Virtual TCP Offload: Optimizing Ethernet Overlay Performance on Advanced Interconnects

Zheng Cui  University of New Mexico
Patrick Bridges  University of New Mexico
Jack Lange  University of Pittsburgh
Peter Dinda  Northwestern University

http://v3vee.org
Overview

• **We need fast virtual Ethernet overlay**
  • Virtual Ethernet overlays are powerful
  • Slow on high-end networks like InfiniBand

• **Problem: Semantic gap between overlay and physical networks**
  • Duplicated protocol processing
  • More virtual interrupts
  • Difficult to efficiently leverage advanced interconnect features

• **Solution: Virtual TCP Offload**
  • Bridges semantic gap
  • Leverages advanced interconnect features

• **Result:** Dramatically improved application performance
Virtualizing High-End Networks

• **Virtual Ethernet overlays are powerful**
  • Enable Ethernet applications on high-end networks
  • Ease network deployment/management
  • Provide location/hardware independence
  • Support broad classes of applications/stacks

• **Performance on high-end networks (e.g., InfiniBand) is slow:**
  • Latency: 40 times higher than native/uverbs
  • Throughput: ~30% of native/uverbs
  • 20-80% HPCC application benchmark slowdown

• **High-end networks need better overlay network support**
Modern Virtual Ethernet Overlay: VNET/P

- Layer 2 virtual Ethernet overlay
- Embedded in Palacios VMM
- Three Components:
  - Virtual NIC for each guest OS
  - VNET core
  - VNET bridge
- 1G Ethernet
  - 3x higher latency
  - Near-native throughput
  - Near-native MPI application performance

Fig. 1. VNET/P architecture.
Semantic Gap Between Overlay Features and Physical Network Features

Application:
Reliable Stream

InfiniBand:
Reliable Stream
Semantic Gap Between Overlay Features and Physical Network Features

**Application:**
- Reliable Stream

**Virtual Ethernet:**
- Unreliable Datagram

**InfiniBand:**
- Reliable Stream
Semantic Gap Between Overlay Features and Physical Network Features

- Application:
  - Reliable Stream

- TCP

- Virtual Ethernet:
  - Unreliable Datagram

- InfiniBand:
  - Reliable Stream

Duplication

Virtual Interrupts
(ACKs)

Semantic Gap
Two Approaches

Virtual Ethernets on heterogeneous interconnects:

• Minimal interconnect features

• Advanced interconnect features without guest knowledge
Approach #1: Minimize Semantic Gap by Using Minimal Features

**Application:**
- Reliable Stream

**TCP**

**Virtual Ethernet:**
- Unreliable Datagram

**UD MTU limitations:** < 4K
- Increasing # of network headers
- Increasing routing decisions
- Increasing protocol processing cost
- Increasing # of virtual interrupts

**InfiniBand:**
- Reliable Stream
- Unreliable Datagram
Approach #2: Minimize Semantic Gap by Translating to Advanced Features

**Application:**
- Reliable Stream

**Duplication**
- Virtual Interrupts (ACKs)

**Semantic Gap**

**TCP**

**Virtual Ethernet:**
- Unreliable Datagram
- Reliable Stream

**InfiniBand:**
- Reliable Stream
Virtual TCP Offload

Add TCP Offload to Virtual NIC

- Keeps Ethernet abstractions
- Guests designate reliable/unreliable traffic at Ethernet level
Virtual TCP Offload architecture

- Guest Applications
  - TOE Driver
  - UDP Stack
  - NIC Driver

- Palacios VMM
  - VTOE
  - Vnet Core CA (TCP)
  - Vnet Host CA

- Host
  - IB RC
  - InfiniBand Driver
  - IPoIB

- IPoIB Interconnect
Map VTOE TCP Connections to Physical Network Connections

- Maps VNET Connection ID (SID) – host shadow Connection ID (CID)
- Manages DMA buffers for zero-copy in overlay
- Translates events/interrupts
VTOE NIC Architecture

Operations:

• **Connection creation/teardown and state changes:**
  • IO Ports
  • Event Queue (shared ring buffer)
  • Connect_Request, Connect_Established, Disconnected, Address_Error, Unreachable, Connect_Rejected ...

