PacketShader: A GPU-Accelerated Software Router

Some images and sentence are from original author Sangjin Han's presentation.

Presenter: Hao Lu

Why? What? How?

- Why used software routers ?
- What is GPU ?
- Why use GPU ?
- How to use GPU ?
- What is PacketShader’s design ?
- How is the performance ?
- If have time, configuration of the system.

Software Router

- Not limited to IP routing
  - You can implement whatever you want on it.
- Driven by software
  - Flexible
- Based on commodity hardware
  - Cheap

What is GPU?

- Graph process units.

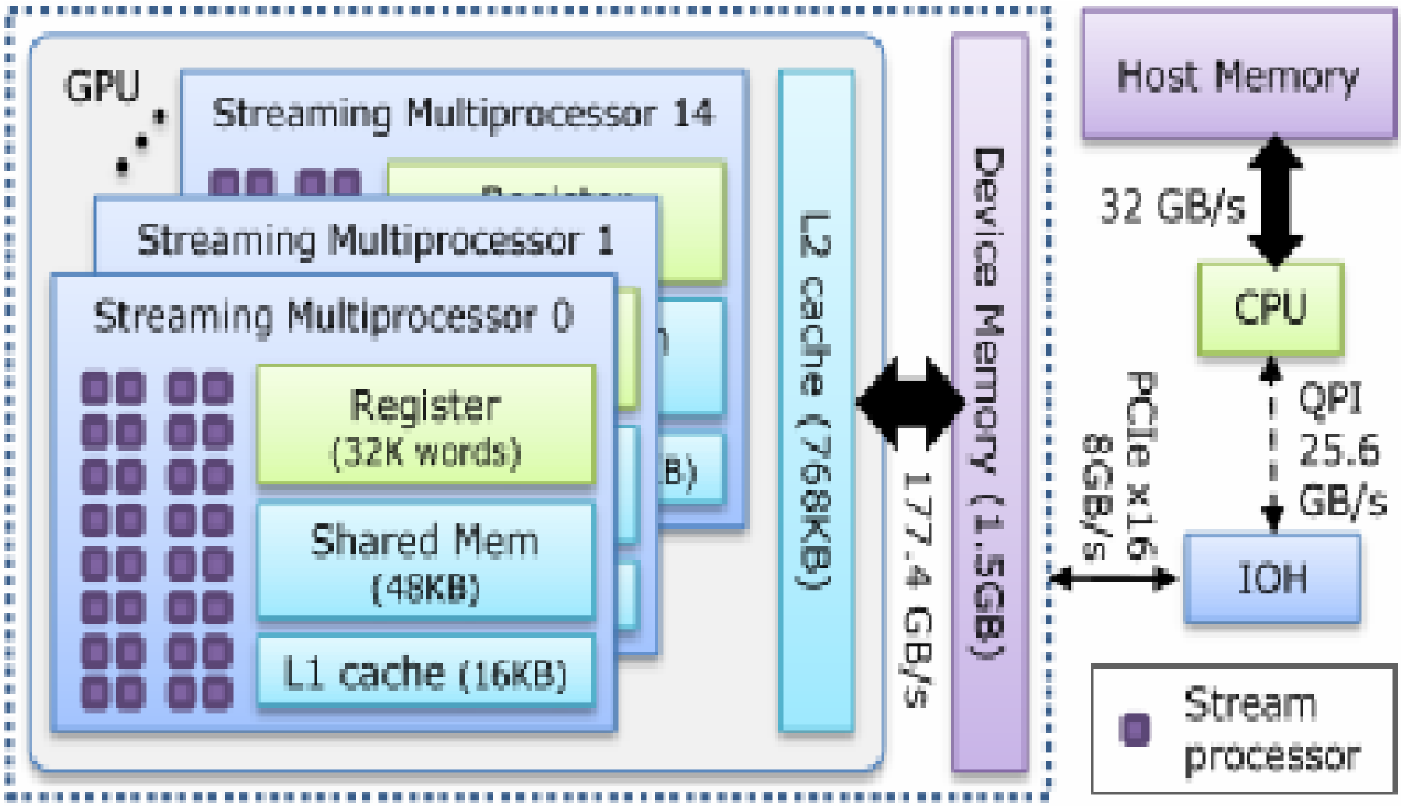


Figure 1: Architecture of NVIDIA GTX480

- 15 Streaming Multiprocessors consist 32 processors = 480 cores

Why use GPU?

