What is the Future for High-Performance Networking?

Wu-chun (Wu) Feng
feng@lanl.gov

RADIANT: Research And Development in Advanced Network Technology
http://www.lanl.gov/radiant
Computer & Computational Sciences Division
Los Alamos National Laboratory
University of California

IEEE Distinguished Visitors Program
Ottawa, Canada, December 4, 2003
What is the Future for High-Performance Networking?

• A loaded question ...
• ... one that opens up a “can of worms” ...

• Why? So many dimensions to consider.
  ➢ Hardware: Optical vs. Electronic
  ➢ End-to-End Connectivity: Circuit- vs. Packet-Switched
  ➢ Routing
    ➢ Wormhole vs. Virtual Cut-Through vs. Store-and-Forward
    ➢ Source vs. IP
  ➢ Resource Usage: Dedicated vs. Shared
  ➢ Quality of Service: Best Effort vs. Guaranteed
  ➢ Environment: LAN vs. SAN vs. MAN vs. WAN
Outline

• High-Performance Networking (HPN) Today
  ➢ Definition: Relative to High-Performance Computing (HPC)
  ➢ What is HPC? → What is HPN?
  ➢ Problems with HPN
    ▪ Host-Interface Bottlenecks
    ▪ Adaptation Bottlenecks

• High-Performance Networking (HPN) Tomorrow

• Conclusion
HPN Today: What is HPC?

- **Tightly-Coupled Supercomputers**
  - LANL’s ASCI Q, Japanese Earth Simulator
- **High-End Clusters / PC Clusters**
  - NCSA’s Titan (part of DTF/TeraGrid), LANL’s *Green Destiny*
- **Distributed Clusters & MicroGrids**
  - OSC’s distributed cluster, Intel’s enterprise microgrid
- **Computational Grids**
  - Industry: Avaki, Entropia, United Devices.
  - Academia & DOE Labs: Earth Systems Grid, Particle Physics Data Grid, Distributed Terascale Facility (DTF a.k.a TeraGrid).

*All the above platforms will continue to exist over the next decade, e.g., NCSA’s Titan will be a cluster in its own right as well as a grid node in DTF/TeraGrid (www.teragrid.org).*
HPN Today: Supporting HPC

Why HPN in Supercomputers & Clusters ≠ HPN in Grids & μGrids

Network

NIC
I/O Bus
Memory Bus
$ CPU

NIC
I/O Bus
Memory Bus
$ CPU

Main Memory
I/O Bridge
Main Memory
I/O Bridge

Wu-chun Feng
feng@lanl.gov

Los Alamos National Laboratory
HPN Today: Supporting HPC

Why HPN in Supercomputers & Clusters ≠ HPN in Grids & μGrids

Myrinet, Quadrics, GigE

Bottleneck for supercomputers and clusters

Wu-chun Feng
feng@lanl.gov

www.lanl.gov/radiant
HPN Today: Supporting HPC

Why HPN in Supercomputers & Clusters ≠ HPN in Grids & μGrids

How to infer what is going on in the network?
NOT AN EASY PROBLEM.
HPN Today: Supporting HPC

- **Tightly-Coupled Supercomputers & High-End Clusters**
  - Network Environment: Generally, SANs using non-IP.
  - Why non-IP (source) routing? Low latency more important.
    - Faster network fabric (wormhole or virtual cut-through).
  - Problems
    - Non-scalable beyond a SAN.
    - Host-interface bottlenecks.

- **Computational Grids & Virtual Supercomputers**
  - Network Environment: WAN using TCP/IP.
  - Why IP routing? Scalability more important.
  - Why is performance so lousy over the WAN?
    - Adaptation bottlenecks.
Host-Interface Bottlenecks

10GigE packet inter-arrival: 1.2 µs
(assuming 1500-byte MTUs)
Null system call in Linux: 5 µs

- **Software**
  - Host can only send & receive packets as fast as OS can process them.
    - Excessive copying. (A known fact.)
    - Excessive CPU utilization. (See next slide.)

- **Hardware (PC)**
  - PCI-X I/O bus. 64 bit, 133 MHz = 8.5 Gb/s.
    - Not enough to support 10-Gigabit Ethernet.
  - Solutions in the Future?
    - PCI Express: Network interface card (NIC) closer to CPU
    - InfiniBand 4x & Beyond: NIC on packet-switched network
    - 3GIO/Arapahoe (Intel)
    - Hypertransport (AMD)
Host-Interface Bottlenecks

10GigE packet inter-arrival: 1.2 µs
(assuming 1500-byte MTUs)
Null system call in Linux: 5 µs

• Software
  - Host can only send & receive packets as fast as OS can process them.
    - Excessive copying. (A known fact.)
    - Excessive CPU utilization.

