SAND2011-3969C

An Overview of Portals 4

Ron Brightwell
Scalable System Software
Sandia National Laboratories
Albuquerque, NM, USA

Sandia is a Multiprogram Laboratory Operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy Under Contract DE-ACO4-94AL85000.

NNSA National Nuclear Security Administration

Sandia National Laboratories

Portals Network Programming Interface

- Network API developed by Sandia, U. New Mexico, Intel
- Previous generations of Portals deployed on several production massively parallel systems
  - 1993: 1800-node Intel Paragon (SUNMOS)
  - 1997: 10,000-node Intel ASCI Red (Puma/Cougar)
  - 1999: 1800-node Cplant cluster (Linux)
  - 2005: 10,000-node Cray Sandia Red Storm (Catamount)
  - 2009: 18,688-node Cray XT5 – ORNL Jaguar (Linux)
- Focused on providing
  - Lightweight “connectionless” model for MPP systems
  - Low latency
  - High bandwidth
  - Independent progress
  - Overlap of computation and communication
  - Scalable buffering semantics
- Supports MPI, Cray SHMEM, ARMCI, GASNet, Lustre, etc.

What Makes Portals Different?

- One-sided communication with optional matching
- Provides elementary building blocks for supporting higher-level protocols well
- Allows structures to be placed in user-space, kernel-space, or NIC-space
- Allows for zero-copy, OS-bypass, and application-bypass implementations
- Scalable buffering of MPI unexpected messages
- Supports multiple higher-level protocols within a process
- Run-time system independent
- Well-defined failure semantics

Portals 4.0: Applying Lessons Learned from Cray SeaStar

- Allow for higher message rate
  - Atomic search and post for MPI receives required round-trip across PCI
  - Eliminate round-trip by having Portals manage unexpected messages
- Provide explicit flow control
  - Encourages well-behaved applications
    - Fail fast
    - Identify application scalability issues early
  - Resource exhaustion caused unrecoverable failure
  - Recovery doesn't have to be fast
  - Resource exhaustion will disable Portal
  - Subsequent messages will fail with event notification at initiator
  - Applies back pressure from network
    - Performance for scalable applications
    - Correctness for non-scalable applications

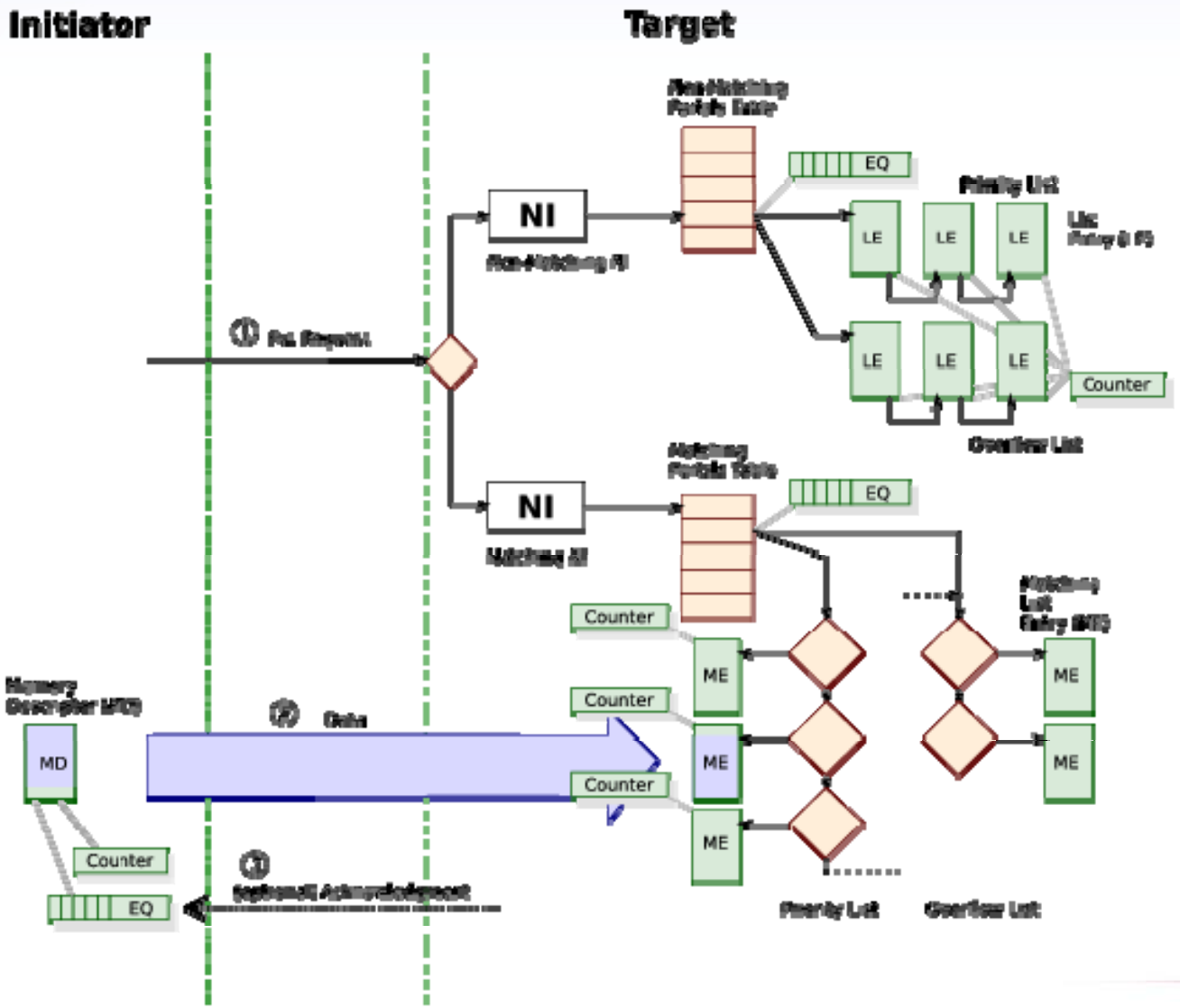
Portals 4.0 (cont'd)

- Enable a better hardware implementation
  - Designed for intelligent or programmable NICs
  - Arbitrary list insertion is bad
  - Remove unneeded symmetry on initiator and target objects
- New functionality for one-sided operations
  - Eliminate matching information
    - Smaller network header
    - Minimize processing at target
  - Scalable event delivery
    - Lightweight counting events
  - Triggered operations
    - Chain sequence of data movement operations
    - Asynchronous collective operations
      - Mitigate OS noise effects
    - Triggered rendezvous protocol
      - Enables progress without bandwidth penalty

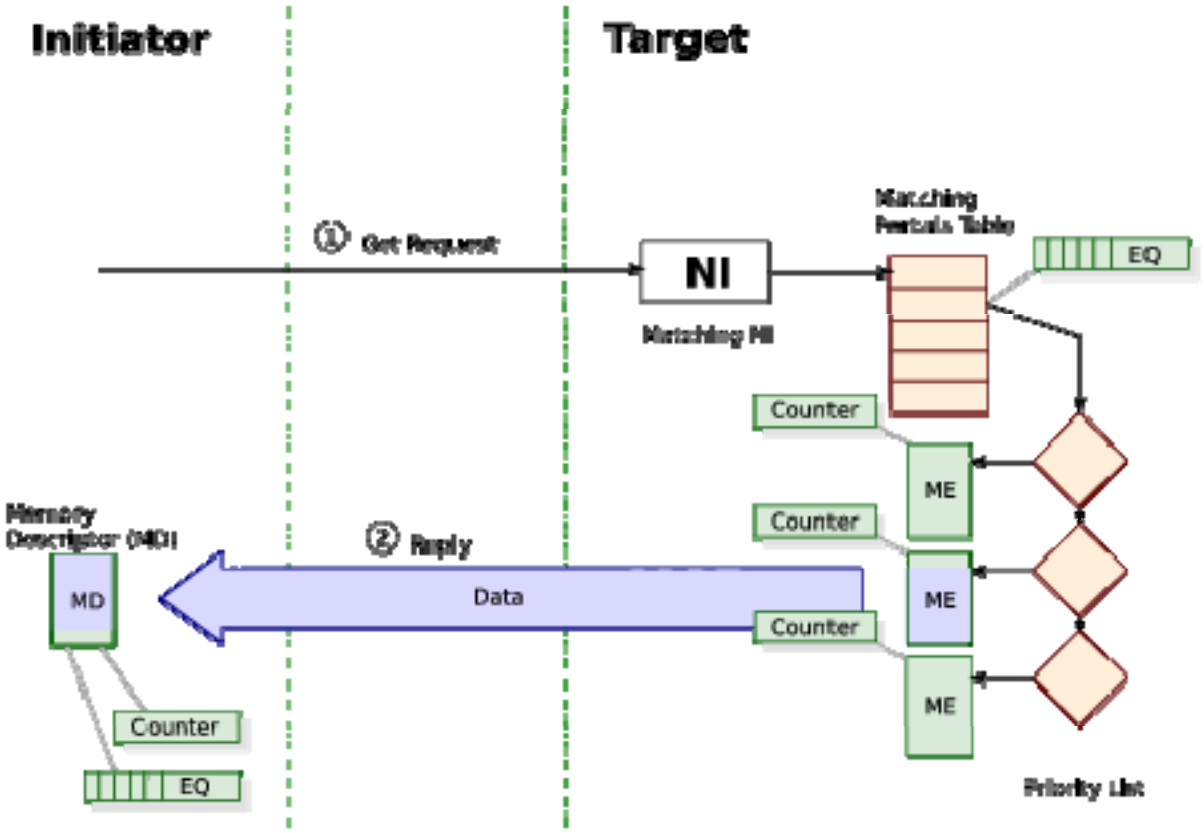
Portals 4 Implementations

- OpenFabrics Verbs
  - Provided by System Fabric Works
  - Provides a high-performance reference implementation for experimentation
  - Help identify issues with API, semantics, performance, etc.
  - Independent analysis of the specification
- Shared memory
  - Offers consistent and understandable performance characteristics
  - Provides ability to accurately measure instruction count for Portals operations
  - Better characterization of operations that impact latency and message rate
  - Evaluation of single-core onloading performance limits
- Structural Simulation Toolkit
  - Partial implementation for exploring NIC structures for offload