• **Data movement**
  • SendWQ and RecvWQ (shared ring buffers)
  • Tagged with SID for each buffer
  • Virtual interrupts
Implementation

Linux Guest over InfiniBand Interconnects

- Connection Management: TCP vs InfiniBand state machines
  - Connection establishment
  - Connection termination

- Data Transfer: Avoiding copy and page-flipping cost [1]
  - Transmission with zero overlay copies
  - Reception with zero overlay copies

- Interfacing with Linux Guests

Implementation: Connection Establishment

TCP CLOSE / IB IDLE
- guest open
- send IB_REQ

TCP LISTEN / IB Listen
- receive IB_REQ
- CONNECT REQUEST

TCP SYN RECV
- IB REQ RCVD
- send IB_REP
- IB REP SENT

TCP SYN SENT
- receive IB_REP

TCP ESTABLISHED / IB ESTABLISHED
- receive IB_RTU
- CONNECT RESPONSE

VTOE EVENT

06/25/13
Implementation: Connection Termination

- Passive close
- Active close

VTOE EVENT

TCP ESTABLISHED/IB ESTABLISHED

- guest send FIN receive IB_DREQ
- guest receive FIN

TCP FIN WAIT1/IB DREQ RCVD

- DISCONNECT Send IB_DREP

TCP TIME WAIT/IB TIME WAIT

TCP CLOSE WAIT

- guest send FIN send IB_DREQ
- receive IB_DREP

TCP LAST ACK/IB DREQ SENT

TCP TIME WAIT/IB TIME WAIT

TCP CLOSE/IB IDLE

timeout

06/25/13
Testbed

- **6-node cluster:** 8-core AMD Opteron CPU + 32GB RAM + Mellanox MT26428 10 Gbps InfiniBand NIC

- **Configuration:**

```plaintext
+-----------------+----------------+-----------------+----------------+------------------+
| Native+Uverbs   | Native+SDP     | Native+IPoIB    | VNET+IPoIB     | VNET+VTOE        |
|                 |                |                 |                |                  |
| Application     | Application    | Application     | Application    | Application      |
| Libverbs        | Busybox        | Busybox         | Busybox        |                  |
| Busybox         | SDP            | IPoIB           | IPoIB          | VNET/P           |
| Linux           | Linux          | Linux           | Linux          | IB NIC           |
| IB NIC          | IB NIC         | IB NIC          | IB NIC         | IB NIC           |

Native                     Native                     Native                     VM                     VM

06/25/13
VTOE: Near-native TCP Bi-directional Throughput on IB

TTCP Bi-directional -l8862 -n4000000

Bandwidth (Gbps)

- Native+SDP
- Native+IPoIB
- VNET+VTOE
- VNET+IPoIB

- BW-stream1
- BW-stream2
VTOE: Near-native TCP Bi-stream Throughput on IB

TTCP Bi-stream -l8862 -n4000000

% CPU Utilization

Native+SDP  Native+IPoIB  VNET+VTOE  VNET+IPoIB

RX CPU

TX CPU
VTOE: Increased MPI P2P Throughput >2X on IB

**IMB Large Message Pingpong**

- **Native + Uverbs**
- **VNET + VTOE**
- **Native + IPoIB**
- **VNET + IPoIB**

Better than Native+IPoIB
VTOE: Reduced MPI P2P Latency >50% on IB
VTOE: 20X higher MPI latency than Uverb on IB
VTOE: Near-native HPCC MPI Application Performance on IB

Execution Rate (Gflop/s)

HPCC MPIFFT

Native + Uverbs
Native + IPoIB
VNET + VTOE
VNET + IPoIB

8 procs
12 procs
16 procs
20 procs
24 procs
Conclusion

• Virtual Ethernet can achieve high tightly-coupled MPI application performance on heterogeneous interconnects

• Challenges in deploying virtual Ethernet over advanced heterogeneous interconnects:
  • MTU limitations
  • Duplicated RC protocol processing overhead

• Optimization approach: Virtual TCP Offload

• Optimization efficiency:
  • Latency: reduced by 50%
  • Throughput: increased by > 2.5x
  • Near-native throughput-sensitive MPI application performances
Future Work

• **Further reduce latency:** Optimistic interrupts [1]
  • Early Virtual Interrupt (EVI) injection
  • End of Coalescing notifications

• **Reduce memory copies:**
  • Guest application buffers/guest kernel space
  • RDMA

Acknowledgement

• DOE Office of Science Advanced Scientific Computing Research award DE-SC0005050 and DE-SC0005343

• NSF grants CNS-0707365 and CNS-0709168

• Scalable System Lab in University of New Mexico
Contact Information

Zheng Cui
Department of Computer Science
MSC01 1130
University of New Mexico
Albuquerque, 87131

Email: cuizheng@cs.unm.edu
zcui293@gmail.com

http://cs.unm.edu/~cuizheng
Questions?