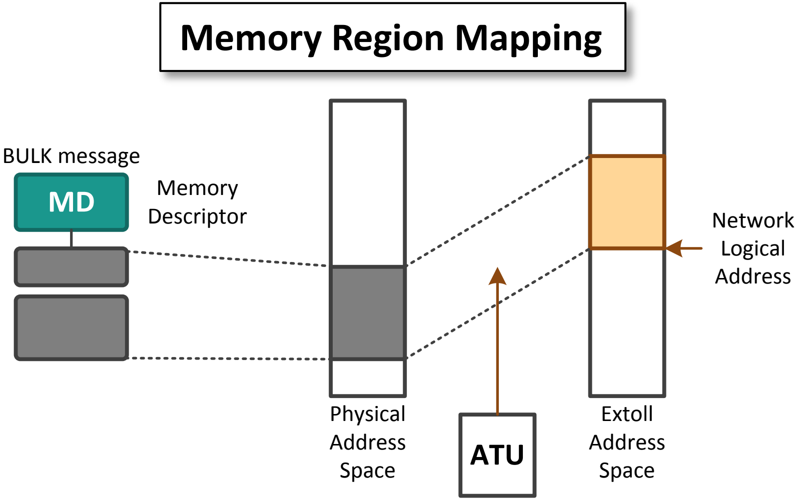
- Built upon Extoll Kernel API
- Data transfer protocols
  - Bulk transfers:
    - Rendezvous protocol: *RMA puts/gets*
    - Memory mapped via ATU
      → Network Logical Address (NLA)
  - Requests and immediate transfers:
    - Eager protocol: *VELO messages*
- Kernel module: **kexlnd.ko**
- Network name: **ex**
- Network adapter: `networks=ex(ex0)`