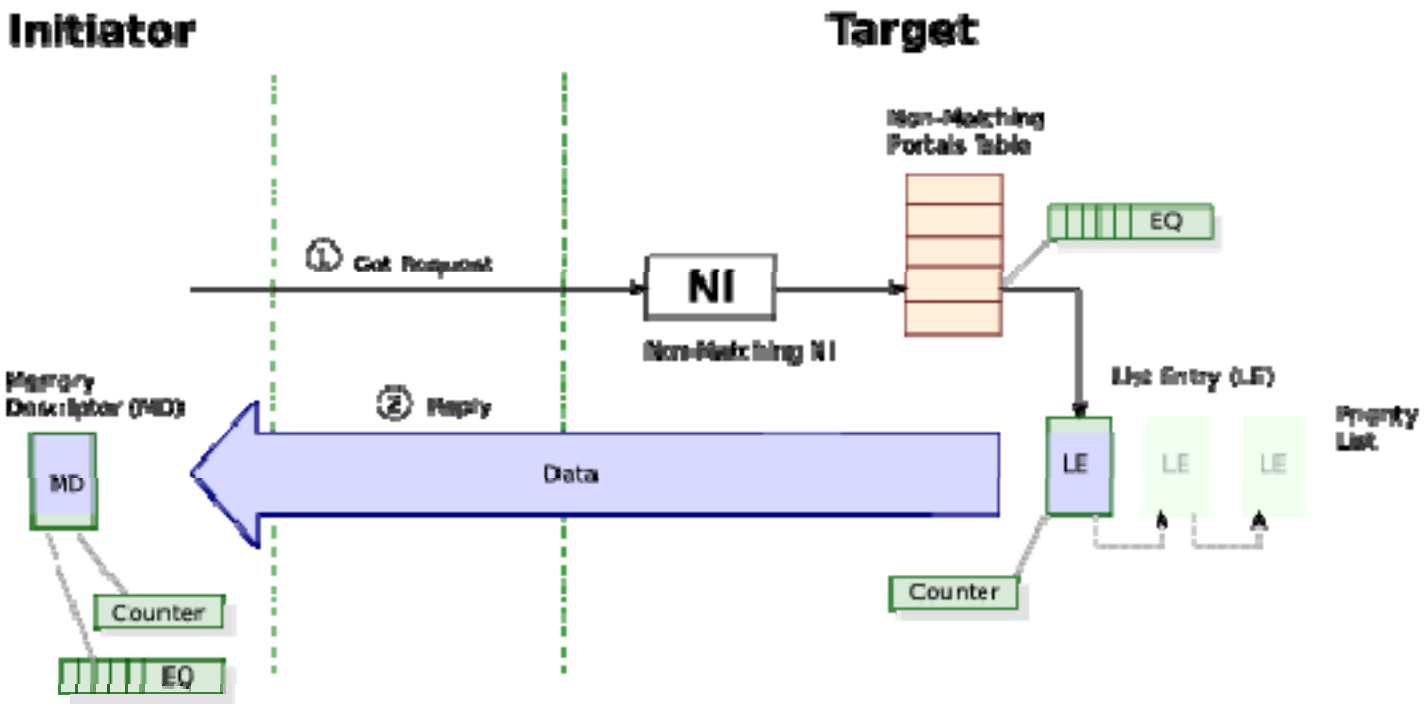
Put Operation



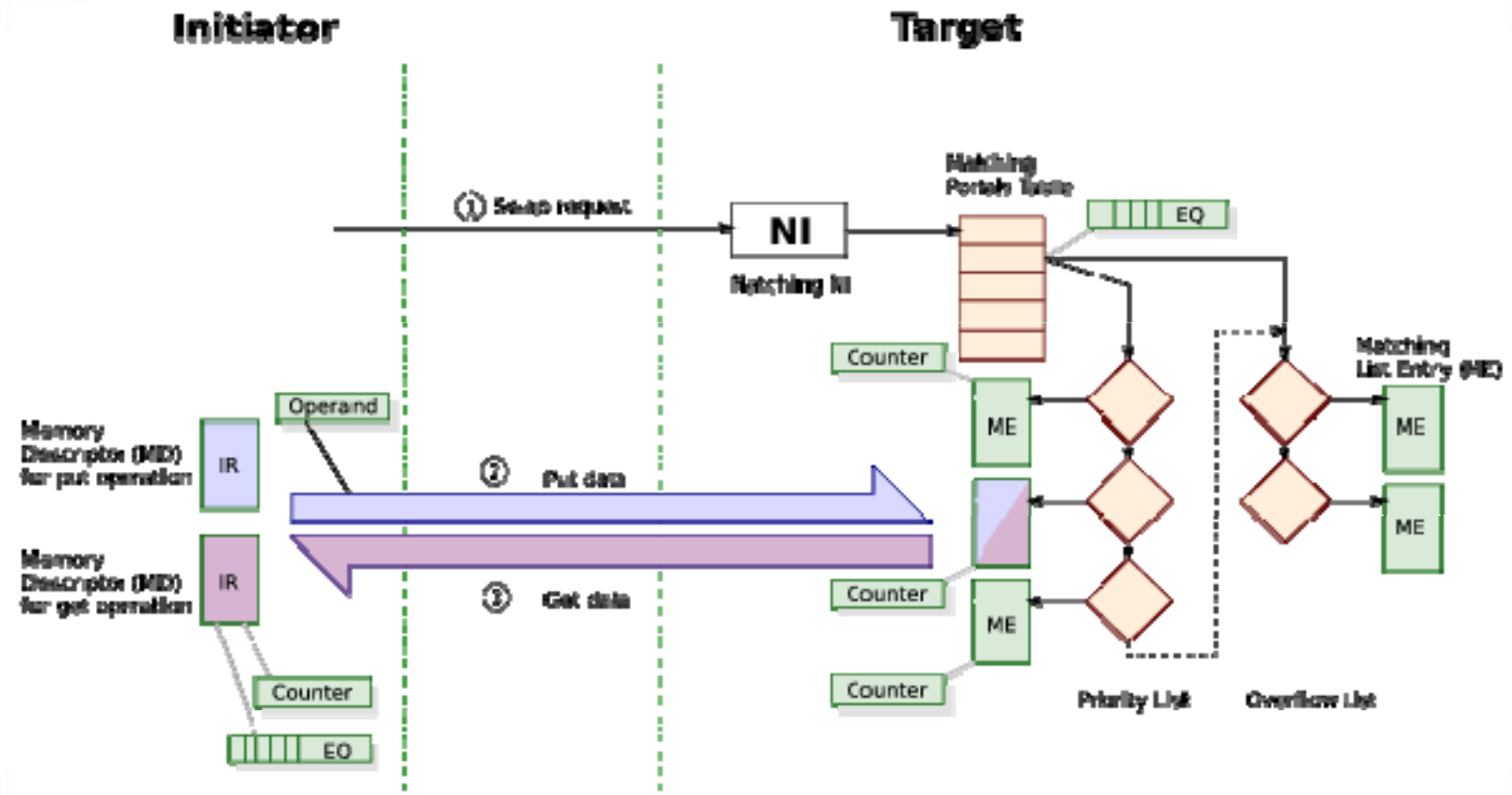
Matched Get Operation



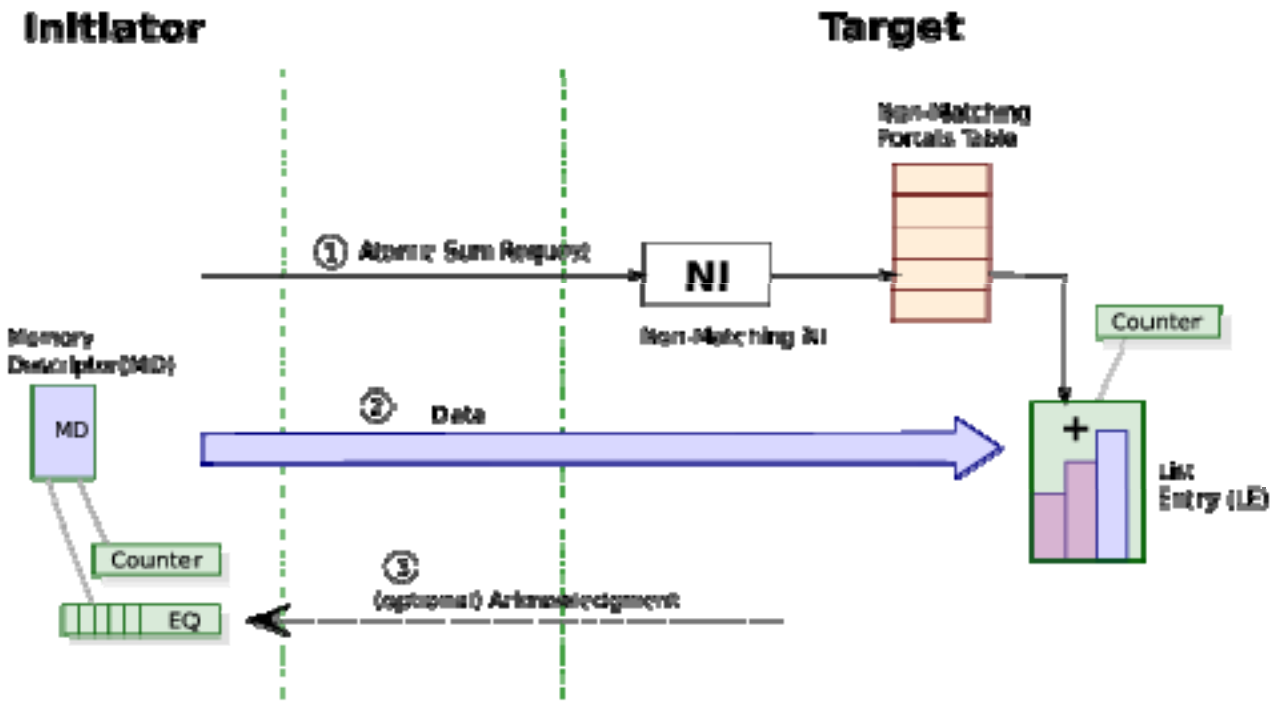