• Hardware (PC)
  - PCI-X I/O bus. 64 bit, 133 MHz = 8.5 Gb/s.
    - Not enough to support 10-Gigabit Ethernet.
  - Solutions in the Future?
    - PCI Express: Network interface card (NIC) closer to CPU
    - InfiniBand 4x & Beyond: NIC on packet-switched network
    - 3GIO/Arapahoe (Intel)
    - Hypertransport (AMD)

We have reached a crossover point with current software and hardware - network speeds are outstripping the ability of the CPU to keep up.
666-MHz Alpha & GigE with Linux
(Courtesy: USC/ISI)

Even jumbograms suffer from high CPU utilization ...

CPU utilization is even worse with 10GigE. For more information, see Feng et al., “Optimizing 10-Gigabit Ethernet ...,” SC2003, Nov. 2003.
Host-Interface Bottleneck (Software)

- **First-Order Approximation**
  - deliverable bandwidth = maximum-sized packet / interrupt latency
  - e.g., 1500-byte MTU / 5 ms = 300 MB/s = 2400 Mb/s = 2.4 Gb/s

- **Problems**
  - Maximum-sized packet (or MTU) is only 1500 bytes for Ethernet.
  - Interrupt latency to process a packet is quite high.
  - CPU utilization for network tasks is too high.

- **“Network Wizard” Solutions**
  - Eliminate excessive copying.
  - Reduce frequency of interrupts.
  - Increase effective MTU size.
  - Reduce interrupt latency.
  - Reduce CPU utilization.

These techniques were used to help smash the Internet2 Land Speed Record in Feb. 2003.
“Network Wizard” Solutions (many non-TCP & non-standard)

- Interrupt Coalescing
  - Increases bandwidth (BW) at the expense of even higher latency.
- Jumbograms
  - Increases BW with minimal increase in latency.
  - Lacks interoperability.
  - Very difficult to build switches to process large packets at high speeds.
- Reduction of CPU Utilization (with OS-based TCP/IP)
  - Provide “zero-copy” TCP, TCP offload engine, or high-performance IP but OS still middleman.
  - Push protocol processing into hardware, e.g., checksums. Dangerous?
- OS-Bypass Protocol with RDMA
  - Increases BW & decreases latency by an order of magnitude or more.
  - Remote Direct Data Placement: RDMA over IP.
"Network Wizard" Solutions (many non-TCP & non-standard)

- **Interrupt Coalescing**
  - Increases bandwidth (BW) at the expense of even higher latency.

- **Jumbograms**
  - Increases BW with minimal increase in latency.
  - Lacks interoperability.
  - Very difficult to build switches to process large packets at high speeds.

- **Reduction of CPU Utilization (with OS-based TCP/IP)**
  - Provide “zero-copy” TCP, TCP offload engine, or high-performance IP but OS still middleman.
  - Push protocol processing into hardware, e.g., checksums. Dangerous?

- **OS-Bypass Protocol with RDMA**
  - Increases BW & decreases latency by an order of magnitude or more.
  - Remote Direct Data Placement: RDMA over IP.
"Network Wizard" Solutions

Network

NIC
I/O Bus
Memory Bus

I/O Bridge

Main Memory

CPU

$
"Network Wizard" Solutions (many non-TCP & non-standard)

- **Interrupt Coalescing**
  - Increases bandwidth (BW) at the expense of even higher latency.

- **Jumbograms**
  - Increases BW with minimal increase in latency.
  - Lacks interoperability.
  - Very difficult to build switches to process large packets at high speeds.

- **Reduction of CPU Utilization (with OS-based TCP/IP)**
  - Provide “zero-copy” TCP, TCP offload engine, or high-performance IP but OS still middleman.
  - Push protocol processing into hardware, e.g., checksums. Dangerous?

- **OS-Bypass Protocol with RDMA**
  - Increases BW & decreases latency by an order of magnitude or more.
  - Remote Direct Data Placement: RDMA over IP.
“Network Wizard” Solutions

Network

NIC
I/O Bus
Memory Bus
$CPU
Main Memory

I/O Bridge

OS

APP

NIC
I/O Bus
Memory Bus
$CPU
Main Memory

I/O Bridge

OS

APP

Wu-chun Feng
feng@lanl.gov
www.lanl.gov/radiant
“Network Wizard” Solutions (many non-TCP & non-standard)

- **Interrupt Coalescing**
  - Increases bandwidth (BW) at the expense of even higher latency.