EXLND Internals

Resource Management – Memory Descriptors

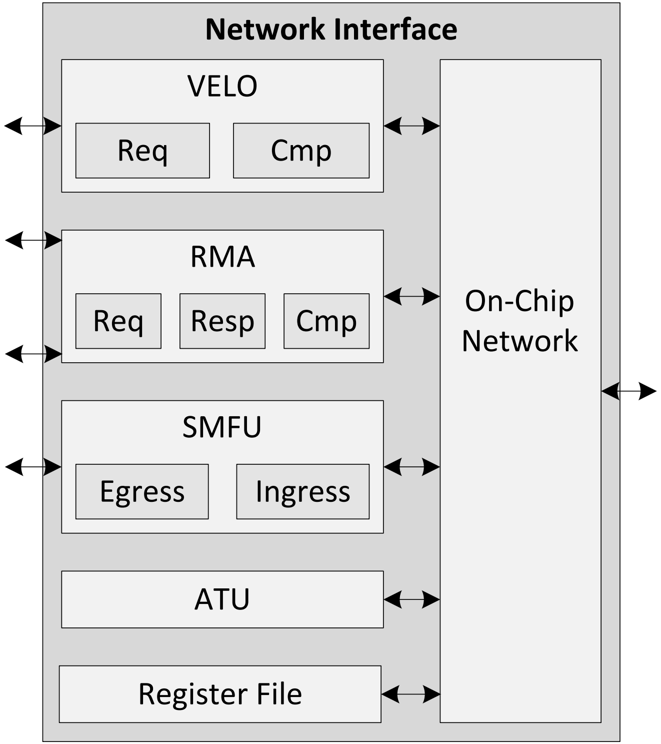




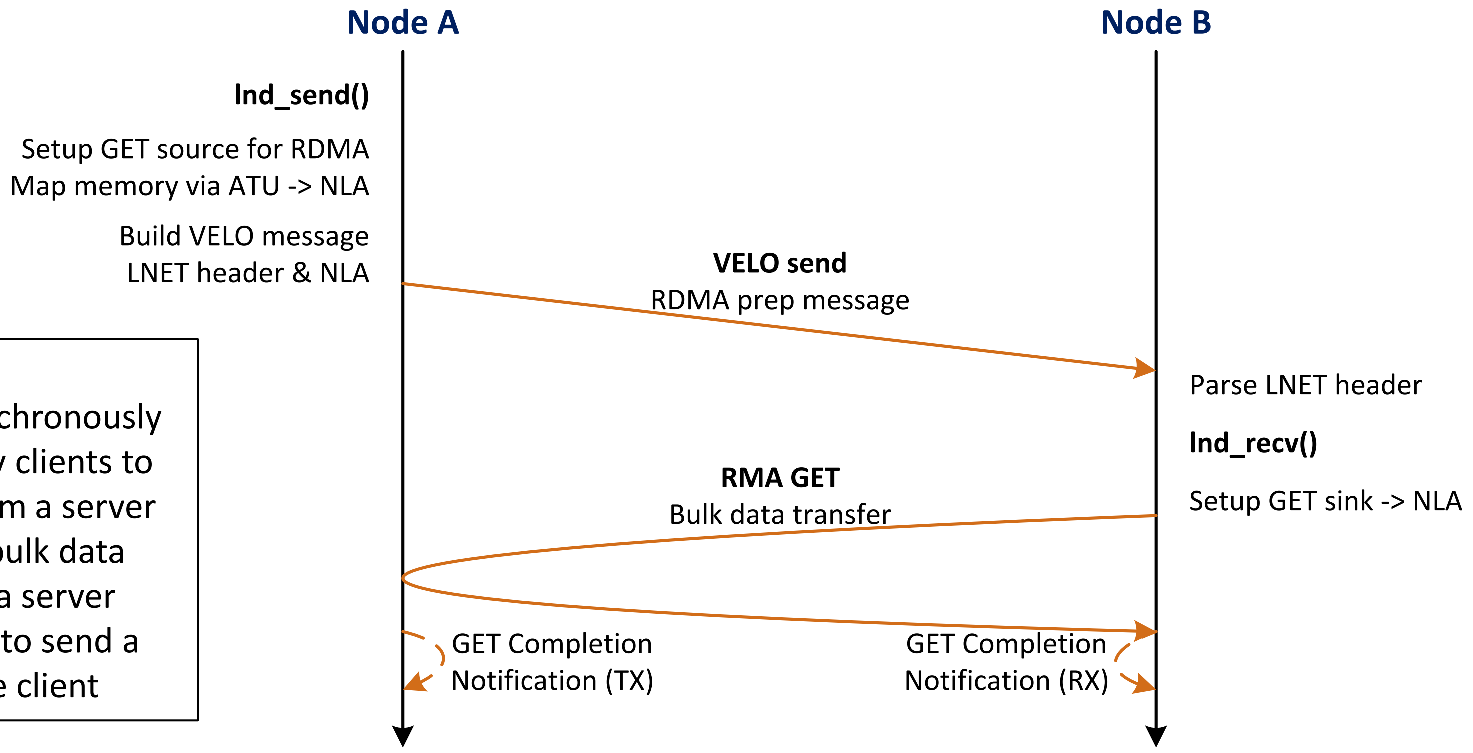
- Memory descriptors (MD) are
  - Typically either *struct kvec* or *page array*
  - Mapped via scatter/gather lists for RDMA (bulk data transfers)
- Idea: map scatter/gather lists via ATU into Extoll address space

EXLND Internals
Resource Management – Completions

- EXLND completions are…
  - Basically TX and RX descriptors
  - Distinguished by RMA sub-units
- Three different queues:
  - *Responder queue* handles LNetPut TX path
  - *Completer queue* handles both LNetPut RX and LNetGet TX
  - *Requester queue* handles LNetGet RX path
- Completions are matched with incoming RMA notifications



