QA Workshop (Nov 2013)
Testing Kernel Network Stack and NIC Drivers

Liang Zheng
Kernel QA
lzheng@suse.com

Agenda

- Tuning and Debug Tools
- How to Maximize Network Throughput
- Test cases
- Questions

Monitor/Debug Tools

Monitor/Debug Tools

- netstat – per nic status,errors,statistics at driver level
- vmstat – vm page info,context swtich,total ints/s,cpu
- lspci – list the devices on pci,indepth driver flags
- modinfo – list information about drivers,version,options
- mpstat – reveals per cpu stats,Hard/Soft Interrupt usage
- oprofile – system level profiling,kernel/driver code
- tcpdump – packet analyzer

Tuning Tools

Tuning Tools

- Ethtool – View and change Ethernet card settings
- Netperf – a benchmark that can be used to measure the performance networking
- Sysctl – View and set /proc/sys settings
- Ifconfig – View and set ethX variables
- Setpci – View and set pci bus params for device
- /proc – OS info,place for changing device tunables

Ethtool

- ethtool -s Queries the specified network device for NIC and driver-specific statistics
- ethtool -c Show interrupt coalesce
- ethtool -i Show the driver information
- ethtool -g Show rx/tx ring buffers information
- ethtool -k Show the HW offload settting information

Ethtool -s --Statistics

```
linux-kvm:~ # ethtool -S eth0
NIC statistics:
     rx_bytes: 38177376167
     rx_error_bytes: 0
     tx_bytes: 3187892780
     tx_error_bytes: 0
     rx_ucast_packets: 19600106
     rx_mcast_packets: 58591418
     rx_bcast_packets: 3367439
     tx_ucast_packets: 11819206
     tx_mcast_packets: 683811
     tx_bcast_packets: 2090
<truncated>
```

Ethtool -c Interrupt Coalesce

```
linux-kvm:~ # ethtool -c eth0
Coalesce parameters for eth0:
Adaptive RX: off  TX: off
stats-block-usecs: 999936
sample-interval: 0
pkt-rate-low: 0
pkt-rate-high: 0

rx-usecs: 18
rx-frames: 12
rx-usecs-irq: 18
rx-frames-irq: 2

tx-usecs: 80
tx-frames: 20
tx-usecs-irq: 18
tx-frames-irq: 2
```

Ethtool -i Driver Information

driver: bnx2

version: 2.1.11

firmware-version: bc 5.2.3 NCSI 2.0.6

bus-info: 0000:03:00.0

supports-statistics: yes

supports-test: yes

supports-eeprom-access: yes

supports-register-dump: yes

Ethtool -g HW Ring Buffers

```
linux-kvm:~ # ethtool -g eth0
Ring parameters for eth0:
Pre-set maximums:
RX:         2040
RX Mini:    0
RX Jumbo:   8160
TX:         255
Current hardware settings:
RX:         255
RX Mini:    0
RX Jumbo:   0
TX:         255
```

Ethtool -k HW Offload Settings

```
linux-kvm:~ # ethtool -k eth0
Offload parameters for eth0:
rx-checksumming: on
tx-checksumming: on
scatter-gather: on
tcp-segmentation-offload: on
udp-fragmentation-offload: off
generic-segmentation-offload: on
generic-receive-offload: on
large-receive-offload: off
rx-vlan-offload: on
tx-vlan-offload: on
ntuple-filters: off
receive-hashing: on
```

sysctl

- Sysctl is a mechanism to show and set the entries under the /proc/sys tree
- Sysctl -a --list all variables
- Sysctl -q --queries a variable
- Sysctl -w --write a variable

Some Important Settings for sysctl

- Misc TCP protocol
  - net.ipv4.tcp_window_scaling -toggles window scaling
  - net.ipv4.tcp_timestamps -toggles TCP timestamp support
  - net.ipv4.tcp_sack -toggles SACK(Selective ACK support)
- TCP Memory Allocations
  - net.ipv4.tcp_rmem -TCP read buffer – in bytes
  - net.ipv4.tcp_wmen -TCP write buffer – in bytes
  - net.ipv4.tcp_mem -TCP buffer space -measured in pages

Some Important Settings for sysctl (cont.)

