Data Center Traffic and Measurements: SoNIC

Hakim Weatherspoon
Assistant Professor, Dept of Computer Science
CS 5413: High Performance Systems and Networking
November 12, 2014

Slides from USENIX symposium on Networked Systems Design and Implementation (NSDI) 2013 presentation of “SoNIC: Precise Realtime Software Access and Control of Wired Networks,”
Goals for Today

• Analysis and Network Traffic Characteristics of Data Centers in the wild
Interpacket Delay and Network Research

- Interpacket gap, spacing, arrival time, ...

- Important metric for network research
  - Can be improved with access to the PHY
Network Research enlightened via the PHY

- Valuable information: Idle characters
  - Can provide precise timing base for control
    - Each bit is ~97 ps wide
Network Research enlightened via the PHY

- Valuable information: Idle characters
  - Can provide precise timing base for control
  - Each bit is ~97 ps wide

Packet Generation

Detecting timing channel

Packet Capture

12 /I/s = 100bits = 9.7ns

One Idle character (/I/) = 7~8 bits

Application
Transport
Network
Data Link
Physical
Principle #1: Precision

Precise network measurements is enabled via access to the physical layer (and the idle characters and bits within interpacket gap)
How to control the idle characters (bits)?

• Access to the entire stream is required

• Issue1: The PHY is simply a black box
  – No interface from NIC or OS
  – Valuable information is invisible (discarded)

• Issue2: Limited access to hardware
  – We are network systems researchers
    a.k.a. we like software
Principle #2: Software

Network Systems researchers need software access to the physical layer
Precision + Software = Physics equipment???

- **BiFocals** [IMC’10 Freedman, Marian, Lee, Birman, Weatherspoon, Xu]
  - Enabled novel network research
  - Precision + Software = Laser + Oscilloscope + Offline analysis
  - Allowed precise control in software

- **Limitations**
  - Offline (not *realtime*)
  - Limited Buffering
  - Expensive
Principle #3: Realtime

Network systems researchers need access and control of the physical layer (interpacket gap) continuously in realtime.
Challenge

- **Goal:** Control *every* bit in *software* in *realtime*
  - Enable novel network research

- **Challenge**
  - Requires unprecedented software access to the PHY
Outline

• Introduction

• SoNIC: Software-defined Network Interface Card
  – Background: 10GbE Network Stack
  – Design

• Network Research Applications

• Conclusion
SoNIC: Software-defined Network Interface Card

• Implements the PHY in software
  – Enabling control and access to every bit in realtime
  – With commodity components
  – Thus, enabling novel network research

• How?
  – Backgrounds: 10 GbE Network stack
  – Design and implementation
    • Hardware & Software
    • Optimizations
10GbE Network Stack

- Application
- Transport
- Network
- Data Link
- Physical
  - 64/66b PCS
    - Encode
    - Decode
  - Scrambler
  - Descrambler
  - Gearbox
  - Blocksync
  - PMA
  - PMD

- Physical layer:
  - 64/66b PCS
  - Encode
  - Decode
  - Scrambler
  - Descrambler
  - Gearbox
  - Blocksync
  - PMA
  - PMD

- Data Link:
  - L3 Hdr
  - Data
  - L2 Hdr
  - L3 Hdr
  - Data
  - Eth Hdr
  - CRC
  - Gap

- Network:
  - L3 Hdr
  - Data
  - L2 Hdr
  - L3 Hdr
  - Data
  - Eth Hdr
  - CRC
  - Gap

- Application:
  - L3 Hdr
  - Data
  - L2 Hdr
  - L3 Hdr
  - Data
  - Eth Hdr
  - CRC
  - Gap

- Transport:
  - L3 Hdr
  - Data
  - L2 Hdr
  - L3 Hdr
  - Data
  - Eth Hdr
  - CRC
  - Gap

- Physical layer:
  - 64 bit
  - 2 bit syncheader
  - 16 bit
  - 10.3125 Gigabits
  - Idle characters (/I/)

- Data Link layer:
  - 64 bit
  - Idle characters (/I/)
  - 16 bit

- Physical layer data:
  - 011010010110100101101001011010010110100101101001011010010110100101101
10GbE Network Stack