Benefit:

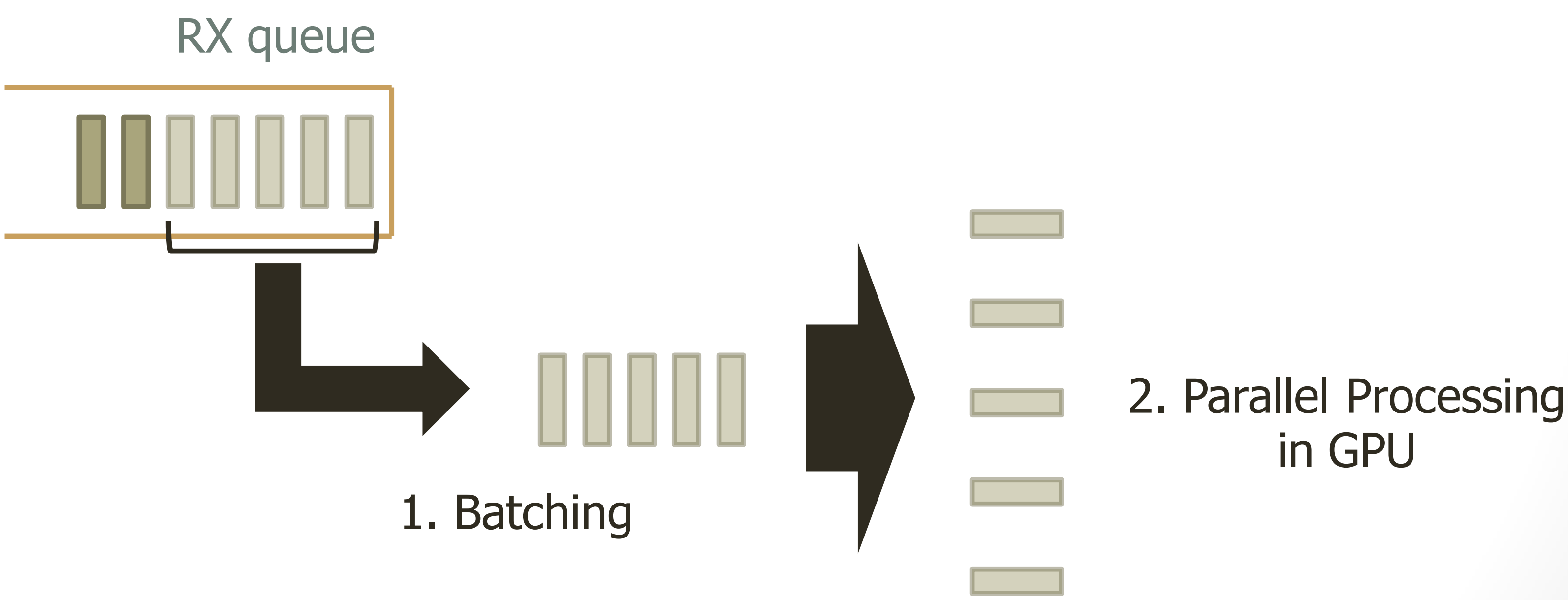
- Higher computation power
  - 1-8 v.s. 480
- Memory access latency
  - Multi-thread to hide the latency
  - CPU has miss register (up to 6)
- Memory bandwidth
  - 32GB v.s. 177GB

Down Sides:

- Thread start latency
- Data transfer rate

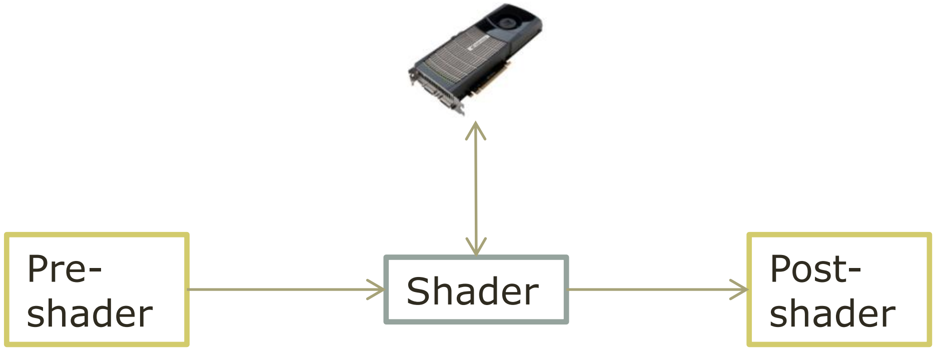
How to use GPU?