- CORE memory settings
  - net.core.rmem_max -max size of rx socket buffer
  - net.core.wmem_max -max size of tx socket buffer
  - net.core.rmem_default -default rx size of socket buffer
  - net.core.wmem_default -default tx size of socket buffer
  - net.core.optmem_max -maximum amount of option memory buffer
- These settings also impact UDP

netperf

- http://netperf.org
- Default test in TCP_STREAM – uses send() call
- TCP_SENDFILE -uses sendfile() call – much less copying
- TCP_RR -Request/Response tests
- UDP_STREAM

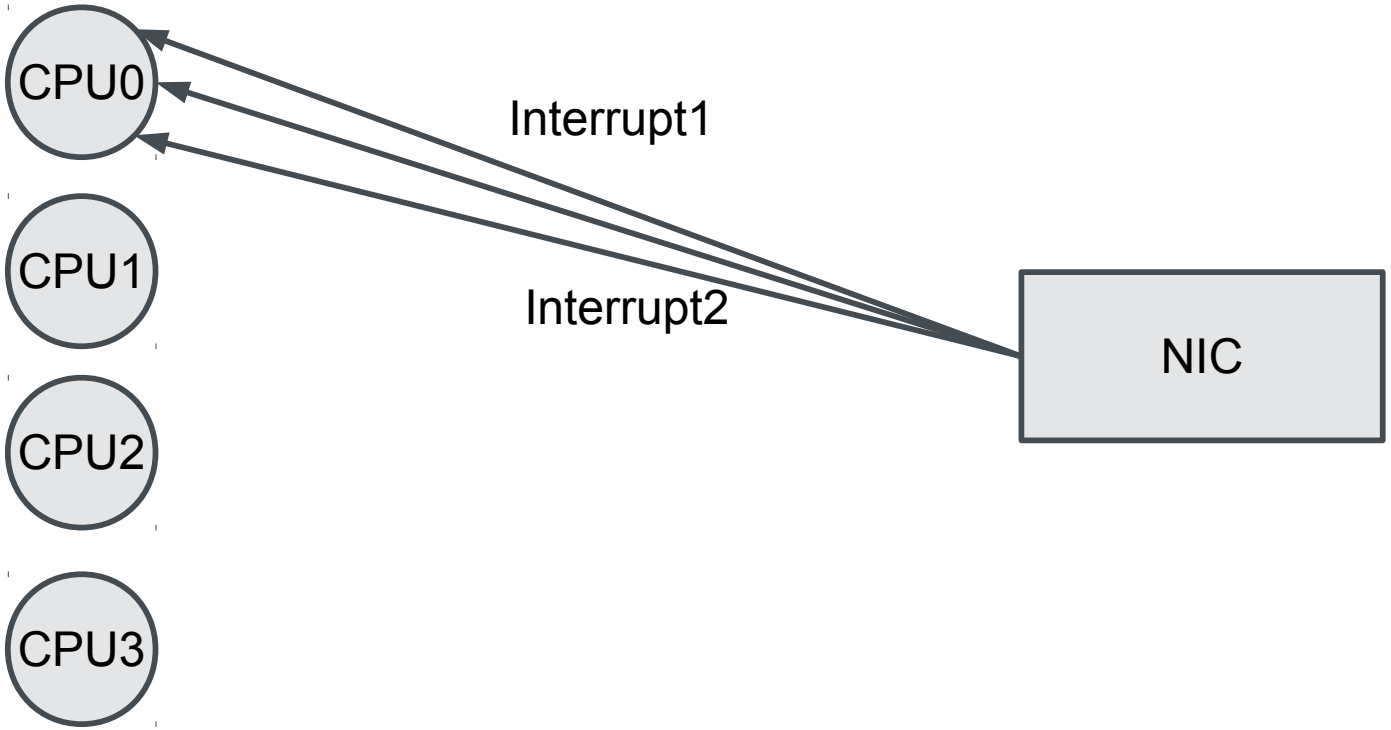
How to Maximize Network Throughput

Performance Tuning Outline

- Enable SMP IRQ Affinity
- Enable RPS+RFS,for transmit node, enable XPS
- Increase/decrease memory parameter for Network
- Driver Setting
  - HW ring buffers
    - ethtool -G <dev>
  - Enable TSO,GRO,GSO,UFO,LRO etc, HW offload
    - ethtool -K <dev> tso on gro on gso on
  - Tuning interrupt coalescing
    - ethtool -C <dev>
  - Enable jumbo frame, for example,set mtu to 9000
    - ifconfig <dev> mtu 9000