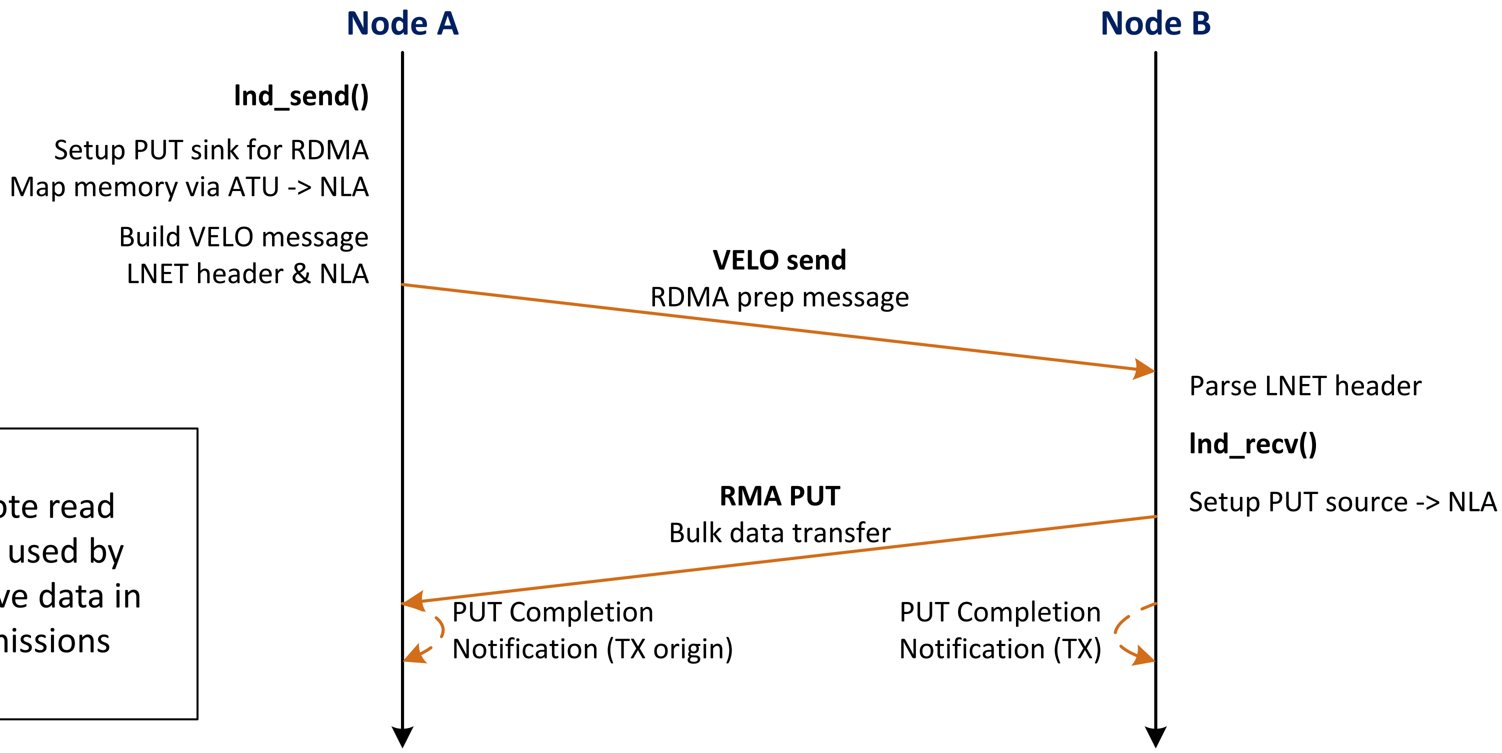
EXLND Internals
Bulk Data Transfer – LNetPut()



LNetPut():
- Sends data asynchronously
- Mostly issued by clients to request data from a server or to request a bulk data transmission to a server
- Used by servers to send a reply back to the client

EXLND Internals

Bulk Data Transfer – LNetGet()



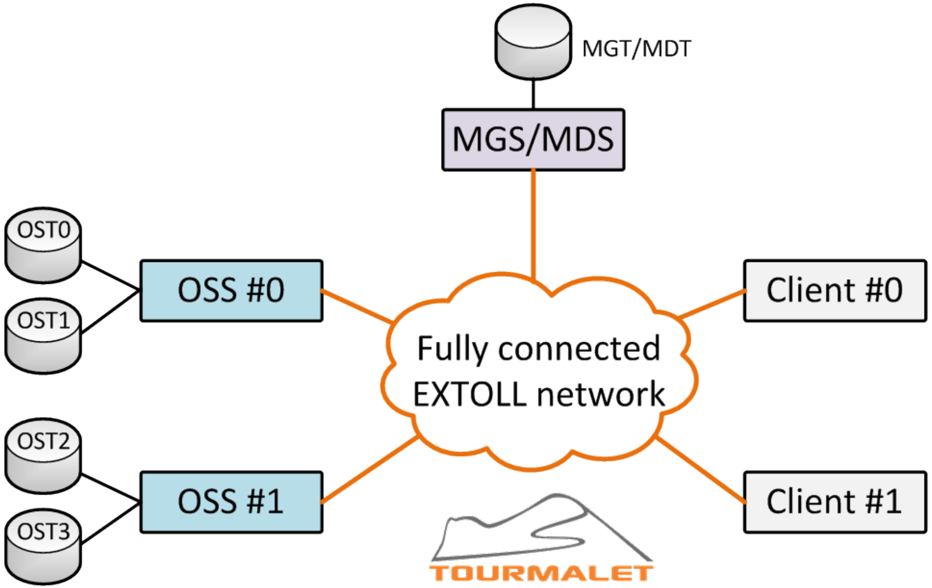
LNetGet():