- **Jumbograms**
  - Increases BW with minimal increase in latency.
  - Lacks interoperability.
  - Very difficult to build switches to process large packets at high speeds.

- **Reduction of CPU Utilization (with OS-based TCP/IP)**
  - Provide “zero-copy” TCP, TCP offload engine, or high-performance IP but OS still middleman.
  - Push protocol processing into hardware, e.g., checksums. Dangerous?

- **OS-Bypass Protocol with RDMA**
  - Increases BW & decreases latency by an order of magnitude or more.
  - Remote Direct Data Placement: RDMA over IP.
"Network Wizard" Solutions
High-Performance IP over Ethernet

- **Lightweight Protocol Off-Loading**
  - (Mis)configure device driver to accept *virtual MTUs* (vMTU) of up to 64 KB → TCP/IP transmits up to 64-KB vMTU to device driver.
  - **Result:** Minimize CPU overhead for fragmentation.
  - Make the firmware on the NIC do the fragmentation.
  - Implement with programmable NIC.
    - Alteon GigE AceNICs.
    - Programmable 10GigE NICs that will be coming out in 2004.
“Network Wizard” Solutions (many non-TCP & non-standard)

- **Interrupt Coalescing**
  - Increases bandwidth (BW) at the expense of even higher latency.
- **Jumbograms**
  - Increases BW with minimal increase in latency.
  - Lacks interoperability.
  - Very difficult to build switches to process large packets at high speeds.
- **Reduction of CPU Utilization (with OS-based TCP/IP)**
  - Provide “zero-copy” TCP, TCP offload engine, or high-performance IP but OS still middleman.
  - Push protocol processing into hardware, e.g., checksums. Dangerous?
- **OS-Bypass Protocol with RDMA**
  - Increases BW & decreases latency by an order of magnitude or more.
  - Remote Direct Data Placement: RDMA over IP.
OS-Bypass Protocol with RDMA
(e.g., ST: Scheduled Transfer and Quadrics Elan)

- **Bottleneck:** Application-to-network interface

<table>
<thead>
<tr>
<th>Host</th>
<th>Appl.</th>
<th>OS</th>
<th>ST</th>
<th>Network</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ST OS-Bypass</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ST OS-Bypass</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Requires</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>&quot;Smart NIC&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>TCP/IP</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>NIC</td>
</tr>
</tbody>
</table>

- **OK for SAN, but what about WAN?**
  - WAN uses IP, not source routing. General concepts still translate, however. See IETF RDDP effort.
  - How would it compare to an OS-based high-performance TCP?
## Bridging the “Wizard Gap” for All (Across All Network Environments)

### Performance Numbers from User Space to User Space

<table>
<thead>
<tr>
<th>Environment</th>
<th>Typical</th>
<th>“State of the Art” w/ Network Wizards</th>
<th>Our Research</th>
</tr>
</thead>
<tbody>
<tr>
<td>LAN with TCP/IP</td>
<td>300-400 Mb/s 100 µs</td>
<td>990 Mb/s → 2500 Mb/s 80 µs → 20 µs</td>
<td>4640 Mb/s → 7329 Mb/s 20 µs → 9 µs</td>
</tr>
<tr>
<td>SAN with OS-Bypass/RDMA</td>
<td>2000 1920 Mb/s 8.5 µs</td>
<td>2456 Mb/s (MPI-to-MPI) 4.9 µs</td>
<td></td>
</tr>
<tr>
<td>SAN with TCP/IP</td>
<td>2003 1968 Mb/s 6.7 µs</td>
<td>7200 Mb/s (MPI-to-MPI) &lt; 3.0 µs</td>
<td></td>
</tr>
<tr>
<td>WAN with TCP/IP (distance normalized)</td>
<td>0.007 Petabit-meters per second</td>
<td>0.270 Petabit-meters per second</td>
<td>23.888 Petabit-meters per second*</td>
</tr>
</tbody>
</table>

Host-Interface Bottlenecks

10GigE packet inter-arrival: 1.2 µs
(assuming 1500-byte MTUs)
Null system call in Linux: 5 µs

• Software
  - Host can only send & receive packets as fast as OS can process them.
    - Excessive copying. (A known fact.)
    - Excessive CPU utilization. (See next slide.)