Unmatched Get Operation



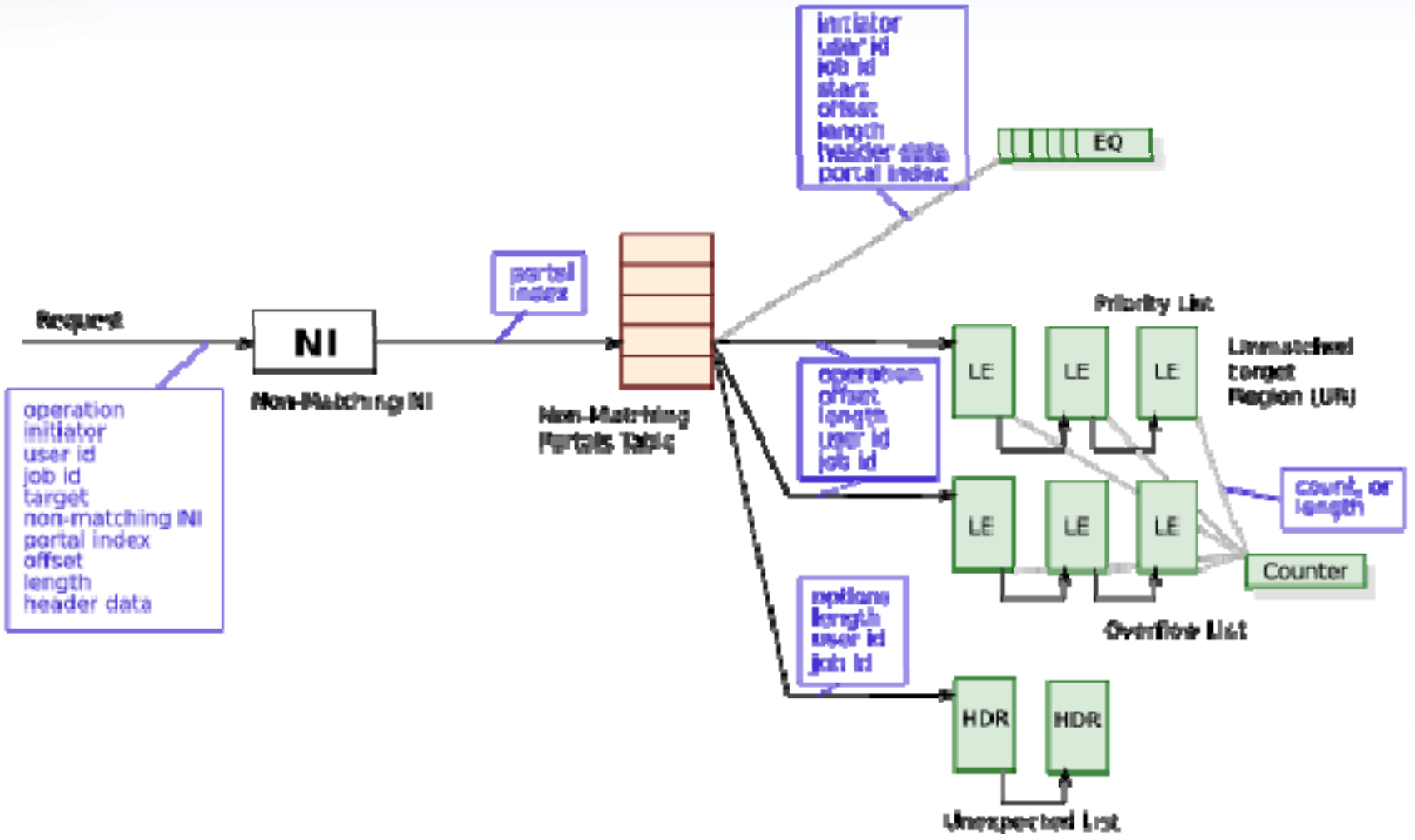
Atomic Operation (Swap)



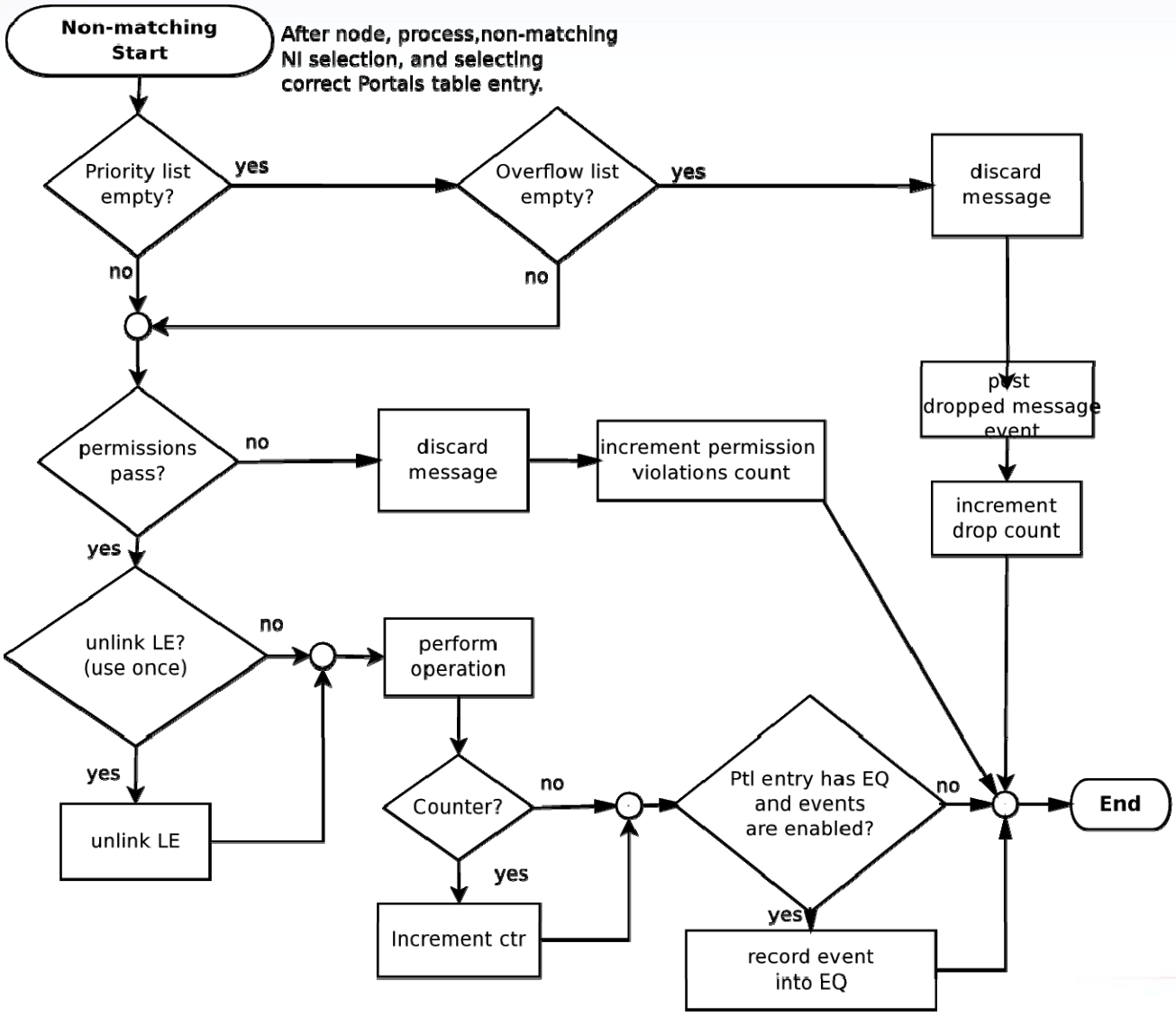
Atomic Operation (Sum)



