Kernel/Hardware for bifrost

- Linux for infrastructure
  - Robustness, robustness, performance
  - No chance to support all HW or SW
  - Selection in lab
  - Very time consuming process
    - Costly, need resources and needs support & skill

Kernel/Hardware for bifrost

- We still on use Opteron
  - Now Shanghai 2382 or close
  - Motherboard TYAN 2915, 2923, two NUMA nodes
  - Memory config to reach 128 bit transfers
  - Chassies, redundant power
  - USB boot seems OK.
  - No recommendations for low or mid range HW!!

Kernel/Hardware for bifrost

- NIC's (Recommended)
  - 10g Intel 82598 chips fixed 10GBASE-SR
  - SUN neptune 10g/1g, XFP modules
  - Intel 82576. GIGE. TP
  -
  - XFP-LR can drive fibre 40km or more
    - Tested at KTH/CSD
  - 10GBASE-T not seen yet
  - Hot-Lava SFP board?

Kernel/Hardware for bifrost

- Drivers
- Critical. Drivers and Kernel support
  - Almost crititial. Open chip documentation
  -
  - Multiqueue. RSS a la MS NDIS 6.0 and later
  - Ixgbe, niu, igb (e1000, e1000e, tulip)
  - Issues: Optical Statistics, DOM etc

Kernel/Hardware for bifrost

- Kernel selection
- Long time monitor and test. Code Freeze.
- Now 2.6.29-rc2 from DaveM git with many pathes

```
do {
   modify_and_patch();
   happy = test();
} while( ! happy);
```

Kernel/Hardware for bifrost

- Multiqueue efforts landed.
  - Needs: NIC, Driver, Affinity, Understanding

Linux Network
framework for MO
Thanks, DaveM

eth-affinity

cat /proc/interrups
IRQ/DMA
consistant naming

driver patches.
ixgbe, niu, igb

Kernel/Hardware for bifrost

- Multiqueue efforts landed.
  - HW classifier splits incoming based on hash etc to different MSI-X IRQ vectors (For RX)
  - We set IRQ affinity so:
    - RXQ1 → CPU1
    - RXQ2 → CPU2 etc. This done automaticly by eth-affinity it can be done due the consistent naming in /proc/interrupts

Kernel/Hardware for bifrost

- Multiqueue efforts landed.
  - At RX the driver records the RX queue in the skb

```
@@ -3815,6 +3824,8 @@ static void igb_receive_skb(struct igb_ring *ring, u8 status,
        struct igb_adapter * adapter = ring->adapter;
        bool vlan_extracted = (adapter->vlgrp && (status & E1000_RXD_STAT_VP));

+        skb_record_rx_queue(skb, rp->rx_channel);
+
```

Kernel/Hardware for bifrost

- Multiqueue efforts landed.
  - At TX the driver selects the TX queue according to RX

```
+static u16 select_queue(struct net_device *dev, struct sk_buff *skb)
+{
+       if( dev->real_num_tx_queues && skb_rx_queue_recorded(skb) )
+               return  skb_get_rx_queue(skb) % dev->real_num_tx_queues;
+
+       return  smp_processor_id() %  dev->real_num_tx_queues;
+}
+
```

Kernel/Hardware for bifrost

- Multiqueue efforts landed

Of course we have also set IRQ affinity so:

TXQ1 → CPU1

TXQ2 → CPU2 etc. This done automaticly by eth-affinity it can be done due the consistent naming in /proc/interrupts

Kernel/Hardware for bifrost

- Multiqueue efforts landed
- This OK for fowarding...
- Packets Per Sec scales with No CPU Cores
- Detailed numbers in the IIS report
- Ixia measued roughly 2.8Mpps Duplex 3.5 Mpps simplex.
- 8.6 Gbits/s (1.8 Mpps) with Internet traffic load with simplex forwarding.

Kernel/Hardware for bifrost

- Known issues
  - Intel new 82599 chip not supported.
  - quagga netlink. 32 vs 64 bit kernel
    - Do we need quagga for 64 bit? No real problem

Kernel/Hardware for bifrost

- New directions for development & research?
-
- Explore advanced classifier benefits
  - Control Plane, Route w/o dst cache etc?
  -
- Energy
  - Low-Power routing and networking

Time for Questions!