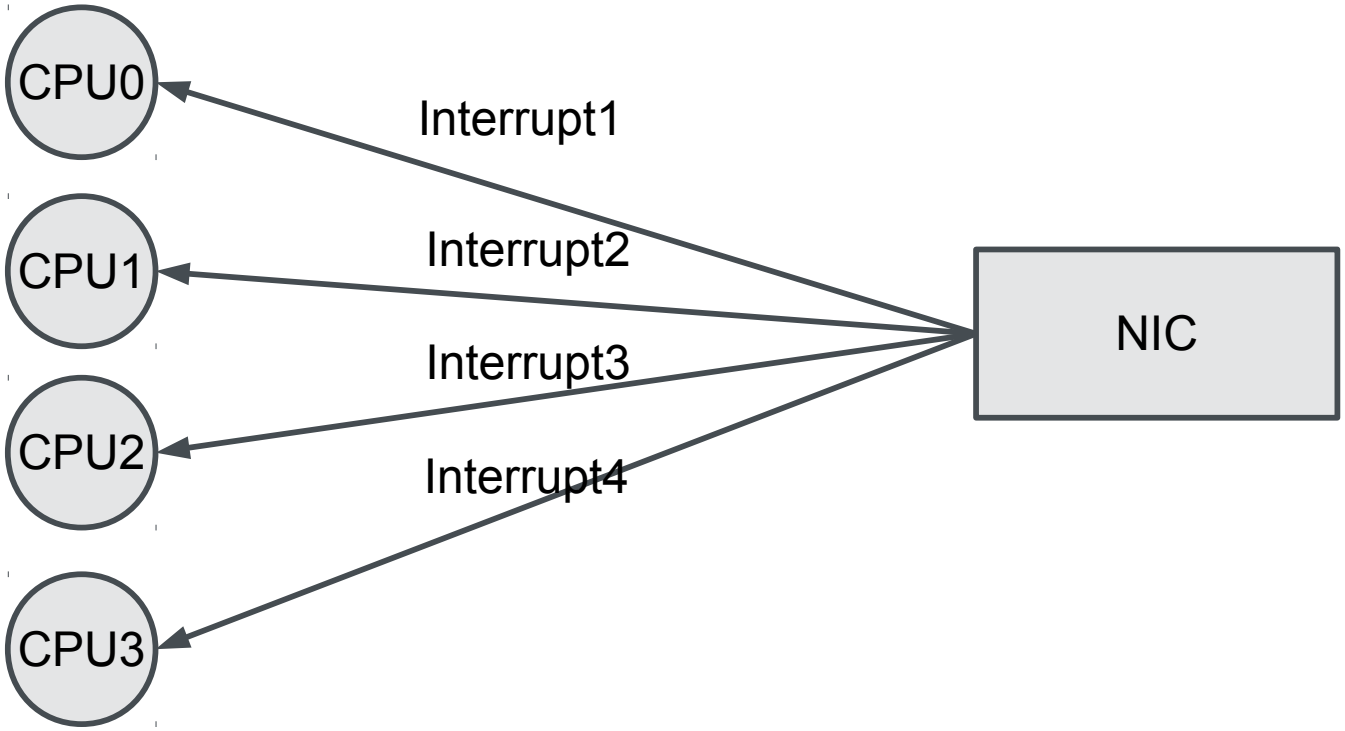
SMP IRQ Affinity

- Before use SMP IRQ Affinity



SMP IRQ Affinity

- After use SMP IRQ Affinity



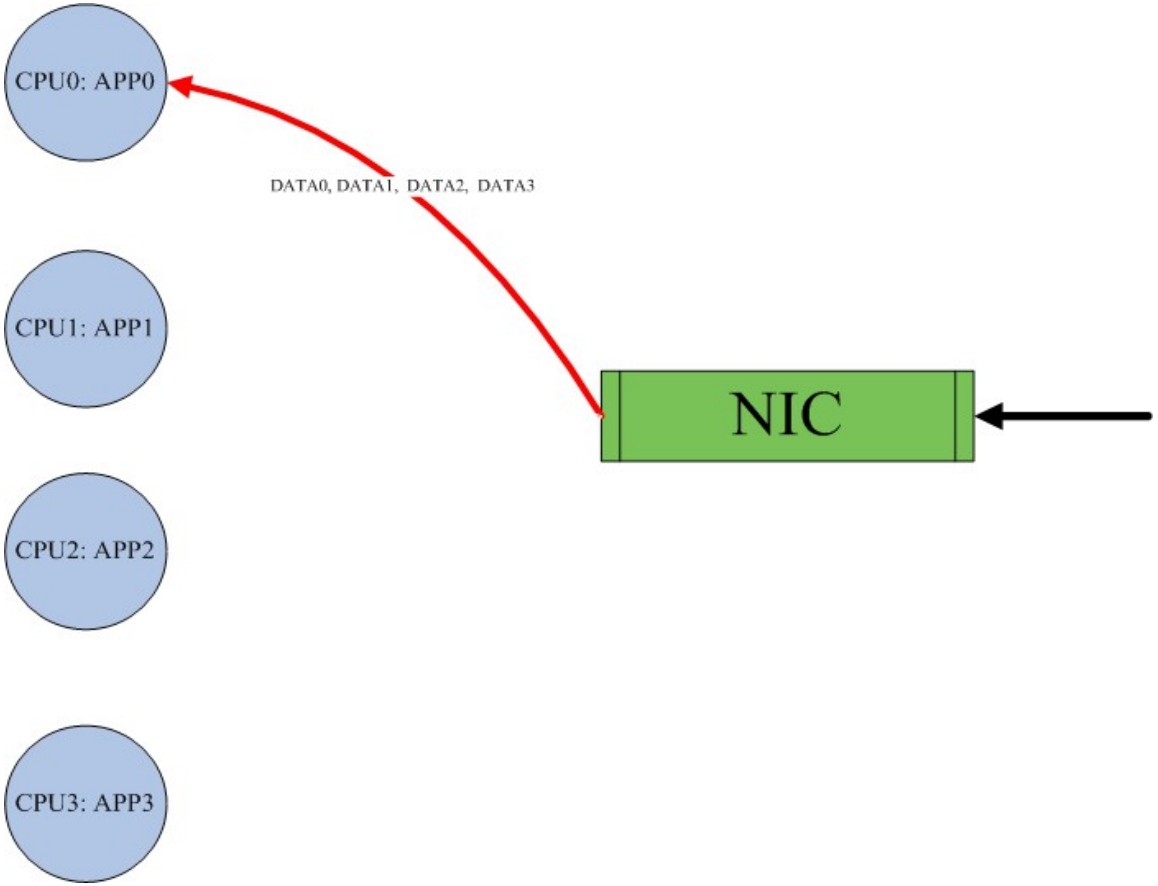
Setup SMP IRQ Affinity

- grep eth0 /proc/interrupt
- echo 80 > /proc/irq/81/smp_affinity
- For example:

```
/proc/irq/74/smp_affinity 000001
/proc/irq/75/smp_affinity 000002
/proc/irq/76/smp_affinity 000004
/proc/irq/77/smp_affinity 000008
/proc/irq/78/smp_affinity 000010
/proc/irq/79/smp_affinity 000020
/proc/irq/80/smp_affinity 000040
/proc/irq/81/smp_affinity 000080
```