- Serves as a remote read operation and is used by servers to retrieve data in bulk read transmissions from clients

EXLND Performance Evaluation

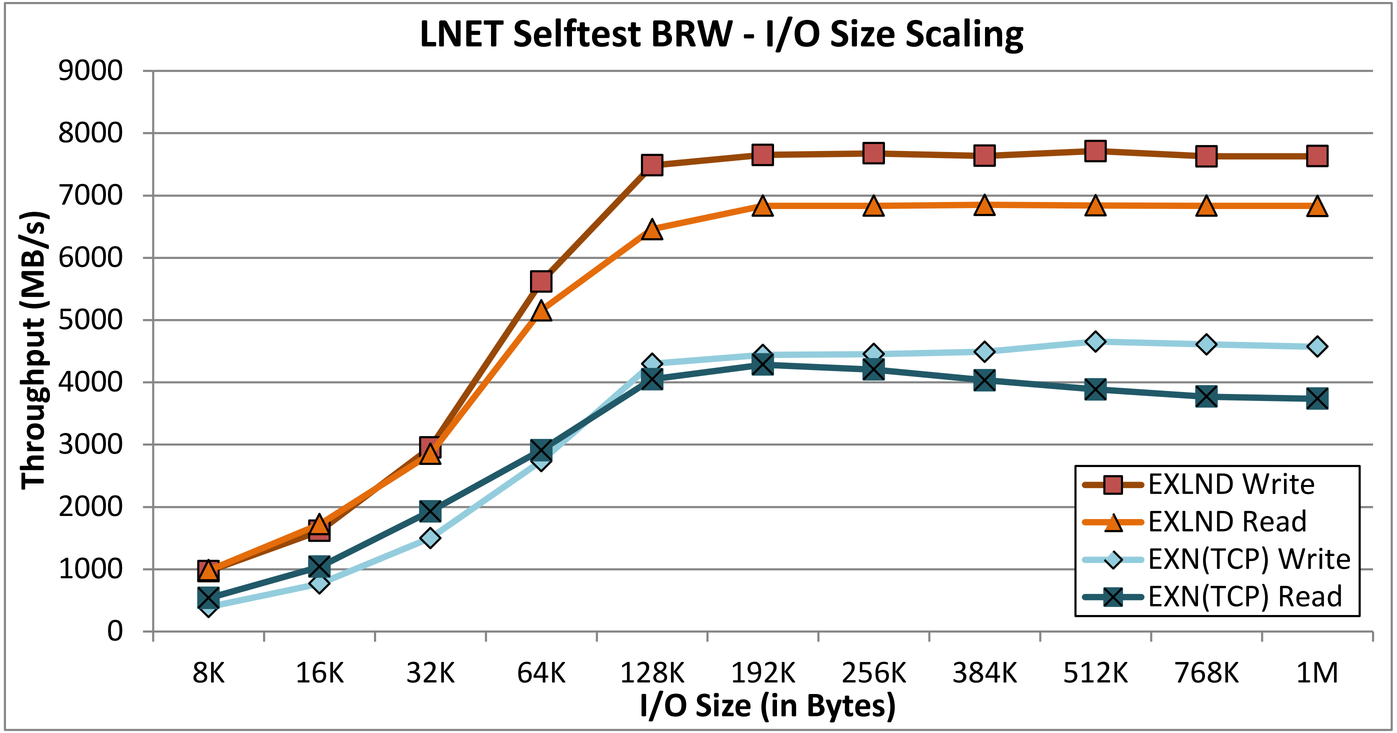
Test System and Lustre Configuration

- 5 *Supermicro Barebone SuperServer SYS-1019GP-TT* each equipped with:
  - Intel Xeon Silver 4110 2,1 GHz
  - 32 GB RAM
  - EXTOLL Tourmalet Card
  - OSS/MDS: Intel SSD 545s 128 GB
- CentOS 7.3
  - Lustre Patch 2.10.1
  - EXTOLL Software Stack v1.4.0
  - EXLND v0.2 (experimental)