• Hardware (PC)
  - PCI-X I/O bus. 64 bit, 133 MHz = 8.5 Gb/s.
    - Not enough to support 10-Gigabit Ethernet.
  - Solutions in the Future?
    - PCI Express: Network interface card (NIC) closer to CPU
    - InfiniBand 4x & Beyond: NIC on packet-switched network
    - 3GIO/Arapahoe (Intel)
    - Hypertransport (AMD)
Host-Interface Bottleneck (Hardware)

- **PCI = Pretty Crappy Interface 😊**
  - Theoretical Peak Bandwidth
    - PCI 2.2, 32/33: 1.06 Gb/s
    - PCI 2.2, 64/33: 2.13 Gb/s
    - PCI 2.2, 64/66: 4.26 Gb/s
    - PCI-X 1.0, 64/100: 6.40 Gb/s
    - PCI-X 1.0, 64/133: 8.51 Gb/s
- **Solutions? More or less out of our control ...**
  - PCI-X → 8.51 Gb/s (today)
  - InfiniBand → 8.51 Gb/s (today), 10 Gb/s, i.e., 4x (soon), ???
  - 3GIO/Arapahoe (full duplex) → 51.2 Gb/s (2004/2005)
  - Hypertransport → 25.6 Gb/s (today)
The Future: Eliminating Host-Interface Bottlenecks for HPN

• Convergence and subsequent “standardization” of software techniques in SAN, but …
  ➢ True high-end HPC: OS-bypass/RDMA over source routing.
  ➢ Commodity HPC: OS-bypass/RDMA over IP (e.g., IETF RDDP) with subsequent extension into the WAN.

• Continued uniqueness in architecture for reducing hardware-based, host-interface bottlenecks.
  ➢ Communications Streaming Architecture → PCI Express (Intel).
  ➢ Hypertransport (AMD, Sun, and many others).
  ➢ Infiniband (companies delivering true high-end HPC)
    ▪ Note Intel’s & Microsoft’s withdrawal from Infiniband.
HPN Today: Supporting HPC

- Tightly-Coupled Supercomputers & High-End Clusters
  - Network Environment: Generally, SANs using non-IP.
  - Why non-IP (source) routing? Low latency more important.
    - Faster network fabric (wormhole or virtual cut-through).
  - Problems
    - Non-scalable beyond a SAN.
    - Host-interface bottlenecks.

- Computational Grids & Virtual Supercomputers
  - Network Environment: WAN using TCP/IP.
  - Why IP routing? Scalability more important.
  - Why is performance so lousy over the WAN?
    - Adaptation bottlenecks.
HPN Today: Supporting HPC

• Tightly-Coupled Supercomputers & High-End Clusters
  - Network Environment: Generally, SANs using non-IP.
  - Why non-IP (source) routing? Low latency more important.
  - Faster network fabric (wormhole or virtual cut-through).
  - Problems
    - Non-scalable beyond a SAN.
    - Host-interface bottlenecks.
  - Addressing adaptation problems not only support HPC today but will also eventually benefit the Internet tomorrow.

• Computational Grids & Virtual Supercomputers
  - Network Environment: WAN using TCP/IP.
  - Why IP routing? Scalability more important.
  - Why is performance so lousy over the WAN?
    - Adaptation bottlenecks.
HPN Today: Supporting HPC

Why HPN in Supercomputers & Clusters ≠ HPN in Grids & μGrids

How to infer what is going on in the network?
NOT AN EASY PROBLEM.
Adaptation Bottlenecks

Big network “pipes” help but are only part of the solution. What are the dynamics? How to ensure end-to-end performance?
Adaptation Bottlenecks

- **Flow Control**
  - End-to-end issue.
  - Receiver advertises to sender how much data it can handle.
  - Advertised window (awnd)
    - Static 32 KB in typical OS.

- **Congestion Control**
  - Global issue.
  - Send infers what the available bandwidth in the network is.
  - Congestion window (cwnd)
    - Dynamic adjustment based on inferred network conditions.

\[
\text{sending window} = \min(\text{awnd}, \text{cwnd})
\]
Flow-Control Adaptation

• Issues
  ➢ No adaptation currently being done in any “standard” TCP.
  ➢ 32-KB static-sized buffer is supposed to work for both LAN & WAN.
• Problem: Large bandwidth-delay products require flow-control windows as large as 1024-KB to fill the network pipe.
• Consequence: As little as 3% of network pipe is filled.
• Preliminary Solutions
  ➢ Manual tuning of buffers at send and receive end-hosts.
    ▪ Too small → low bandwidth. Too large → waste memory (LAN).
  ➢ Automatic tuning of buffers.
    ▪ Auto-tuning (similar to Linux auto-tuning) by Semke et al. @ PSC.
      - Sender-based flow control.
    ▪ Dynamic right-sizing by Feng et al. @ LANL.
      - Receiver-based flow control.