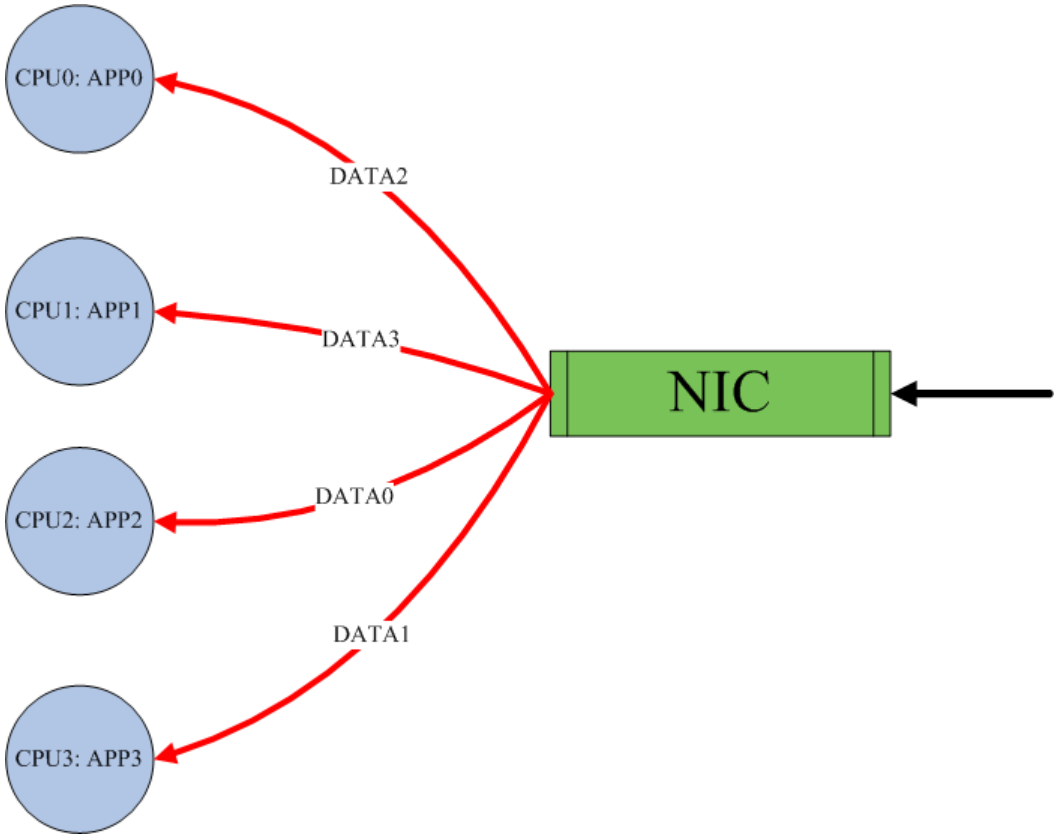
Receive Packet Steering (RPS)

Before enable RPS:



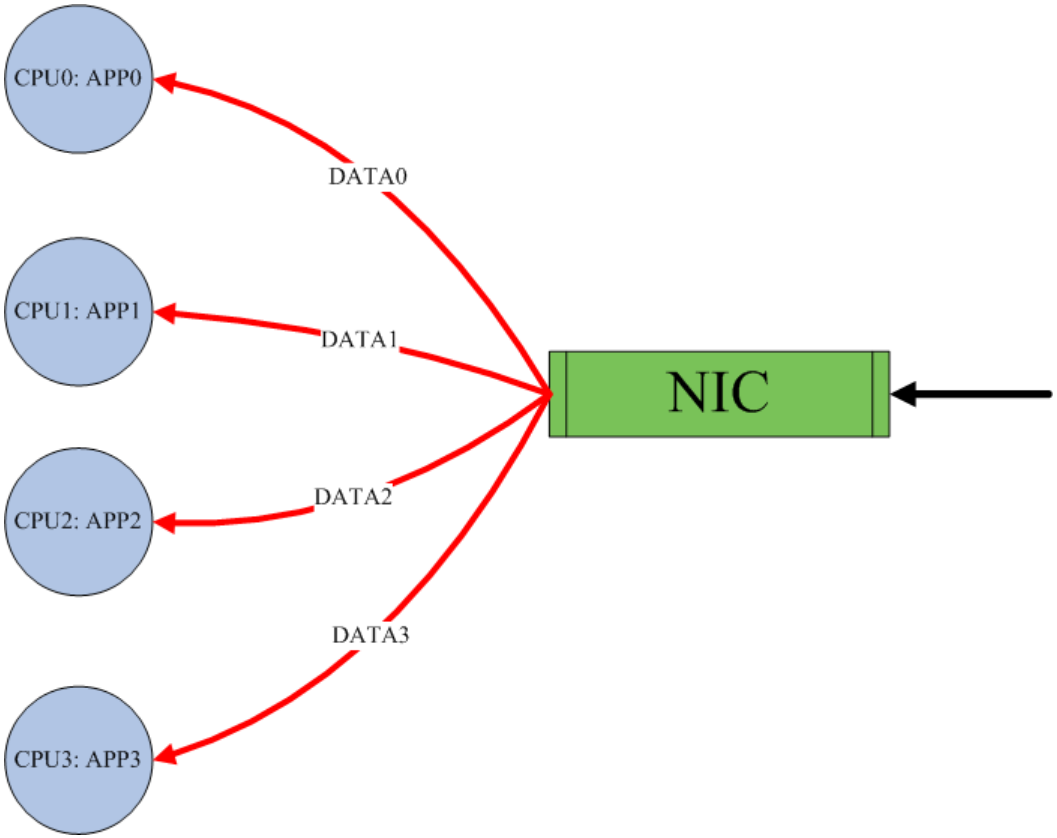
Receive Packet Steering (RPS)

After enable RPS:
echo numEntries > /sys/class/net/<dev>/queues/rx-<n>/rps_cpus



Receive Flow Steering (RFS)

Enable both RPS+RFS
echo numEntries > /sys/class/net/<dev>/queues/rx-<n>/rps_flow_cnt



Transmit Packet Steering (XPS)

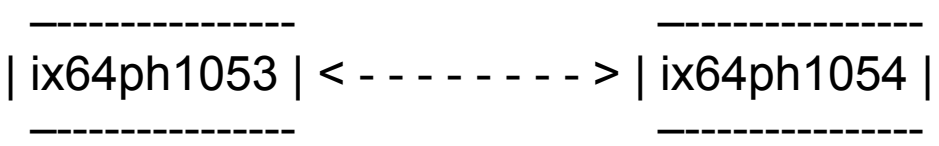
- For send/transimt node, we can enable XPS
- Under sysfs we have xps_cpus entry.
- For example, for tx-0

  /sys/class/net/eth0/queues/tx-0/xps-cpus

Test Cases

Test Scenario

1. Platform : SUSE Linux Enterprise Server 11 SP3 (x86_64)
2. Test NIC driver: sfc
3.Kernel version : 3.0.76-0.11-default
4.Test topo:
    The two machines are back-to-back connected with a fiber cable.