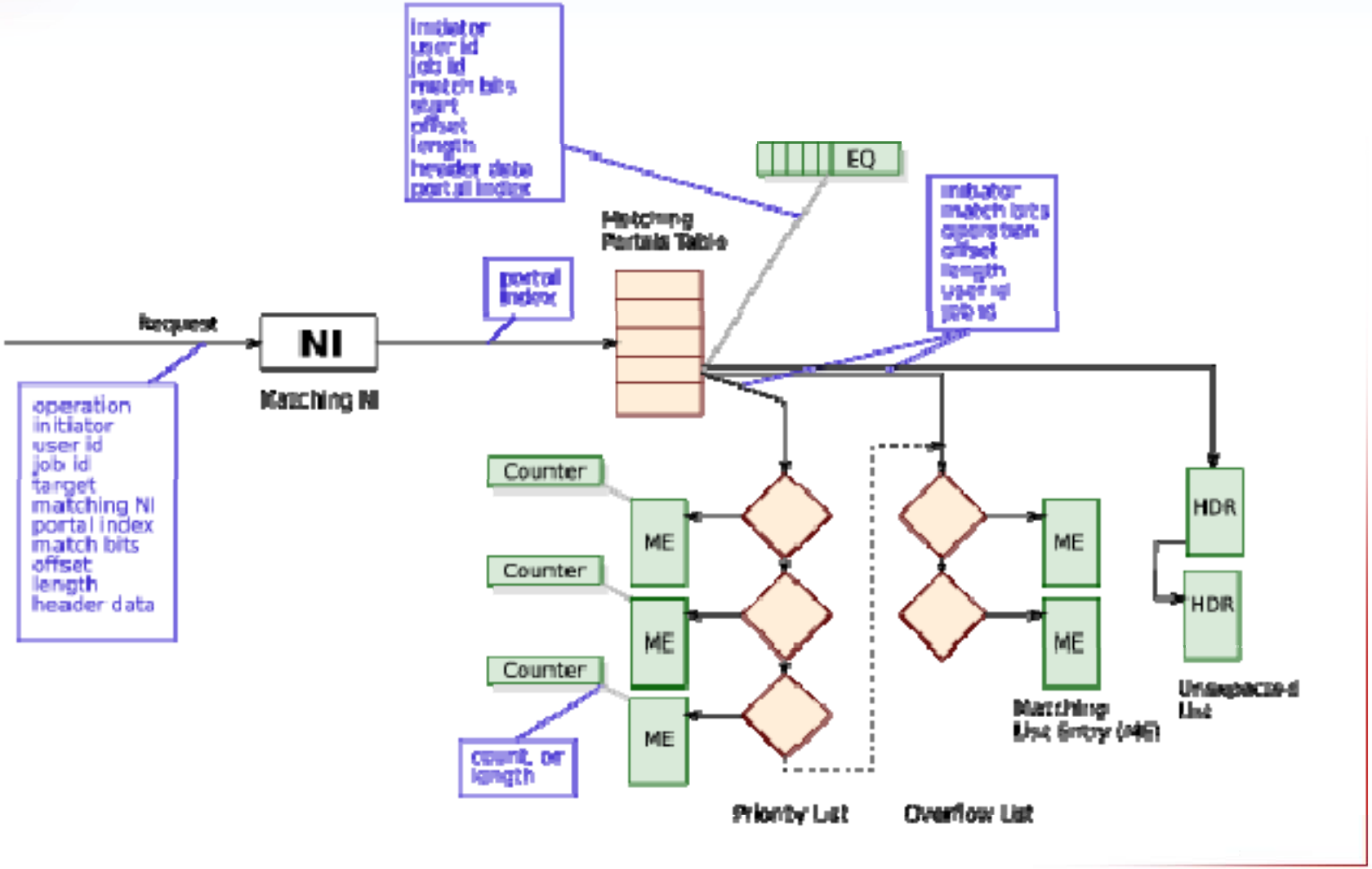
Non-Matching Address Translation



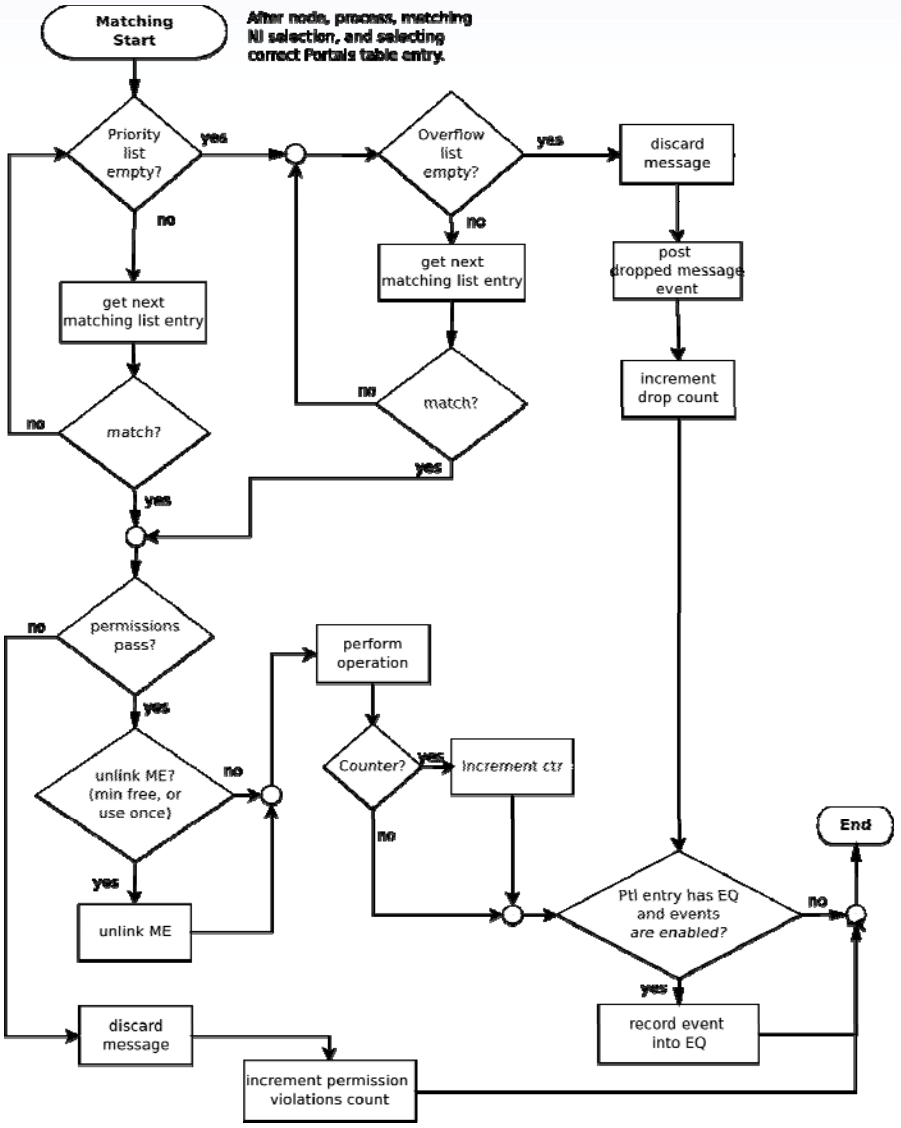
Non-Matching Address Translation



Matching Address Translation



Matching Address Translation



OpenSHMEM on Portals 4.0

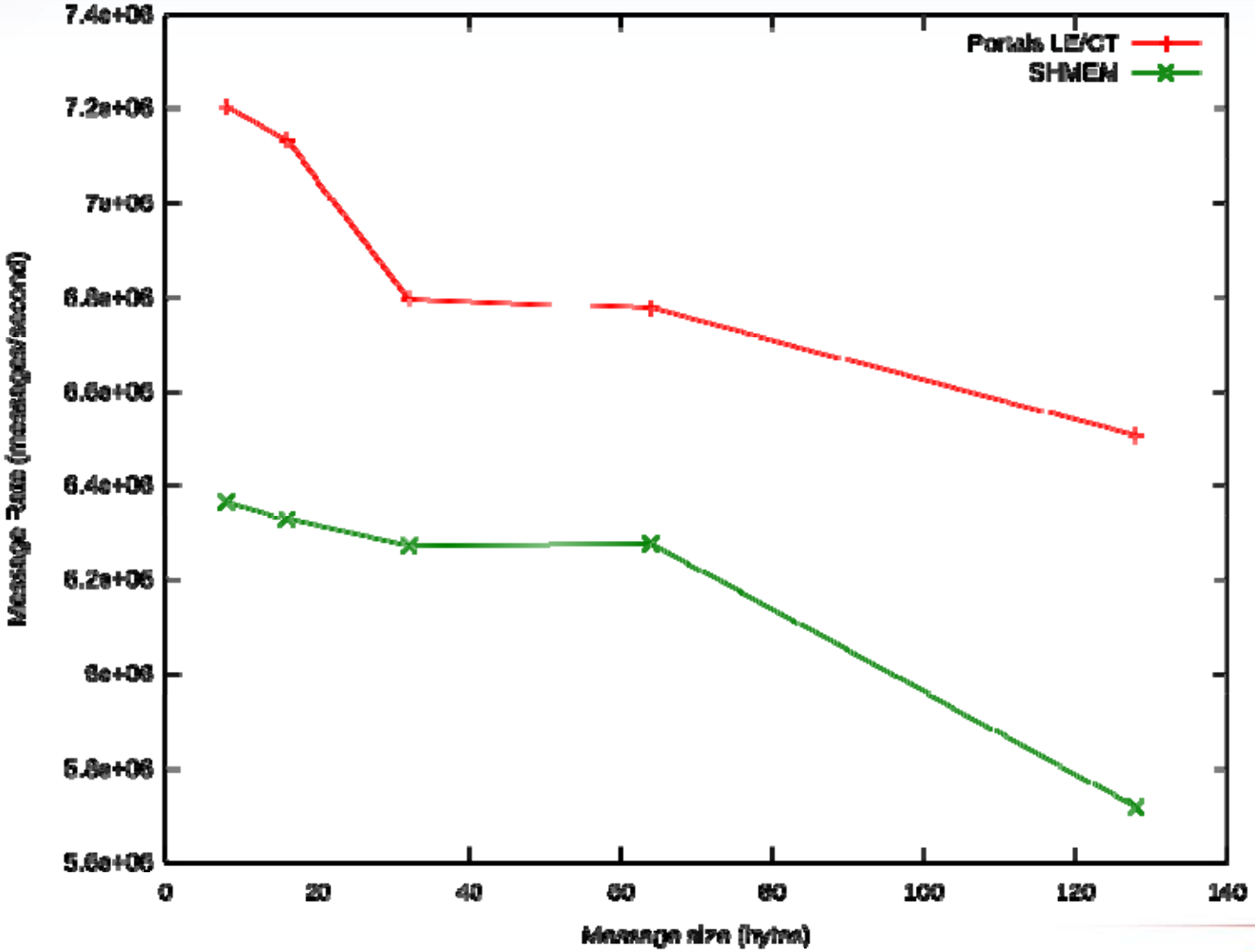
8-Byte Latency

	Latency
Portals LE/CT	0.43
OpenSHMEM	0.39

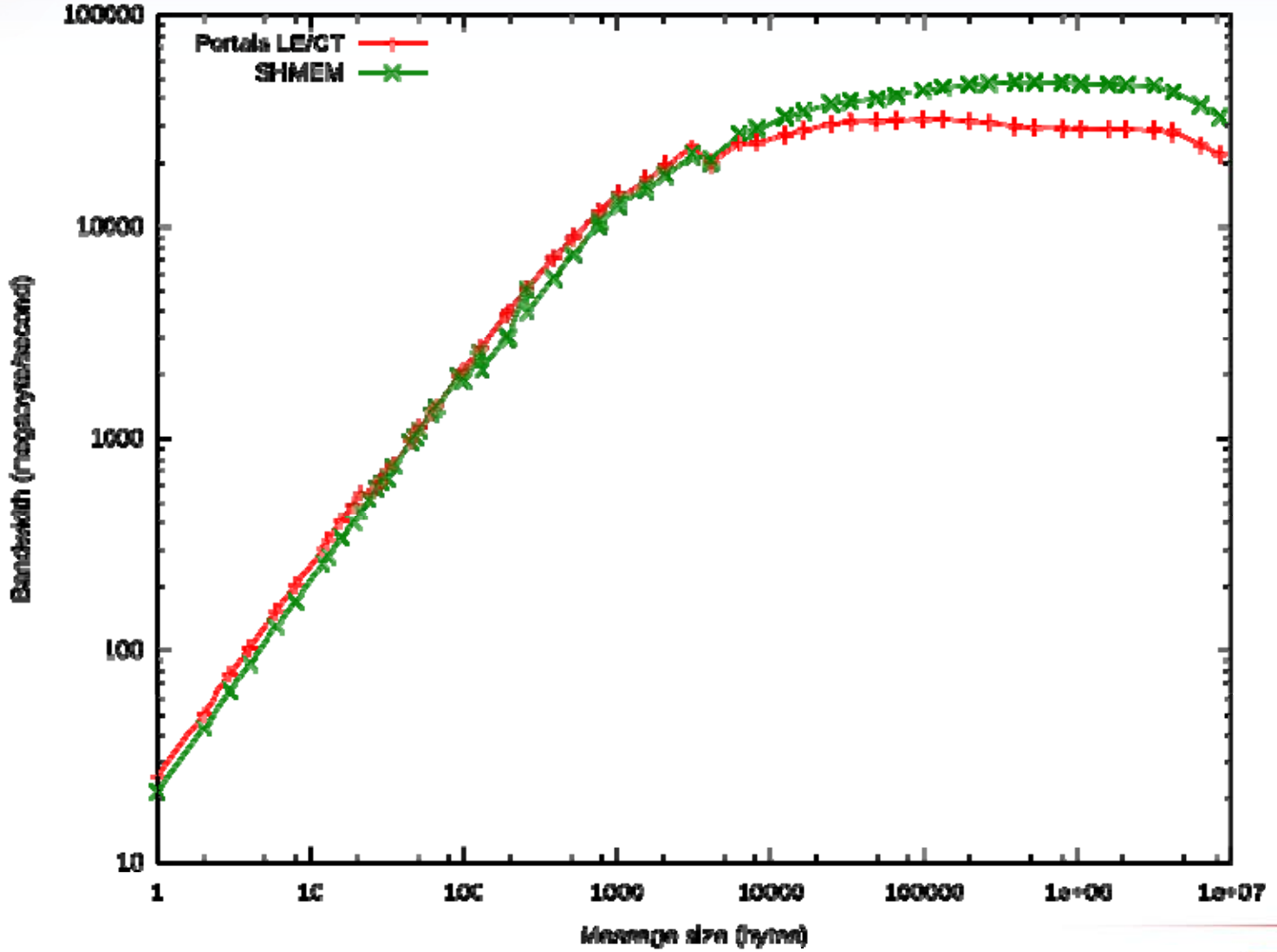
```
void
shmem_int_wait(int *var, int value)
{
   int ret;
   ptl_ct_event_t ct;

   while (*var == value) {
      ret = PtlCTGet(target_ct_h, &ct);
      if (PTL_OK != ret) { HANDLE_ERROER(); }
      if (*var != value) return;
      ret = PtlCTWait(target_ct_h,
                      ct.success + ct.failure + 1,
                      &ct);
      if (PTL_OK != ret) { HANDLE_ERROR(); }
   }
}
```