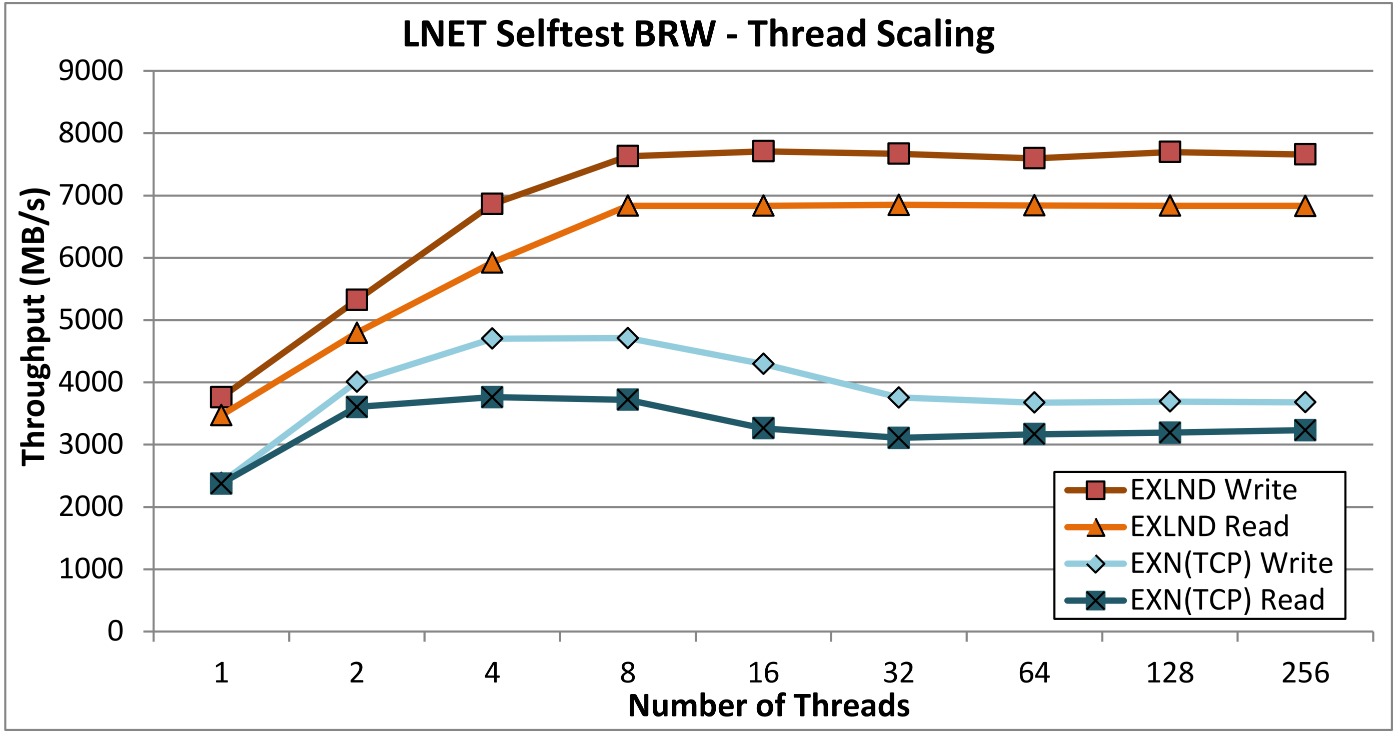
EXLND Performance Evaluation
LNET Selftest – I/O Size Scaling

UNIVERSITÄT HEIDELBERG
ZUKUNFT SEIT 1386



EXLND Performance Evaluation

LNET Selftest – Thread Scaling



EXLND Performance Evaluation
Other Benchmarks / Applications

UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

WORK IN PROGRESS

- Application performance: *IOR Benchmark*
  - Sequential read/write, file-per-process
  - Throughput limited by write/read speed of SSDs: ~350 MB/s per SSD
  - Need larger system (more OSSs/OSTs) for evaluation

WORK IN PROGRESS

- Metadata traffic performance: *mdtest*
  - Measure values like file creations/seconds or stat operations/seconds

#Tasks	1	2	4	8	16	32
File creation Ops/s	1782	3134	10161	17857	19925	25766
File stat Ops/s	6141	10046	18957	27391	38746	49356

EXLND Development Roadmap

- Finalize LND code
  - Perform code optimizations including
    - Improved scatter/gather I/O
    - Multiple scheduler threads
    - Default credit configuration and tunables
  - Handle remaining corner cases
  - Large scale stability tests
- Ensure compatibility with recent LNET changes
  - Currently supports latest Lustre 2.8 and 2.10 releases
- Push code to the Lustre community

Thank you for your attention.
Questions?