Case 1 – tcp_window_scaling

```
ix64ph1053:~/netperf-2.6.0# sysctl -w net.ipv4.tcp_window_scaling=0
```

```
ix64ph1053:~/netperf-2.6.0 # netperf -H 192.168.1.2
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to
192.168.1.2 () port 0 AF_INET
enable_enobufs failed: setsockopt
Recv   Send    Send
Socket Socket  Message  Elapsed
Size   Size    Size     Time     Throughput
bytes  bytes   bytes    secs.    10^6bits/sec

 87380  16384  16384    10.00    3997.91
```

Case 1 – tcp_window_scaling

```
ix64ph1053:~/netperf-2.6.0# sysctl -w net.ipv4.tcp_window_scaling=1
```

```
ix64ph1053:~/netperf-2.6.0 # netperf -H 192.168.1.2
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to
192.168.1.2 () port 0 AF_INET
enable_enobufs failed: setsockopt
Recv   Send    Send
Socket Socket  Message  Elapsed
Size   Size    Size     Time     Throughput
bytes  bytes   bytes    secs.    10^6bits/sec

 87380  16384  16384    10.00    9194.36
```

Case 2 – Kick up MTU = 9000

#./netperf -P1 -l 30 -H 192.168.10.10 -T 5,5 -t TCP_SENDFILE -F /data.file

Recv Send Send

Socket Socket Message Elapsed

Size Size Size Time Throughput

bytes bytes bytes secs. 10^6bits/sec

87380 16384 16384 30.00 9888.66

Case 3 – Lower RX Latency

```
# ethtool -c eth6
Coalesce parameters for eth6:
<truncate>
rx-usecs: 125
rx-frames: 0
rx-usecs-irq: 0 rxframesirq: 0
```

```
# ./netperf -H 192.168.10.12 -t TCP_RR
Local /Remote
Socket Size Request Resp. Elapsed Trans.
Send Recv Size Size Time Rate
bytes Bytes bytes bytes secs. per sec

16384 87380 1 1 10.00 8000.27
```

Lower rx-usecs on the receiver and rerun

```
# ethtool -C eth6 rx-usecs 100
```

```
# ./netperf -H 192.168.10.12 -t TCP_RR

16384 87380 1 1 10.00 10009.83
```

Case 4 – Turn off Generic Receive Offload

ethtool -K eth5 gro off
ix64ph1054:~/netperf-2.6.0 # ethtool -k eth5 | grep generic-receive-offload
generic-receive-offload: off

ix64ph1053:~/netperf-2.6.0 # netperf -H 192.168.1.2
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to 192.168.1.2 () port 0 AF_INET
enable_enobufs failed: setsockopt

Recv Socket Size bytes	Send Socket Size bytes	Send Message Size bytes	Elapsed Time secs.	Throughput 10^6bits/sec
87380	16384	16384	10.00	6196.54

Case 4 – Turn on Generic Receive Offload

```
# ethtool -K eth5 gro on
ix64ph1054:~/netperf-2.6.0 # ethtool -k eth5 | grep generic-receive-offload
generic-receive-offload: on
```

```
ix64ph1053:~/netperf-2.6.0 # netperf -H 192.168.1.2
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to
192.168.1.2 () port 0 AF_INET
enable_enobufs failed: setsockopt
Recv   Send    Send
Socket Socket  Message  Elapsed
Size   Size    Size     Time     Throughput
bytes  bytes   bytes    secs.    10^6bits/sec

 87380  16384  16384    10.00    9183.48
```

Case 5 – Turn off TCP Segmentation Offload

On the sender node:
ethtool -K eth5 tso off

ix64ph1053:~/netperf-2.6.0 # netperf -H 192.168.1.2
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to 192.168.1.2 () port 0 AF_INET
enable_enobufs failed: setsockopt
Recv Send Send
Socket Socket Message Elapsed
Size Size Size Time Throughput
bytes bytes bytes secs. 10^6bits/sec

