RI2N/DRV: Multi-link Ethernet for High-Bandwidth and Fault-Tolerant Network on PC Clusters

Shin’ichi Miura, Toshihiro Hanawa, Taiga Yonemoto, Taisuke Boku, and Mitsuhiisa Sato
University of Tsukuba
Background

- Performance of network is most important problem for PC clusters.
  - Several SANs (System Area Network) are developed for PC clusters.
    - Network’s cost/performance ratio is not good.
  - Although cost/performance ratio of Ethernet is very good, two big problems exist.
    - Sustained performance of Ethernet is lower than SAN.
    - Robustness of Ethernet is relatively low compared with SAN components.

- Our solution …
  - Utilizing multiple Ethernet links
    - In normal state;
      - Distributing packets on multiple links to get wider bandwidth
    - In faulty state;
      - Transmitting packets on only normal links to continue communication.
RI2N

- RI2N: Redundant Interconnection with Inexpensive Network

- RI2N/UDP (T. Okamoto et al, CAC2007)
  - User level implantation on multi-thread library and UDP socket

- RI2N/DRV
  - System level implementation
    - High system transparency
  - Retransmission and congestion control relies on upper layer protocol such as TCP/IP
    - Low Communication cost
Related Works

- **Hardware implementation**
  - IEEE 802.3ad. (LACP)
    - Special switch is required.
    - Multiplexing of a switch is not considered.
    - Single point of failure
  - Special switch is required.
- **Software implementation**
  - Linux Channel Bonding (*balance-rr* mode)
    - Packet disordering problem
      - Low throughput
    - Similar to RI2N
  - Solution for packet disordering problem
    - TCP
      - Remodeling TCP might influence the user application
      - Heavy CPU Load
      - Increased ACK packet from Receiver side
      - Traffic congestion
    - SCTP
      - Low program portability
Linux Channel Bonding

- **Packet Transmitter**
  - LCB distribute packets to multiple physical network links in a round-robin manner

- **Packet Receiver**
  - NIC transfer the packet directly to a upper layer protocol such as IP.
    - Interrupt coalescing
    - NAPI on Linux device driver
      - Multiple packets are pushed all at once to save the amount of interrupt handling
    - Disordering packets
      - A lot of ACK is sent to Sender side
  - LCB does not care of receive operation.
RI2N

- Packet Transmitter
  - Round-robin as well as transmitter of LCB
- Packet Receiver
  - NIC transfer the packet to RI2N
  - Packets are saved to the buffer of RI2N
  - Reordering received packets
  - Transfer it to a upper layer protocol with correct order
    - Does not generate a lot of ACK packets by correct order

- This Reordering Mechanism improve throughput
**Implementation of RI2N**

- RI2N makes pseudo network device to transmit and receive packets
  - Compatible with ordinary Ethernet device
  - The user application can use TCP/IP and UDP/IP communication without special modification

- RI2N enhances the Ethernet packet to packet reordering
  - Insert “RI2N Header” to Ethernet packet between Ethernet Header and Ethernet Payload

- Receive Operation
  - Receives Ethernet packet instead of upper layer protocol such as IP
  - Transfer Ethernet packet after packet reordering

- Receive Operation requires 2 software interrupts to reordering packet for each packet
  - It cause large latency at short message
Basic Performance Evaluation

- Benchmarks:
  - Amount of ACK Packets
  - Latency
  - Throughput on MPI
  - Two-Way Communication

- Network:
  - Single
    - Single Gigabit Ethernet
  - LCB
    - Linux Channel bonding with 2 GbE
  - RI2N
    - RI2N/DRV with 2 GbE

---

**CPU**
- Intel Xeon 3110
- 3.0 GHz dual core

**Memory**
- DDR2/800 4096MB

**OS**
- Linux Kernel 2.6.27.7

**NIC**
- Intel PRO/1000PT Dual port
- MTU=1500

**NIC driver**
- e1000e 0.3.3.3-k

**MPI**
- OpenMPI 1.2.4

---

**Switch**

**Node A**

**Switch**

**Node B**

---
### Amount of ACK Packets

- By packet reordering in RI2N, unnecessary ACK packets are reduced.