- GPU is used for highly parallelizable tasks.
- With enough threads to hide the memory access latency

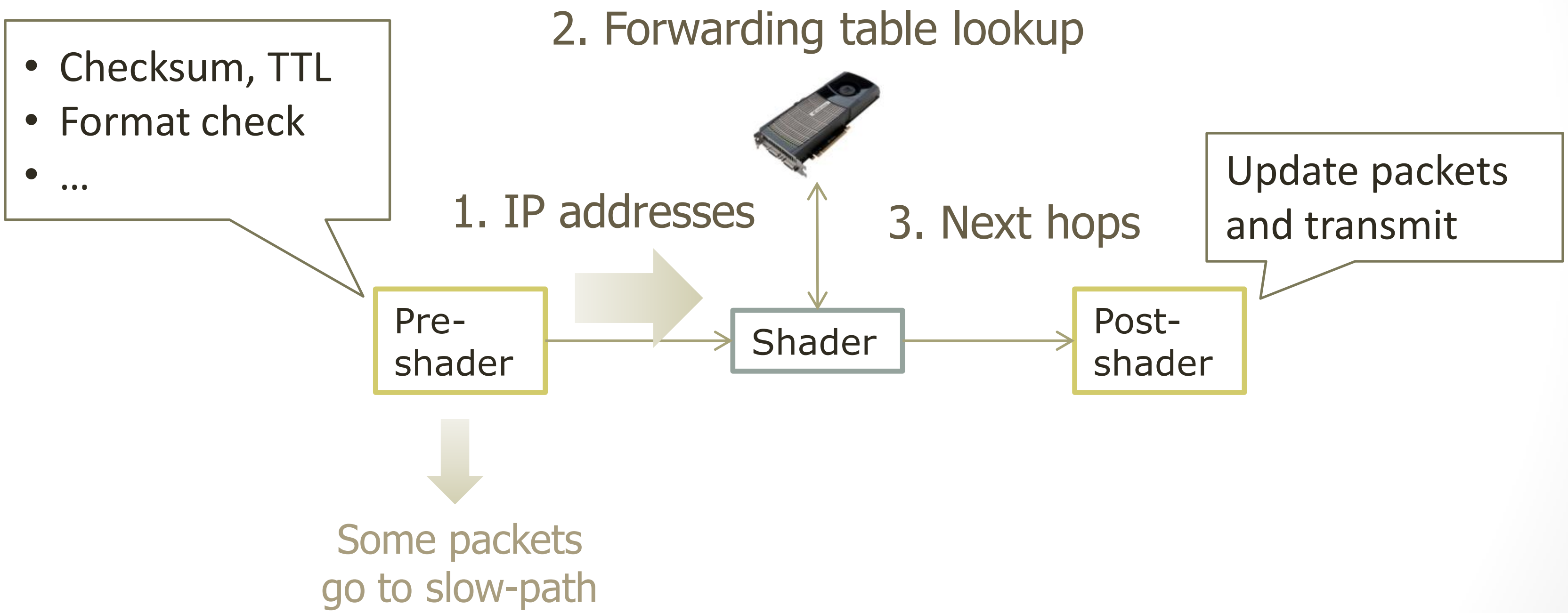


PacketShader Overviw

- Three stages in a streamline
  - Pre-shader
    - Fetching packets from RX queues.
  - Shader
    - Using the GPU to do what it need to be done
  - Post-shader
    - Gather the result and scatter to each TX queue