Application
Transport
Network
Data Link
Physical
64/66b PCS
Encode
Decode
Scrambler
Descrambler
Gearbox
Blocksync
PMA
PMD

11/12/2014
SoNIC NSDI 2013
10GbE Network Stack

Application
Transport
Network
Data Link

Physical
64/66b PCS
   Encode
   Decode
   Scrambler
   Descrambler
   Gearbox
   Blocks

PMA
PMD

SoNIC
NetFPGA

Packet i
Packet i+1

HW

SW

L3 Hdr
L2 Hdr
Eth Hdr
Preamble

SW

HW

SoNIC NSDI 2013

11/12/2014
SoNIC Design and Architecture

Application
Transport
Network
Data Link

Physical
64/66b PCS
  Encode
  Decode
Scrambler
Descrambler
Gearbox
Blocksync

SoNIC

64/66b PCS
  SW
  HW

PMA
PMD

TX PCS
RX PCS

Gearbox
Blocksync
Transceiver
Transceiver
SFP+

APP

L2 Hdr
L3 Hdr

Data

Data Link
Network
Transport
Application

Userspace
Kernel

Data

L3 Hdr
L2 Hdr

Gap

Preamble
Eth Hdr

11/12/2014
SoNIC NSDI 2013
SoNIC Design: Hardware

- To deliver every bit from/to software
  - High-speed transceivers
  - PCIe Gen2 (=32Gbps)
- Optimized DMA engine
SoNIC Design: Software

- Dedicated Kernel Threads
  - TX / RX PCS, TX / RX MAC threads
  - APP thread: Interface to userspace

- Application
- Transport
- Network
- Data Link

- Physical
- 64/66b PCS
  - Encode
  - Decode
  - Scrambler
  - Descrambler
  - Gearbox
  - Blocksync
  - PMA
  - PMD

Packet i   Packet i+1
SoNIC Design: Synchronization

Application
Transport
Network
Data Link
Physical
64/66b PCS
   Encode
   Decode
Scrambler
Descrambler
Gearbox
Blocksync
PMA
PMD

Port 0
APP
TX MAC
TX PCS
RX MAC
RX PCS
APP

Port 1
TX MAC
TX PCS
RX MAC
RX PCS
APP

Low-latency FIFOs

SW
HW

FPGA
PCIe Gen2

Pointer-polling
No Interrupts

11/12/2014
SoNIC NSDI 2013
SoNIC Design: Optimizations

- **Scrambler**
  
  \[ G(x) = x^{58} + x^{39} + 1 \]

  **Naïve Implementation**

  ```
  s \leftarrow \text{state} \\
  d \leftarrow \text{data} \\
  \text{for } i = 0 \rightarrow 63 \text{ do} \\
  \hspace{1cm} \text{in} \leftarrow (d \gg i) \& 1 \\
  \hspace{1cm} \text{out} \leftarrow (\text{in} \oplus s \gg 38) \oplus (s \gg 57) \& 1 \\
  \hspace{1cm} s \leftarrow (s \ll 1) \mid \text{out} \\
  \hspace{1cm} r \leftarrow r \mid (\text{out} \ll i) \\
  \hspace{1cm} \text{state} \leftarrow s \\
  \hspace{1cm} \text{end for} 
  ```