The Future: Transparent Flow-Control Adaptation

- Without a “network wizard” ...
  - Wide-area transfer between SNL & LANL of a 150-GB dataset.
    - OC-3 (155 Mb/s): 8 Mb/s → 42 hours  “Wizard Magic”: 55 Mb/s
    - OC-12 (622 Mb/s): 8 Mb/s → 42 hours  “Wizard Magic”: 240 Mb/s
  - The bandwidth of a driving tapes of the data from SNL to LANL is a LOT better! 150 GB / 1.75 hours = 190 Mb/s.

```
Sender
A A A A A A A A A A A A A A A A A A A A
Receiver
```

Transparencyly provide end-to-end performance to the application, thus “eliminating” the need for network wizards.
Congestion-Control Adaptation

• Adaptation mechanisms will not scale due to
  - Additive increase / multiplicative decrease (AIMD) algorithm.
    - Linear increase of MSS too small for the next-generation Internet.

• TCP Reno congestion control
  - Bad: Allow/induce congestion.
    - Detect & recover from congestion.
  - Analogy: “Deadlock detection & recovery” in OS.
  - Result: “At best” 75% utilization in steady state (assuming no buffering).

• TCP Vegas congestion control
  - Better: Approach congestion but try to avoid it.
    - Usually results in better network utilization.
  - Analogy: “Deadlock avoidance” in OS.
“Optimal” Bandwidth

- The future performance of computational grids (as well as clusters & supercomputers trying to get away from ULNI scalability problems) looks bad if we continue to rely on the current version of the widely-deployed TCP Reno.

Example: High BW-delay product: 1 Gb/s WAN * 100 ms RTT = 100 Mb

- Additive increase
  - when window size is 1 → 100% increase in window size.
  - when window size is 1000 → 0.1% increase in window size.

Re-convergence to “optimal” bandwidth takes nearly 7 minutes!
(Performance is awful if network uncongested.)

Solutions: (1) Faster converging congestion control. (2) Larger MTU. (3) Different paths or multiple paths.
The Future: Non-AIMD Congestion Control But “TCP-Friendly”

- AIMD is “stable & fair” but
  - Not well-suited for emerging applications (e.g., remote computational steering of a visualization dataset)
    - Its reliability and ordering semantics increase end-to-end delays and delay variations.
    - Streaming applications generally do not react well to the large and abrupt reductions in transmission rate caused by AIMD.
  - Potential General Solutions
    - Deploy “TCP-friendly” (non-AIMD) congestion-control algorithms, e.g., binomial congestion-control algorithms.
    - Use network measurement, monitoring, and tomography to enable better adaptation in support of grids.
  - Specific Solutions on the Horizon
    - FAST TCP (led by Low @ Caltech with CERN, LANL, and SLAC).
    - Scalable TCP (Kelly @ CERN)
    - HS-TCP (Floyd @ ICIR)
    - SCTP (IETF effort)
Conclusion: What is the Near-Term Future of HPN?

- **Host-Interface Bottlenecks**
  - **Software**
    - A host can only send and receive packets as fast as the OS can process the packets.
  - **Hardware (PC)**
    - PCI I/O bus. 64 bit, 133 MHz = 8.5 Gb/s.

- **Adaptation Bottlenecks**
  - **Flow Control**
    - No adaptation currently being done in any standard TCP.
    - Static-sized window/buffer is supposed to work for both the LAN and WAN.
  - **Congestion Control**
    - Adaptation mechanisms will not scale, particularly TCP Reno (although TCP Reno w/ SACK helps immensely).
Conclusion: What is the Long-Term Future of HPN?

- It’s here in Canada!

- For the next ten years, Canarie will eliminate the need to deal with adaptation bottlenecks.
  - Bottleneck moves to scheduling lightpaths efficiently.

- In ten years?
  - If CHEETAH over Canarie-like network is efficient, ok.
  - Otherwise, packet-switched optical ...
Recent & Relevant Publications …

A Sample of Recent Media Coverage
