Packet Capture in 10-Gigabit Ethernet Environments
Using Contemporary Commodity Hardware

Fabian Schneider    Jörg Wallerich    Anja Feldmann
{fabian,joerg,anja}@net.t-labs.tu-berlin.de

Technische Universität Berlin
Deutsche Telekom Laboratories

Passive and Active Measurement Conference
5th April 2007
Motivation

Example Scenario: Network security tool at the edge of your network
- need access to packet level data for application layer analysis
- High-speed networks ⇒ high data and packet rate

Challenge: capture full packets without missing any packet
- One approach: specialized hardware
  - e.g. Monitoring cards from Endace
  - Drawbacks: high costs, single purpose

Question: Is it feasible to capture traffic with commodity hardware?
Outline

1 Monitoring 10-Gigabit
   Approach
   Link Bundling

2 Comparing 1-Gigabit Monitoring Systems

3 Results

4 Summary
Approach for 10-Gigabit Monitoring

- **Problem:** No recent host bus or disk system can handle the bandwidth needs of 10-Gigabit environments

- **Solution:** split up traffic and distribute the load (e.g. 10-Gigabit on multiple 1-Gigabit links)
  - Use a switch: e.g. link bundling feature
  - Use specialized hardware

- Keep corresponding data together!
Etherchannel (Cisco) feature enables link-bundling for:

- higher bandwidth, redundancy, …
- or load-balancing e.g. for Webservers

**Feasibility test:**

- Tested on a Cisco 3750
- 1-Gigabit Ethernet link split on eight FastEthernet (100 Mbit/s) links.
- Assign packets to links based on both IP addresses.

⇒ It works with real traffic!
Link Bundling

Load-Balancing

- Simple switches use only MAC addresses
  ⇒ Not useful for a router-to-router link
- On a Cisco 3750: any combination of IP and/or MAC addresses
  ⇒ is sufficient for our example scenario
- On a Cisco 65xx: MAC’s, IP’s, and/or Port Numbers
Comparing 1-Gigabit Monitoring Systems

1. Monitoring 10-Gigabit

2. Comparing 1-Gigabit Monitoring Systems
   - Methodology
   - System under Test
   - Measurement Setup

3. Results

4. Summary
Comparing 1-Gigabit Monitoring Systems

Methodology

- Comparable priced systems with
  - Different processor architectures
  - Different operating systems

- Task of those systems:
  - Capture full packets
  - Do not analyze them (Out-of-Scope)

- Workload:
  - All system are subject to identical input
  - Increase bandwidth up to a fully loaded Gigabit link
  - Realistic packet size distribution

- Measurement Categories:
  - Capturing Rate: number of captured packets (simple libpcap app)
  - System Load: CPU usage while capturing (simple top like app)
Systems under Test

Two examples of any of the systems:

- One installed with Linux
- The other with FreeBSD

First set of systems purchased in 2005:

- 2x AMD Opteron 244 (1 MB Cache, 1.8 GHz),
- 2x Intel Xeon (Netburst, 512 kB Cache, 3.06 GHz),

Second set purchased in 2006:

- 2x Dual Core AMD Opteron 270 (1 MB Cache, 2.0 GHz)

All: 2 Gbytes of RAM, optical Intel Gigabit Ethernet card, RAID array
Measurement Setup

Generator (LKPG)

SNMP Interface
Counter Queries

optical Splitter (multiplies every Signal)

eth0
eth1
eth2

Workload ->

Cisco C3500XL

Linux/
AMD Opteron

Linux/
Intel Xeon
(Netburst)

FreeBSD/
AMD Opteron

FreeBSD/
Intel Xeon
(Netburst)

Control Network
Results

1. Monitoring 10-Gigabit

2. Comparing 1-Gigabit Monitoring Systems

3. Results
   - Using multiple processors?
   - Increasing buffer sizes
   - Additional Insights (I)
   - Write to disk
   - Additional Insights (II)

   \{\text{first set of systems measurements}\}
   \{\text{second set of systems measurements}\}

4. Summary
Using multiple processors?

Results

Single processor, 1st Set

Upper Part: Capturing Rate

Lower Part: CPU Usage

SP: 100% corresponds to one fully utilised processor

MP: 50% corresponds to one fully utilised processor

X-Axis: Generated Bandwidth

Datarate [Mbit/s]

Capturing Rate [%]

CPU usage [%]

(32) no SMP, no HT, std. buffers, 1 app, no filter, no load

Linux/AMD
Linux/Intel
FreeBSD/AMD
FreeBSD/Intel

Capturing Rate [%]

CPU usage [%]

X-Axis: Generated Bandwidth

Datarate [Mbit/s]

Capturing Rate [%]

CPU usage [%]
Results

Using multiple processors?