Unidirectional Message Rate

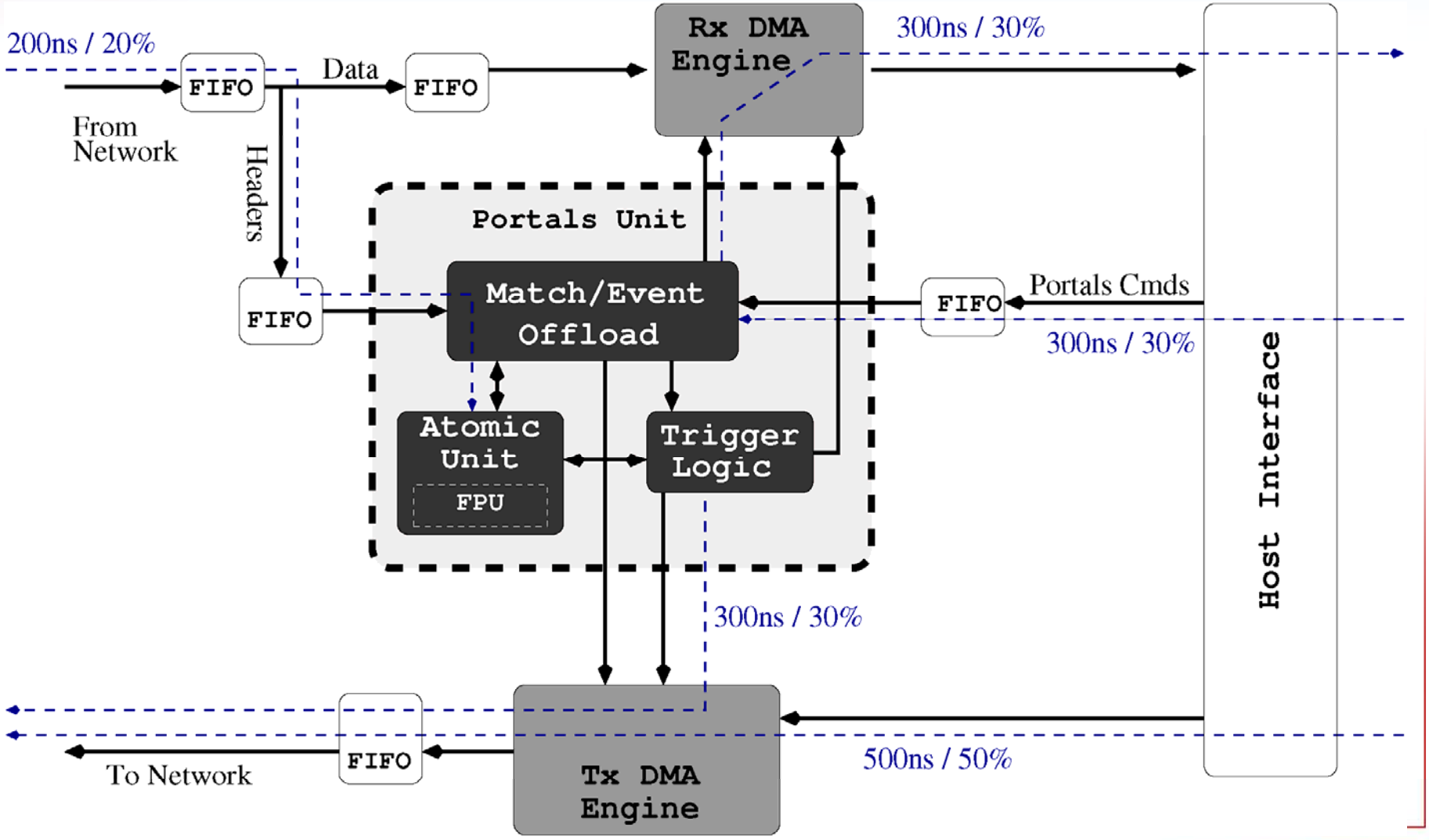


NetPIPE Bandwidth



Using Triggered Operations for Collective Communication

High-Level NIC Architecture



Simulation Settings

(a) simulation parameters

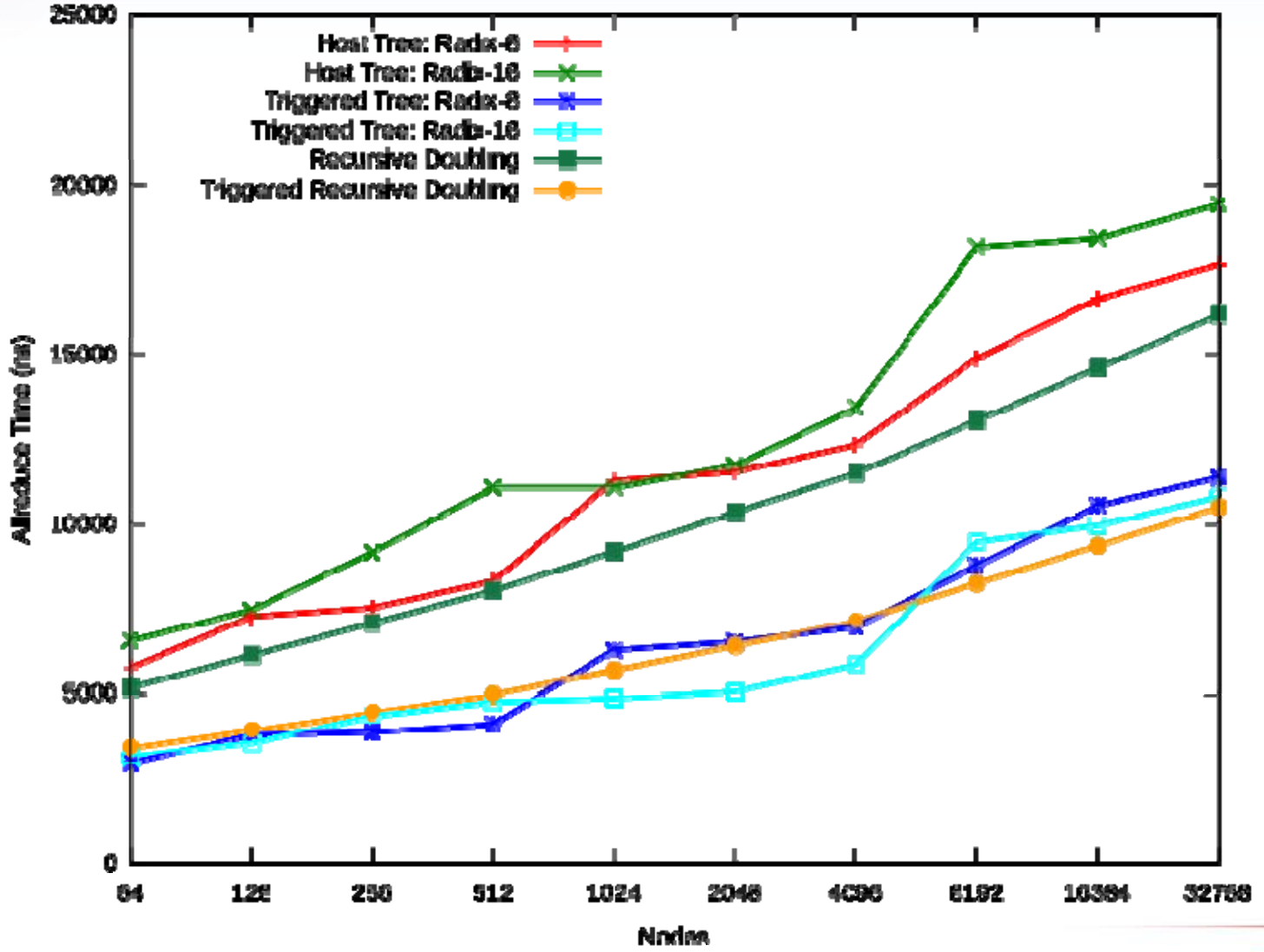
Property	Range
Msg Latency	500 ns, 1000 ns, 1500 ns
Msg Rate	5 Mmsgs/s, 10 Mmsgs/s
Overhead	$\frac{1}{MsgRate}$
NIC Msg Rate	62.5 Mmsgs/s
Rtr Latency	50 ns
Setup Time	200 ns
Cache Line	64 Bytes
Miss Latency	100 ns
Noise	250 ns @ 100KHz, 25 μs @ 1KHz, 2.5 ms @ 10Hz