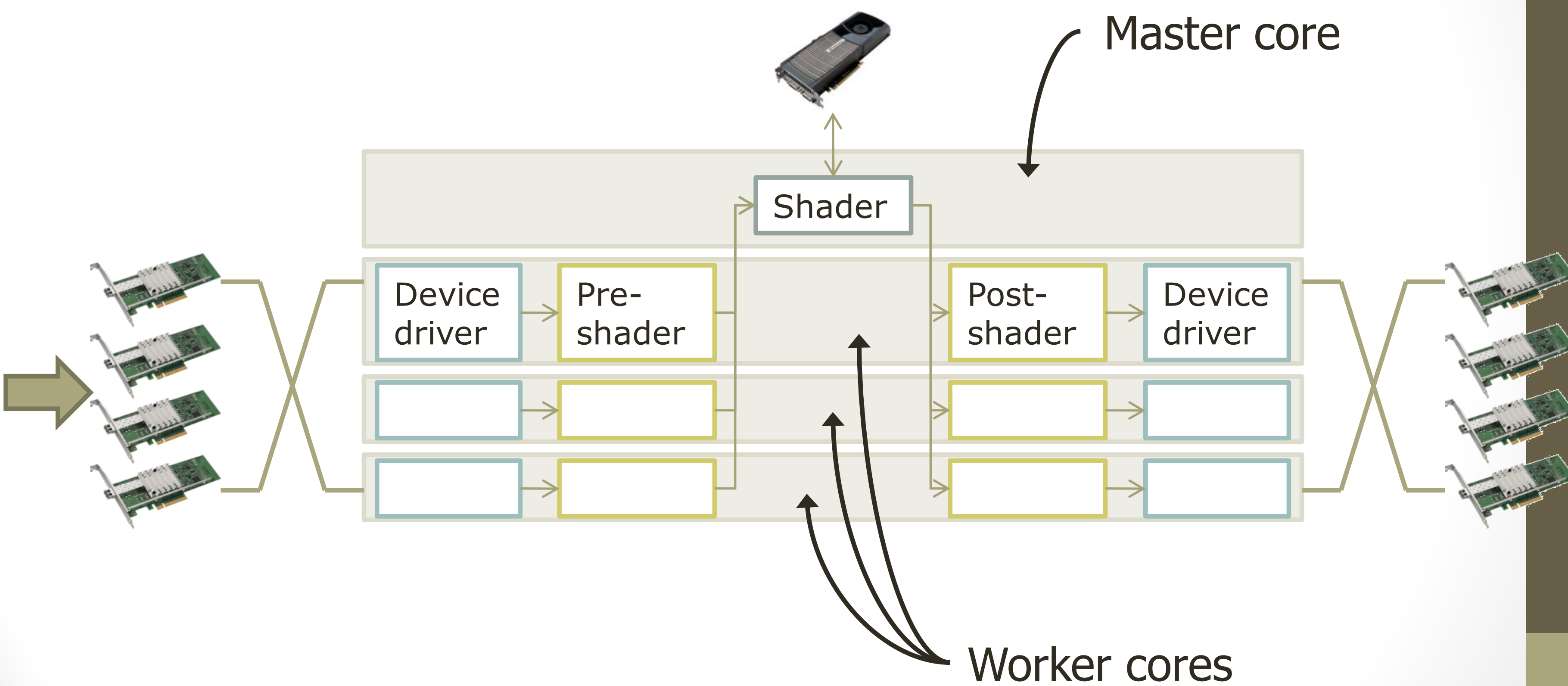


IPv4 Forwarding Example



Scaling with Muti-Core CPU

- Problems:
  - GPU are not as efficient if more than one CPU access it.



