OPENFABRICS ALLIANCE

iWARP Learnings and Best Practices

Author: Michael Fenn, Penn State
Date: March 28, 2012

Introduction

- Last year, the Research Computing and Cyberinfrastructure group at Penn State collaborated with Intel to produce some data on how iWARP performs in a real environment with real codes
- Results of that study are available at: http://rcc.its.psu.edu/education/white_papers/PennState_iWARP_WP_FINAL.pdf
- Today I'll be sharing what we learned along the way about running an iWARP network

Roadmap

- What is iWARP?
- Our Test Environment
- Networking Considerations
- iWARP Software Setup
- MPI Considerations
- Performance Observations
- Multi-Fabric Hosts
- Conclusions
- Acknowledgements

What is iWARP?

- Internet Wide Area RDMA Protocol
- Essentially, RDMA over TCP
  - thus allowing RDMA applications to work over an arbitrary TCP connection
- Ways to lower latency:
  - Kernel bypass drivers
  - Acceleration of the transport protocols
  - A low-latency, well-provisioned fabric

Our Test Environment

- Dell PowerEdge R710 servers
  - Two Xeon X5560 processors (2.80 GHz)
  - 48GB DDR3 1333 memory
  - Intel NetEffect 10Gb Ethernet Adapter
- Red Hat Enterprise Linux 5.6
- OFED 1.5.2
- Arista 7148SX 10Gb Ethernet Switch

The Fabric

- For good performance in RDMA applications, you need a low-latency Ethernet switch
- Our 7148SX has 1.2µs latency
  - It's a couple years old, newer switches are well into the 100's of ns
- Dropped packets and retransmissions kill performance
  - Use flow control: 802.3x if you must, but 802.1Qbb if you can
  - Better yet, design a fully non-blocking network

Non-blocking Ethernet Fabrics

- Is the dream yet a reality?
- With a small number of hosts, this is easy:
  - < 48, use a fixed-port switch
  - < ~384, use a chassis switch with non oversubscribed line cards
- Beyond that?
  - That darn spanning tree gets in the way of designing a true fat tree fabric
  - Transparent Interconnect of Lots of Links (TRILL) seems to be the solution, but so far implementations are proprietary

Jumbo Frames

- If your network doesn't support Jumbo frames yet, don't worry
- VASP results:
  - iWARP without jumbo frames: 52.30 minutes
  - iWARP with jumbo frames: 51.26 minutes
  - Surprisingly (at least to me) 1500 byte frames are only about 2% slower than 9000 byte frames in message passing applications
- My theory is that these classes of application are more latency-sensitive than bandwidth-sensitive
- This was largely borne out in our larger comparison between IB and iWARP

iWARP Software Setup

- BIOS setup similar to other high-performance RDMA networks (IB, RoCE)
  - Disable C-states
  - Disable PCIe link power management
- Increase memlock ulimits
- Need to use a recent OFED
  - RHEL 5's bundled OFED is too old

More Software Setup

- The iw_nes driver needs some extra parameters in /etc/modprobe.conf:

```
options iw_nes nes_drv_opt=0x110
options rdma_cm unify_tcp_port_space=1
alias eth2 iw_nes
install iw_nes /sbin/sysctl -w net.ipv4.tcp_sack=0
> /dev/null 2>&1; /sbin/modprobe --ignore-install
iw_nes
```

- Needing to keep track of which eth* device the NetEffect card is can somewhat complicate deployments on diverse hardware

More Software Setup

- The file /etc/dat.conf lets you pass parameters to uDAPL providers
- The OpenIB-iwarp and ofa-v2-iwarp need to be modified to refer to the NetEffect eth* device

Optional TCP Tuning

- These sysctl parameters control the behavior of the Linux TCP stack, but don't affect the hardware TCP engine in the NetEffect:

```
net.ipv4.tcp_timestamps=1
net.ipv4.tcp_sack=0
net.ipv4.tcp_rmem=4096 87380 4194304
net.ipv4.tcp_wmem=4096 16384 4194304
net.core.rmem_max=131071
net.core.wmem_max=131071
net.core.netdev_max_backlog=1000
net.ipv4.tcp_max_syn_backlog=1024
net.ipv4.tcp_window_scaling=1
net.core.rmem_default=126976
net.core.wmem_default=126976
net.core.optmem_max=20480
```