(b) simulation configurations

	500 ns	1000 ns	1500 ns
5 Mmsgs/s		X	X
10 Mmsgs/s	X	X	

Allreduce

500ns, 10 Mmsgs/s

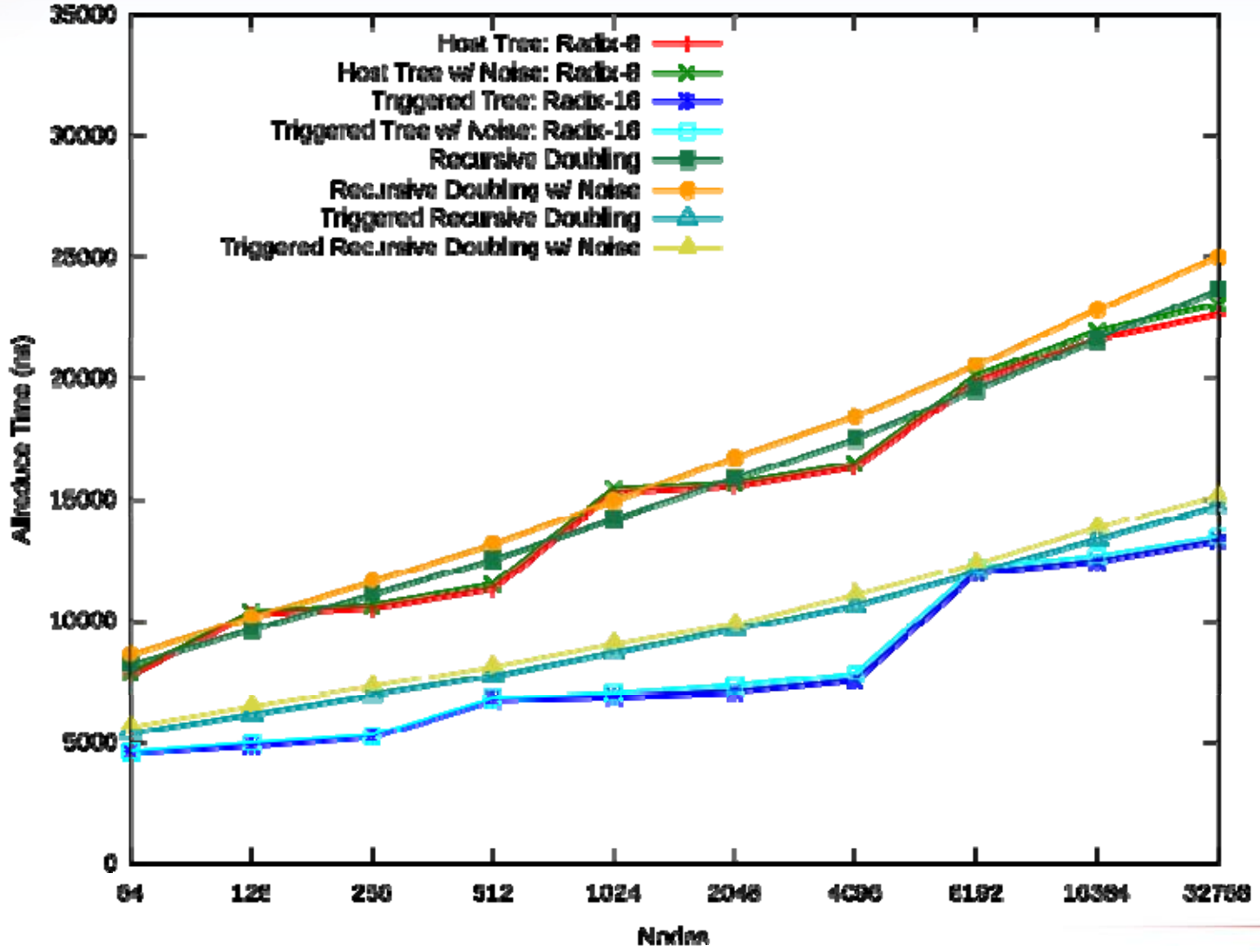


NNSA National Nuclear Security Administration

Sandia National Laboratories

Allreduce

250 ns @ 100KHz

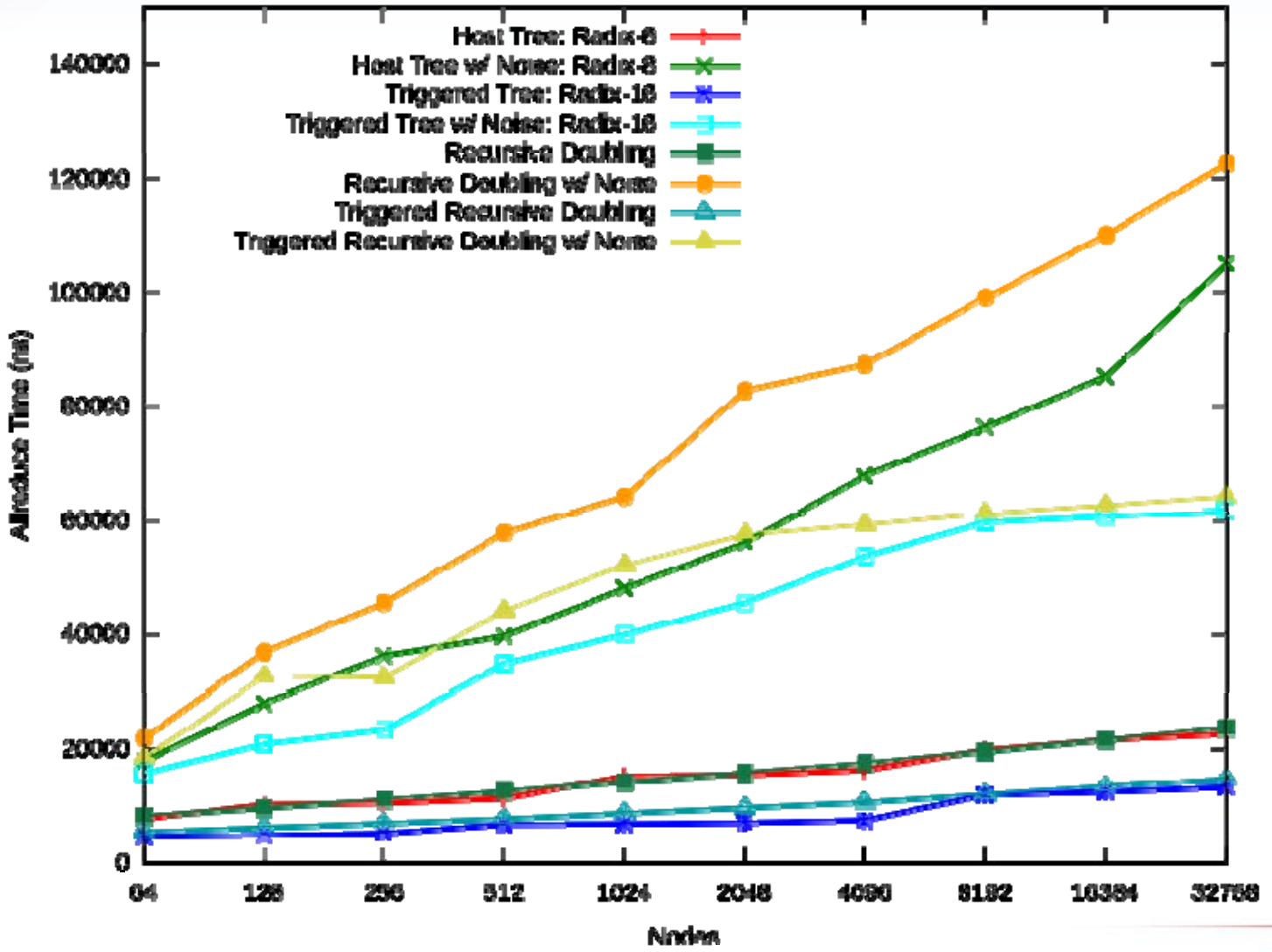


NNSA National Nuclear Security Administration

Sandia National Laboratories