Another view

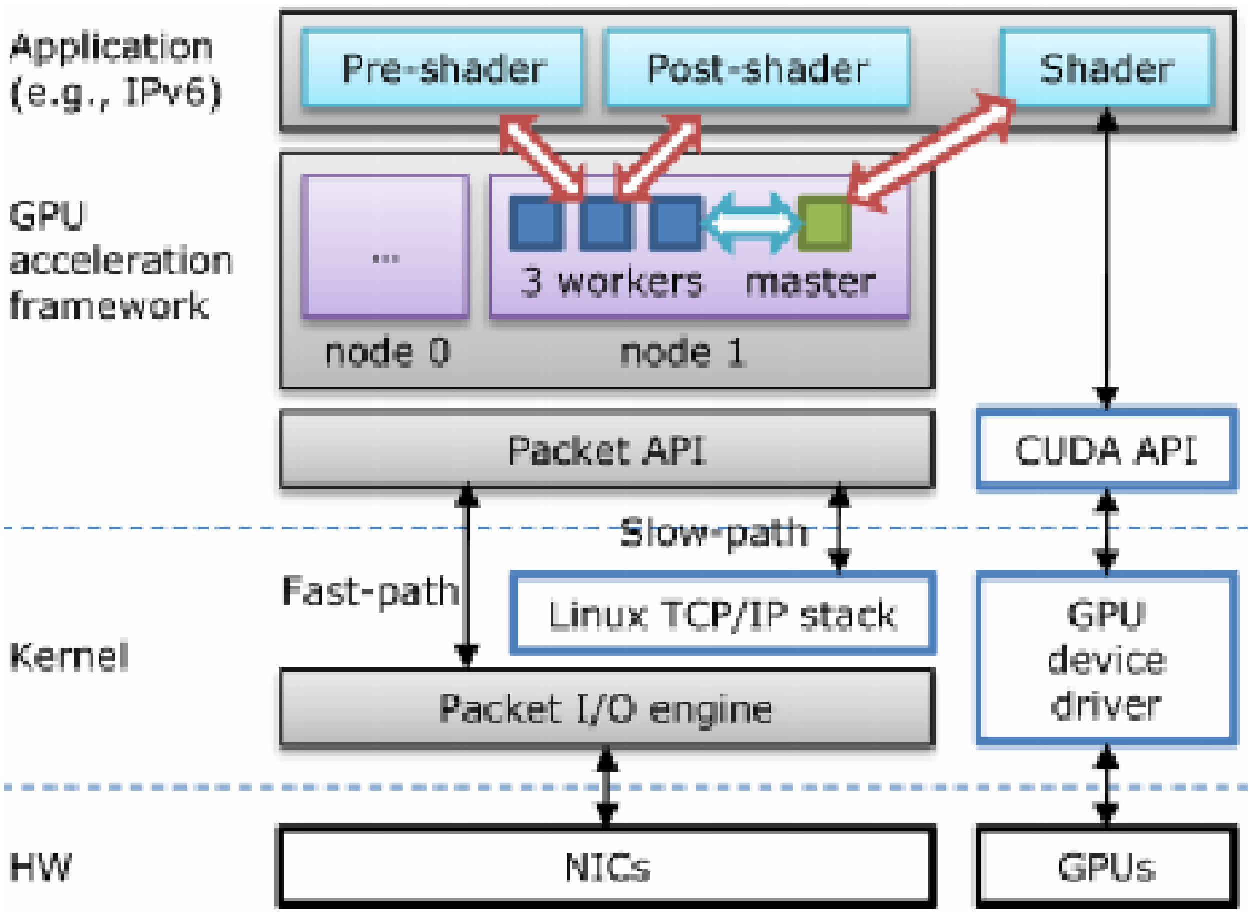
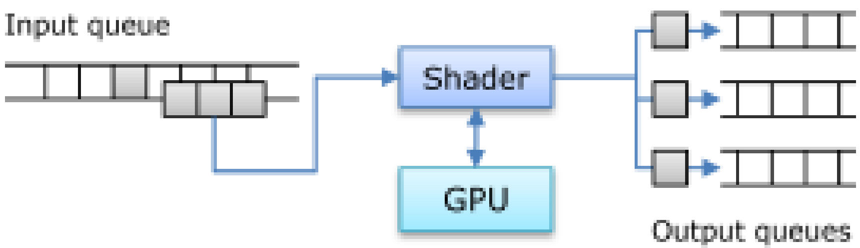


Figure 7: PacketShader software architecture

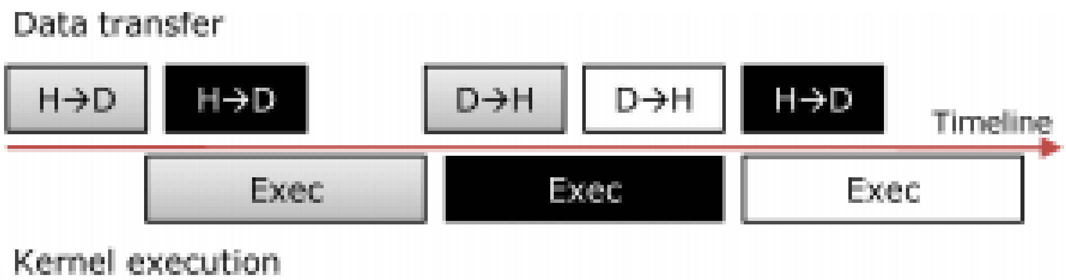
Optimization

- Chuck Pipelining:

- Gather/Scatter



- Concurrent Copy and Execution



Performance: hardware

Item	Specification	Qty	Unit price
CPU	Intel Xeon X5550 (4 cores, 2.66 GHz)	2	$925
RAM	DDR3 ECC 2 GB (1,333 MHz)	6	$64
M/B	Super Micro X8DAH+F	1	$483
GPU	NVIDIA GTX480 (480 cores, 1.4 GHz, 1.5 GB)	2	$500
NIC	Intel X520-DA2 (dual-port 10GbE)	4	$628