<table>
<thead>
<tr>
<th></th>
<th>Number of ACK Packets</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single</td>
<td>370</td>
</tr>
<tr>
<td>LCB</td>
<td>741</td>
</tr>
<tr>
<td>RI2N</td>
<td>376</td>
</tr>
</tbody>
</table>

LCB has twice number of ACK compared with RI2N. RI2N is same as SINGLE.

RI2N reduces ACK packets due to packet disordering.
Latency over Normal TCP/IP

Latency of RI2N is larger than the other environment in a short message.

Receive operation of RI2N requires 2 software interrupts to reordering packet.
Throughput on MPI (Ping-Pong)

Most typical applications require high communication bandwidth.

RI2N Throughput is better than LCB.

RI2N Throughput is lower than LCB.

The 9th Workshop on Communication Architecture for Clusters (CAC2009)

25 May, 2009
Two-way Communication (TCP/IP)

<table>
<thead>
<tr>
<th></th>
<th>Throughput</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single</td>
<td>117 MB/sec</td>
</tr>
<tr>
<td>LCB</td>
<td>159 MB/sec</td>
</tr>
<tr>
<td>RI2N</td>
<td>214 MB/sec</td>
</tr>
</tbody>
</table>

The performance of LCB and RI2N are better than single.

Difference is 55 MB/sec

ACK packet disturbs the stream on the opposite way.

Packet disordering frequently occurs.

Transmits a large number of ACK packets.

25 May, 2009

The 9th Workshop on Communication Architecture for Clusters (CAC2009)
Bonnie++ Benchmarks on NFS

- Bonnie++ Benchmarks
  - Sequential Write (put_block)
  - Sequential Read (get_block)
  - Sequential Rewrite (rewirte)
- File Size: 2.0 GB
- NFS block sizes is 32 KB
- Dual-link Gigabit Ethernet provides up to more than 200 Mbytes/sec of data transfer performance
  - Prepared a tempfs memory disk on the NFS server
  - Direct I/O performance takes about 1.4 Gbytes/sec
Bonnie++ Benchmarks on NFS (Results)

The cause of the difference of these performances of LCB and RI2N is the performance of the two way communication.

LCB is almost equal throughput to single.

RI2N achieve highest performance.

RI2N provide lower CPU usage than LCB.

High CPU utilization of RI2N.
NAS Parallel Benchmarks

- NAS Parallel Benchmarks 3.3
  - CLASS: B
  - Number of Process: 4 and 8
  - Benchmarks:
    - 5 Kernels
      - EP, FT, IS, MG and CG
    - 3 Application
      - LU, BT and SP

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>Intel Xeon 5110 1.6GHz dual core</td>
</tr>
<tr>
<td>Memory</td>
<td>DDR2/800 4092MB</td>
</tr>
<tr>
<td>OS</td>
<td>Linux 2.6.27.7</td>
</tr>
<tr>
<td>NIC</td>
<td>Intel PRO 1000PT Dual port</td>
</tr>
<tr>
<td></td>
<td>MTU=1500</td>
</tr>
<tr>
<td>NIC driver</td>
<td>e1000e 0.3.3.3-k</td>
</tr>
<tr>
<td>MPI</td>
<td>OpenMPI 1.2.4</td>
</tr>
</tbody>
</table>

Switch

Switch

0 1 2 3 4 5 6 7
NAS Parallel Benchmarks (Results)

The performance of RI2N is improved by high bandwidth

Decrease the message size

Relative performance becomes worse

The performance of LCB worse than SINGLE environment though the bandwidth has increased

performance has not improved

Relative performance

EP FT IS MG CG LU BT SP

- single (4 node)
- LCB (4 node)
- RI2N (4 node)
- single (8 node)
- LCB (8 node)
- RI2N (8 node)
Conclusion

- RI2N;
  - Provide High-bandwidth and Fault-tolerance with multiple Ethernet link
  - Implemented as a pseudo network interface of Linux
- A user can use this system transparently
  - An existing system can use this system without the re-compilation.
- Prototype Implementation
  - The performance is moderately good
    - However, it is necessary to improve it.
- RI2N can be applied not only for inter-node communication such as MPI but also for any UNIX network service that requires a high-speed, reliable network
Thank you for your kind attention