Allreduce

25 us @ 1 KHz

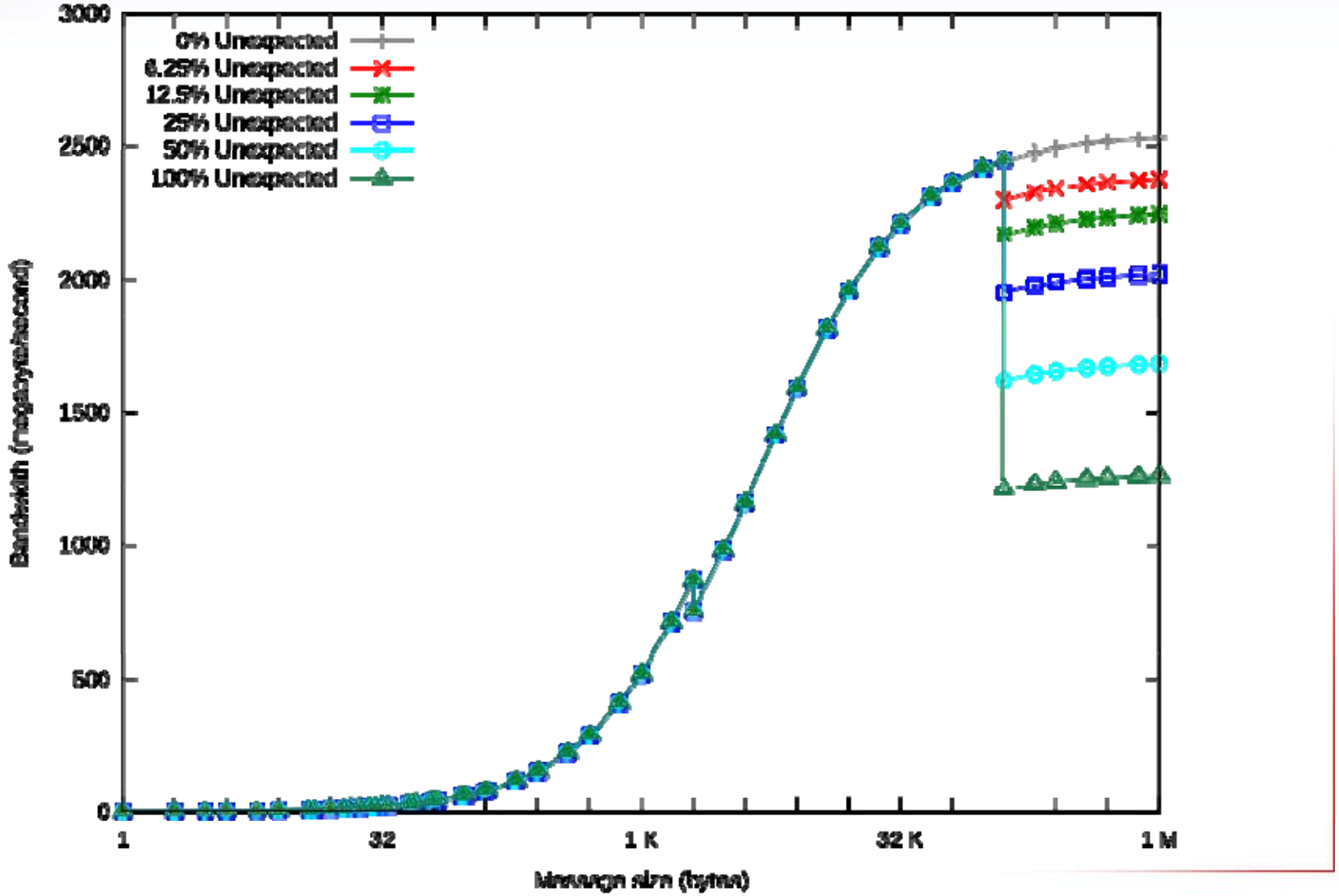


NNSA National Nuclear Security Administration

Sandia National Laboratories

Using Triggered Operations for an Offloaded Rendezvous Protocol

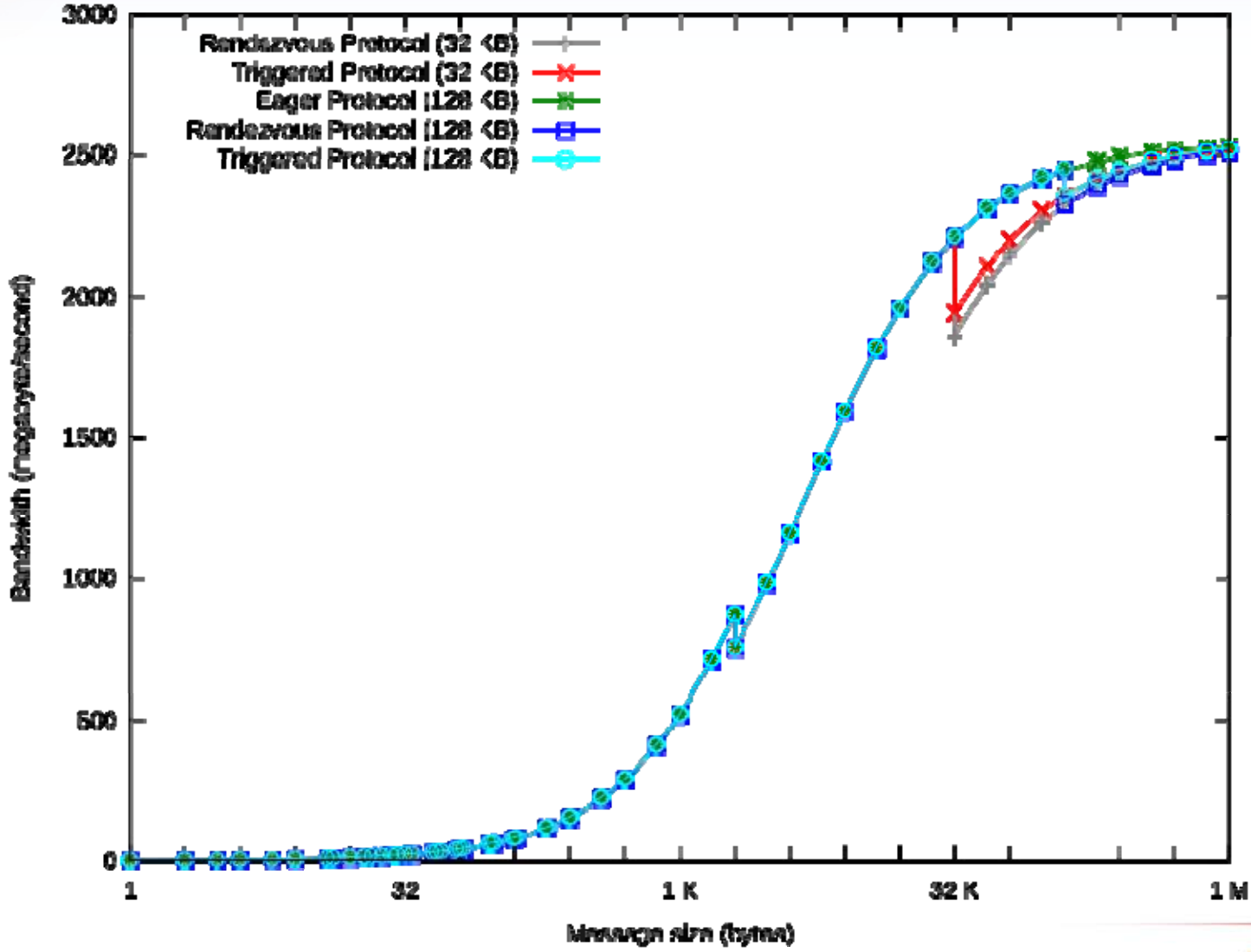
Ping-Pong Bandwidth



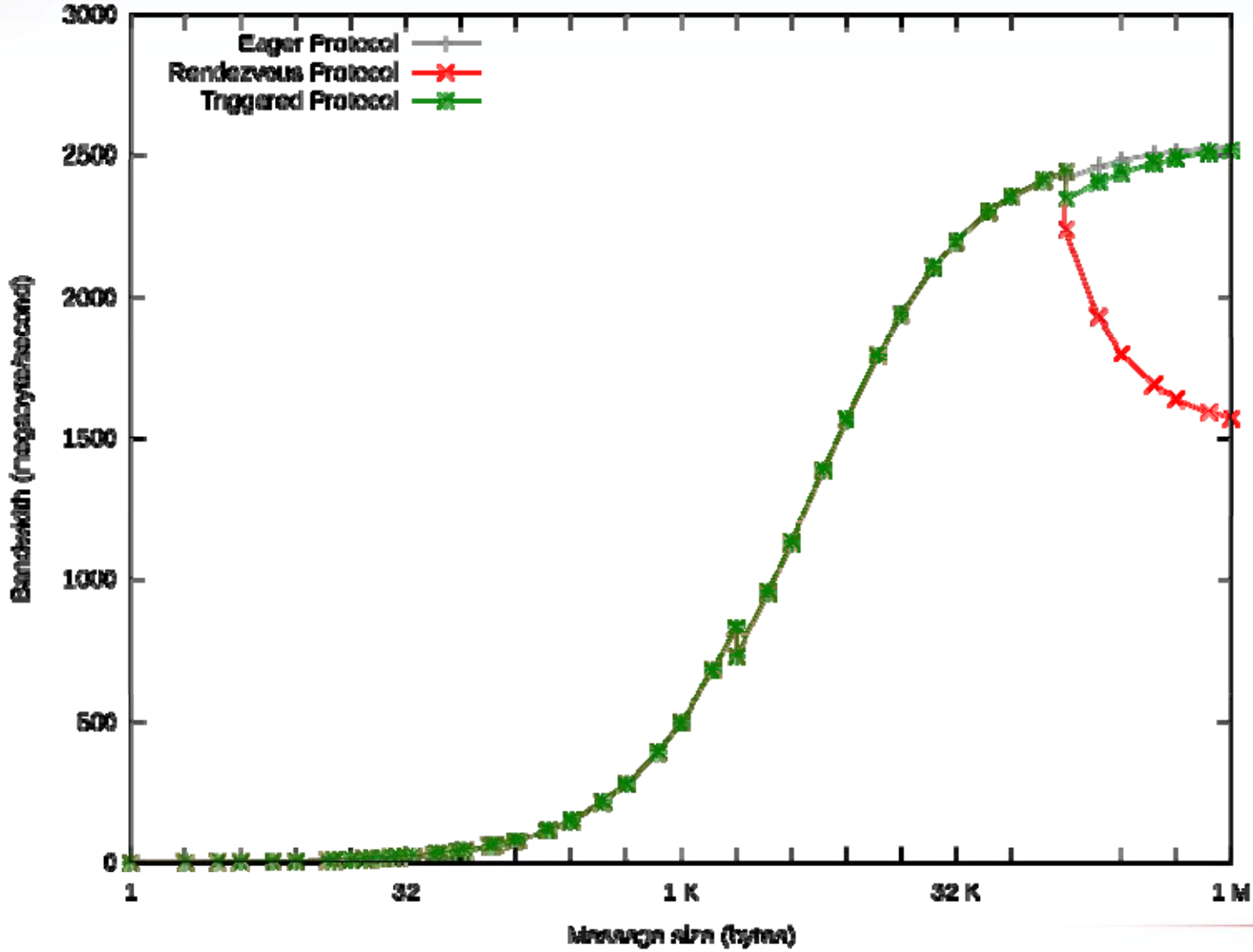
NNSA National Nuclear Security Administration