Schneider, Wallerich, Feldmann (TU Berlin/DT Labs)
Packet Capturing on 10-Gigabit Ethernet Links
PAM 2007 12 / 20

Single processor, 1\textsuperscript{st} Set

(32) no SMP, no HT, std. buffers, 1 app, no filter, no load

Capturing Rate [%] vs. Datarate [Mbit/s]

Linux/AMD
Linux/Intel
FreeBSD/AMD
FreeBSD/Intel
Capturing Rate [%]
CPU usage [%]
(32) no SMP, no HT, std. buffers, 1 app, no filter, no load

**Single processor, 1\textsuperscript{st} Set**

- **Opteron/FreeBSD system performs best**
- **Sharp decline at high data rates**
Multiprocessor (SMP), 1st Set

(31) SMP, no HT, std. buffers, 1 app, no filter, no load

<table>
<thead>
<tr>
<th>Datarate [Mbit/s]</th>
<th>Capturing Rate [%]</th>
<th>CPU usage [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>150</td>
<td></td>
<td></td>
</tr>
<tr>
<td>200</td>
<td></td>
<td></td>
</tr>
<tr>
<td>250</td>
<td></td>
<td></td>
</tr>
<tr>
<td>300</td>
<td></td>
<td></td>
</tr>
<tr>
<td>350</td>
<td></td>
<td></td>
</tr>
<tr>
<td>400</td>
<td></td>
<td></td>
</tr>
<tr>
<td>450</td>
<td></td>
<td></td>
</tr>
<tr>
<td>500</td>
<td></td>
<td></td>
</tr>
<tr>
<td>550</td>
<td></td>
<td></td>
</tr>
<tr>
<td>600</td>
<td></td>
<td></td>
</tr>
<tr>
<td>650</td>
<td></td>
<td></td>
</tr>
<tr>
<td>700</td>
<td></td>
<td></td>
</tr>
<tr>
<td>750</td>
<td></td>
<td></td>
</tr>
<tr>
<td>800</td>
<td></td>
<td></td>
</tr>
<tr>
<td>850</td>
<td></td>
<td></td>
</tr>
<tr>
<td>900</td>
<td></td>
<td></td>
</tr>
<tr>
<td>950</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Using multiple processors?

Multiprocessor (SMP), 1\textsuperscript{st} Set

All systems are benefitting . . .

. . . even though the second processor is not used extensively

(31) SMP, no HT, std. buffers, 1 app, no filter, no load

Capturing Rate [%]

CPU usage [%]

Datarate [Mbit/s]
Increasing Buffer Sizes?

Setup:

- First Set of systems
- Dual processor
- Increased buffer sizes

Operating system buffers:

**FreeBSD 6.x:** sysctl’s net.bpf.bufsize and net.bpf.maxbufsize

**FreeBSD 5.x:** sysctl’s debug.bpf_bufsize and debug.maxbpf_bufsize

**Linux:** /proc/sys/net/core/rmem_default, /proc/sys/net/core/rmem_max, and /proc/sys/net/core/netdev_max_backlog
increased buffers, 1st Set

(17) SMP, no HT, inc. buffers, 1 app, no filter, no load

Capturing Rate [%]
CPU usage [%]

Datarate [Mbit/s]
increased buffers, 1st Set

The capturing rate could be increased again
Additional Insights

First set of measurements

- Filtering is cheap with respect to its benefit (reduced packet processing)
- Running multiple capturing applications concurrently leads to bad performance.
- Measurement with additional compression show some advantage for Intel Systems
- Intel Hyperthreading does not change the performance
- using the memory-map patch from Phil Woods (Linux only) does help
Writing packets to disk?

Preliminary measurements have shown that

- newer system do not lose any packet: with buffers, SMP, etc.
- disk writing speed is not the bottleneck

Setup:

- Newer systems: 2x dual core AMD systems
  \[\Rightarrow\] CPU usage: 25% correspond to one fully utilized processor
- Increased buffer sizes
- No filter
- Linux vs. FreeBSD
- 32bit vs. 64bit OS’es
(2-8) SMP, no HT, inc. buffers, 1 app, no filter, writing to disk

Writing to disk, 2\textsuperscript{nd} Set
(2-8) SMP, no HT, inc. buffers, 1 app, no filter, writing to disk

Writing to disk, 2\textsuperscript{nd} Set

Feasible up to 600-700 Mbit/s!
Additional Insights

Second set of measurements

- additional load (copying the packets in memory) shows significantly better performance for FreeBSD
- 64bit systems drop more packets
- Using 4 cores (2x Dual Core) is slightly better than 2 cores (1x Dual Core)
Summary

• Split up 10-Gigabit on multiple 1-Gigabit monitoring systems
• FreeBSD/AMD Opteron combination in general performs best
• Utilizing multiple processors proves to be benefitting
• Choosing large enough buffer size is important
• Capturing full traces to disk is feasible up to about 600-700 Mbit/s

For further information see: High Performance Packet Capture
http://www.net.t-labs.tu-berlin.de/research/hppc/