- **CRC computation**
- **DMA engine**
SoNIC Design: Interface and Control

- Hardware control: `ioctl` syscall
- I/O : character device interface
- Sample C code for packet generation and capture

```
#include "sonic.h"

struct sonic_pkt_gen_info info = {
    .mode = 0,
    .pkt_num = 1000000000UL,
    .pkt_len = 1518,
    .mac_src = "00:11:22:33:44:55",
    .mac_dst = "aa:bb:cc:dd:ee:ff",
    .ip_src = "192.168.0.1",
    .ip_dst = "192.168.0.2",
    .port_src = 5000,
    .port_dst = 5000,
    .idle = 12,
};

/* OPEN DEVICE*/
fd1 = open(SONIC_CONTROL_PATH, O_RDWR);
fd2 = open(SONIC_PORT1_PATH, O_RDONLY);

/* CONFIG SONIC CARD FOR PACKET GEN*/
ioctl(fd1, SONIC_IOC_RESET)
ioctl(fd1, SONIC_IOC_SET_MODE, PKT_GEN_CAP)
ioctl(fd1, SONIC_IOC_PORT0_INFO_SET, &info)

/* START EXPERIMENT*/
ioctl(fd1, SONIC_IOC_START)
// wait till experiment finishes
ioctl(fd1, SONIC_IOC_STOP)

/* CAPTURE PACKET */
while ((ret = read(fd2, buf, 65536)) > 0) {
    // process data
}

/* OPEN DEVICE*/
fd1 = open(SONIC_CONTROL_PATH, O_RDWR);
fd2 = open(SONIC_PORT1_PATH, O_RDONLY);

19: /* CONFIG SONIC CARD FOR PACKET GEN*/
20: ioctl(fd1, SONIC_IOC_RESET)
21: ioctl(fd1, SONIC_IOC_SET_MODE, PKT_GEN_CAP)
22: ioctl(fd1, SONIC_IOC_PORT0_INFO_SET, &info)
23
24: /* START EXPERIMENT*/
25: ioctl(fd1, SONIC_IOC_START)
26: // wait till experiment finishes
27: ioctl(fd1, SONIC_IOC_STOP)
28:
29: /* CAPTURE PACKET */
30: while ((ret = read(fd2, buf, 65536)) > 0) {
31: // process data
32: }
33:
34: close(fd1);
35: close(fd2);
```
 Outline

• Introduction
• SoNIC: Software-defined Network Interface Card
• Network Research Applications
  – Packet Generation
  – Packet Capture
  – Covert timing channel
• Conclusion
Network Research Applications

- Interpacket delays and gaps

![Diagram showing IPD and IPG between packets](image)
Packet Generation and Capture

- Basic functions for network research
  - Generation: SoNIC allows control of IPGs in # of /I/s
  - Capture: SoNIC captures what was sent with IPGs in bits

9Gbps, IPD = 13992 bits (1357ns)
Packet Generation

- **SoNIC allows precise control of IPGs**

![CDF of generated IPDs]

- Specialized NIC
  - Higher variance
- SoNIC
  - Zero variance!!!

![Interpacket delays (ns)]

- 9Gbps, IPD = 13992 bits (1357ns)
Packet Capture

- **SoNIC captures what is sent**

![CDF of captured IPDs](image)

- Interpacket delays (ns)
- 9Gbps, IPD = 13992 bits (1357ns)
Covert Timing Channel

- Embedding signals into interpacket gaps.
  - Large gap: ‘1’
  - Small gap: ‘0’

- Covert timing channel by modulating IPGs at 100ns
  - Overt channel at 3 Gbps
  - Covert channel at 250 kbps
  - Over 4-hops with < 1% BER
Covert Timing Channel

- Modulating IPGS at 100ns scale (=128 /I/s)

Interpacket delays (ns)

CDF

BER = 0.37%

‘0’: 3562 – 128 /I/s

‘1’: 3562 + 128 /I/s

‘0’: 3562 – a /I/s

‘1’: 3562 + a /I/s
Contributions

• Network Research
  – Unprecedented access to the PHY with commodity hardware
  – A platform for cross-network-layer research
  – Can improve network research applications

• Engineering
  – Precise control of interpacket gaps (delays)
  – Design and implementation of the PHY in software
  – Novel scalable hardware design
  – Optimizations / Parallelism

• Status
  – Measurements in large scale: DCN, GENI, 40 GbE
Conclusion

• Precise Realtime Software Access to the PHY

• Commodity components
  – An FPGA development board, Intel architecture

• Network applications
  – Network measurements
  – Network characterization
  – Network steganography

• Webpage: [http://sonic.cs.cornell.edu](http://sonic.cs.cornell.edu)
  – SoNIC is available Open Source.
Before Next time

- Project Interim report
  - Due Monday, November 24.
  - And meet with groups, TA, and professor
- Fractus Upgrade: Should be back online

- **Required review and reading for Friday, November 14**

- Check website for updated schedule