Sandia National Laboratories

Ping-Pong Bandwidth

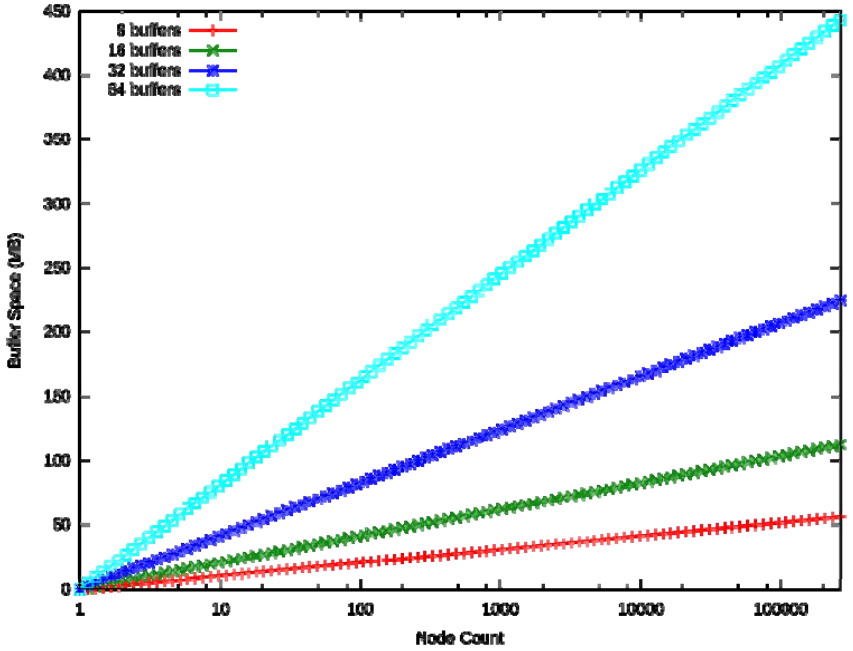


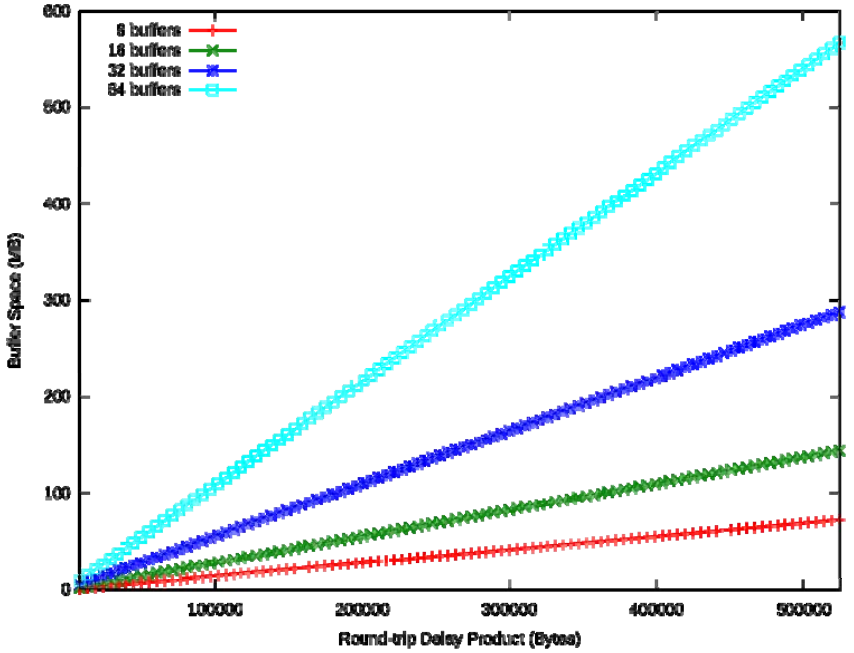
Ping-Pong Bandwidth



Eager Medium Protocol to Reduce Buffer Space

Buffer Space Scaling

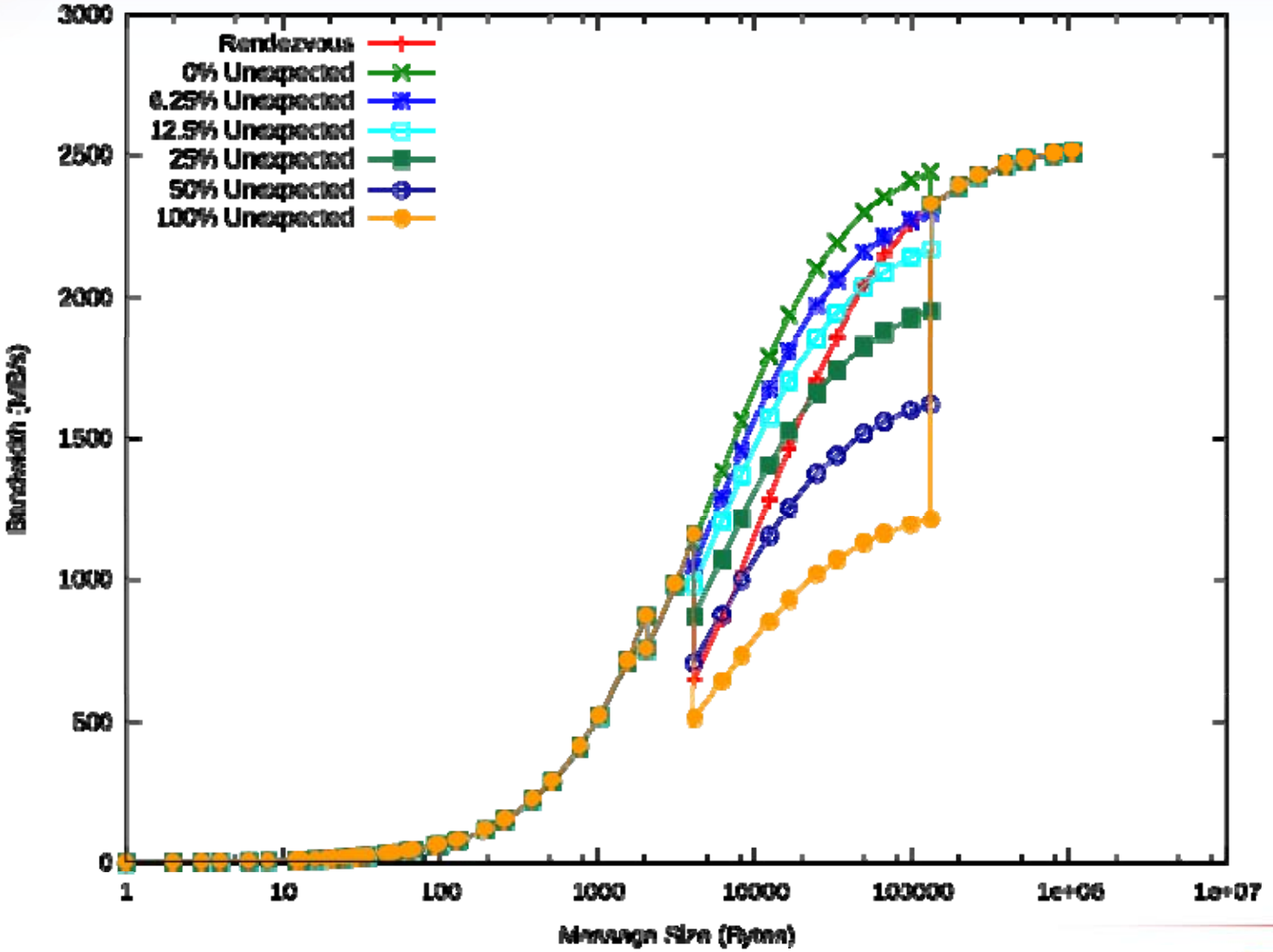




National Nuclear Security Administration

Sandia National Laboratories

Ping-Pong Bandwidth



Acknowledgments

- Sandia
  - Brian Barrett
  - Scott Hemmert
  - Michael Levenhagen
  - Kevin Pedretti
  - Kyle Wheeler
- Intel
  - Keith Underwood
  - Jerrie Coffman
  - Roy Larsen