MPI Implementations

- iWARP is well-supported by popular MPI implementations
  - OpenMPI
  - MVAPICH2
  - Intel MPI
  - Platform MPI (neé HP-MPI)
- We used OpenMPI and HP-MPI in our testing

OpenMPI

- Example arguments to mpiexec
- `mpiexec -np 16 -machinefile mymachines --mca btl_openib_flags 2 --mca btl_mpi_leave_pinned 0 --mca btl_openib_max_inline_data 64 --mca btl openib,self,sm …`
- Notice we are using the verbs interface here
- We found that OpenMPI over the verbs interface works well with the NetEffect adapter
- Setting the btl_openib_flags to 2 allows the sender to use RDMA writes on the NetEffect adapter
- Also leaving memory pinned, adjusting some buffer sizes

HP-MPI

- HP-MPI (now Platform MPI) is a very popular MPI implementation for ISV's to use in their software products due to its stable ABI
- HP-MPI uses the uDAPL interface to communicate with the NetEffect card
- `mpirun -UDAPL -prot -intra=shm -e MPI_HASIC_UDAPL=ofa-v2-iwarp -1sided`
- The MPI_HASIC_UDAPL environment variable controls which uDAPL provider to use (set back in /etc/dat.conf
- However …

More HP-MPI

- /etc/dat.conf parsing is completely broken!
- This was a real head scratcher until we realized that HP-MPI doesn't actually parse the providers in /etc/dat.conf, it just picks the first one!
- The "solution" is to make sure that the provider you want to use is the first (or only) line in the file

Performance Observations

- With our 7148SX, 7.5µs IMB PingPong latency
  - 2.4µs is attributable to going through the 7148SX twice
- Application codes scaled well, (within the limits of our environment and benchmark)
  - VASP (OpenMPI)
  - Quantum Espresso Plane Wave (OpenMPI)
  - WRF (OpenMPI)
  - Abaqus (HP-MPI)
  - LAMMPS (OpenMPI)
  - MPPdyna (OpenMPI)

More Performance Observations

- Sometimes we did notice performance degradation
  - High amount of time spent in systems calls
  - No apparent extra load on the network
  - Some more driver tuning would be useful (probably addressed in newer OFED)
- Read the paper for the full results

Multi-Fabric Hosts

- What if you want to have iWARP and Infiniband interconnects on the same machine?
  - Could imagine a situation where RDMA over TCP is used for (say) storage, but Infiniband is used for MPI interconnect
  - Another case is in a benchmarking environment, it is very useful to be able to run tests back to back with no hardware reconfiguration required
- This is possible, at least for some subset of cases

More Multi-Fabric Hosts

- OpenMPI is easy, it is an mca parameter:
  - NetEffect: `--mca btl_openib_if_include nes0`
  - Mellanox: `--mca btl_openib_if_include mlx4_0`
  - Others are possible
- HP-MPI should be easy, just change MPI_HASIC_UDAPL
  - However, since /etc/dat.conf parsing is broken, changing fabrics ends up requiring an system config file change

Conclusions

- iWARP and RDMA over Ethernet networks in general require a change in mindset
  - A 48-port 10GbE switch with a few uplinks is not sufficient
  - Need a fully non-blocking network, either in a chassis switch or with TRILL
- NetEffect hardware “looks” similar enough to IB to be supported by MPI with minor alterations (MPI applications themselves don’t care)

More Conclusions

- However, the rest of the ecosystem is still catching up
  - ISV codes bundled with old MPI versions are the biggest offender
- Impact depends on your environment
  - Could be a non-issue for environments with heavy open-source or community code usage
  - If you heavily rely on ISV codes, it could be a big impediment to an iWARP deployment

Acknowledgements

- Julie Cummings of Intel for providing expert technical assistance
- Tom Stachura and William Meigs of Intel for coordinating everything

Questions?