87380 16384 16384 10.00 5842.46

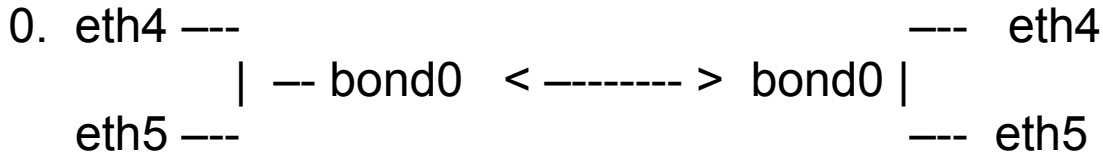
Case 5 – Turn on TCP Segmentation Offload

On the sender node:
ethtool -K eth5 tso on

ix64ph1053:~/netperf-2.6.0 # netperf -H 192.168.1.2
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to 192.168.1.2 () port 0 AF_INET
enable_enobufs failed: setsockopt
Recv Send Send
Socket Socket Message Elapsed
Size Size Size Time Throughput
bytes bytes bytes secs. 10^6bits/sec

87380 16384 16384 10.00 9133.79

Case 6 – Bonding Performance



1. Load bonding module with active-backup mode
modprobe bonding mode=1 miimon=100

2. Setup the bond0 interface with a private ipv4 address
ifconfig bond0 192.168.2.1/24 up

3. Enslave eth4 & eth5 (sfc driver) into the bond0 interface
ifenslave bond0 eth4 eth5

4. Run netperf to check the network throughput.

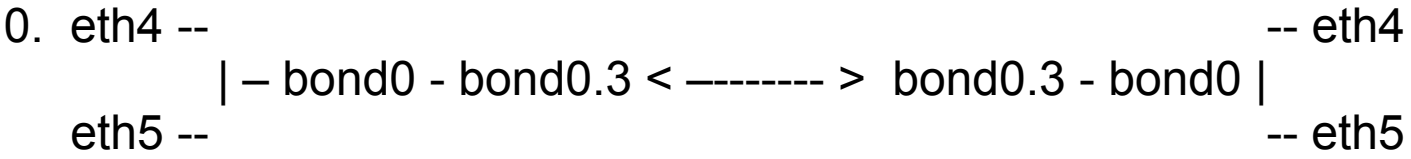
Case 6 – Bonding Performance

1. Test result:

```
ix64ph1053:~ # netperf -H 192.168.3.2
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to
192.168.3.2 () port 0 AF_INET
enable_enobufs failed: setsockopt
Recv   Send    Send
Socket Socket  Message  Elapsed
Size   Size    Size     Time     Throughput
bytes  bytes   bytes    secs.    10^6bits/sec

 87380  16384  16384    10.00    7470.45
```

Case 7 – Bonding/VLAN Performance



1. Load bonding module with active-backup mode
modprobe bonding mode=1 miimon=100

2. Setup the bond0
ifconfig bond0 up

3. Enslave eth4 & eth5 (sfc driver) into the bond0 interface
ifenslave bond0 eth4 eth5

4. Add a VLAN interface on top of the bond0
#vconfig add bond0 3

5. Setup the bond0.3 a ipv4 address
#ifconfig bond0.3 192.168.3.1/24 up

6. Run netperf to check the network throughput.

Case 7 – Bonding/VLAN Performance

```
ix64ph1053:~ # netperf -H 192.168.3.2
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to
192.168.3.2 () port 0 AF_INET
enable_enobufs failed: setsockopt
Recv   Send    Send
Socket Socket  Message  Elapsed
Size   Size    Size     Time     Throughput
bytes  bytes   bytes    secs.    10^6bits/sec

 87380  16384  16384    10.00    2470.45 < - - - Oops
```

We find a performance bug here!!!

Case 7 – Bonding/VLAN Performance

TODO:

1. Is it a regression bug ? Does it issue on SLES SP2 or earlier version ?
2. Does it issue on other drivers, or just on sfc driver ?
3. Does it issue on the upstream kernel?
4. What cause this bug ?

For more details, go to:

qa.suse.de/workshop

Thank you.

Corporate Headquarters
Maxfeldstrasse 5
90409 Nuremberg
Germany

+49 911 740 53 0 (Worldwide)
www.suse.com

Join us on:
www.opensuse.org