Table 2: Test system hardware specification (total $7,000)

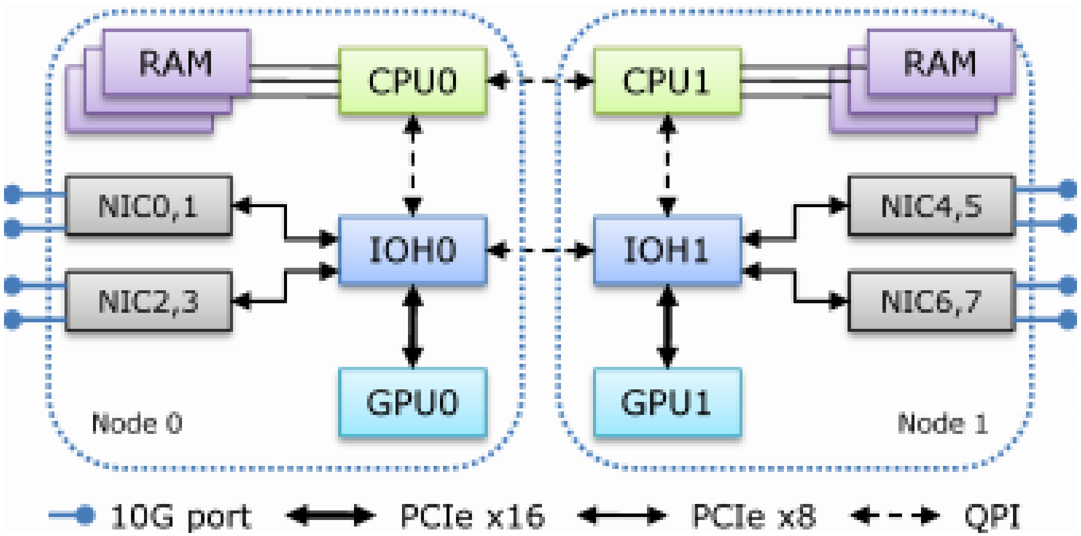
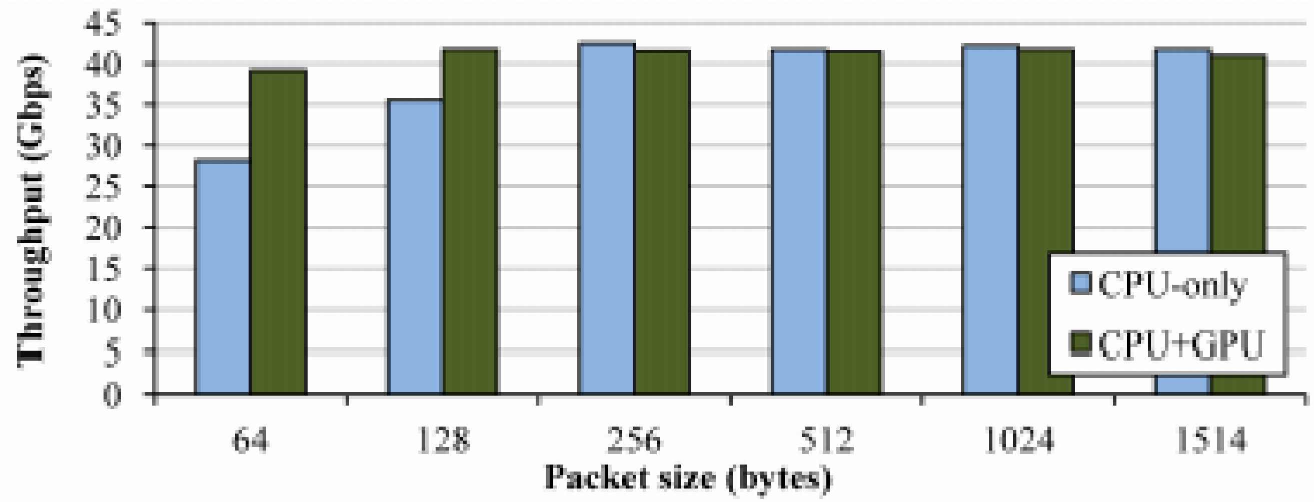


Figure 3: Block diagram of our server

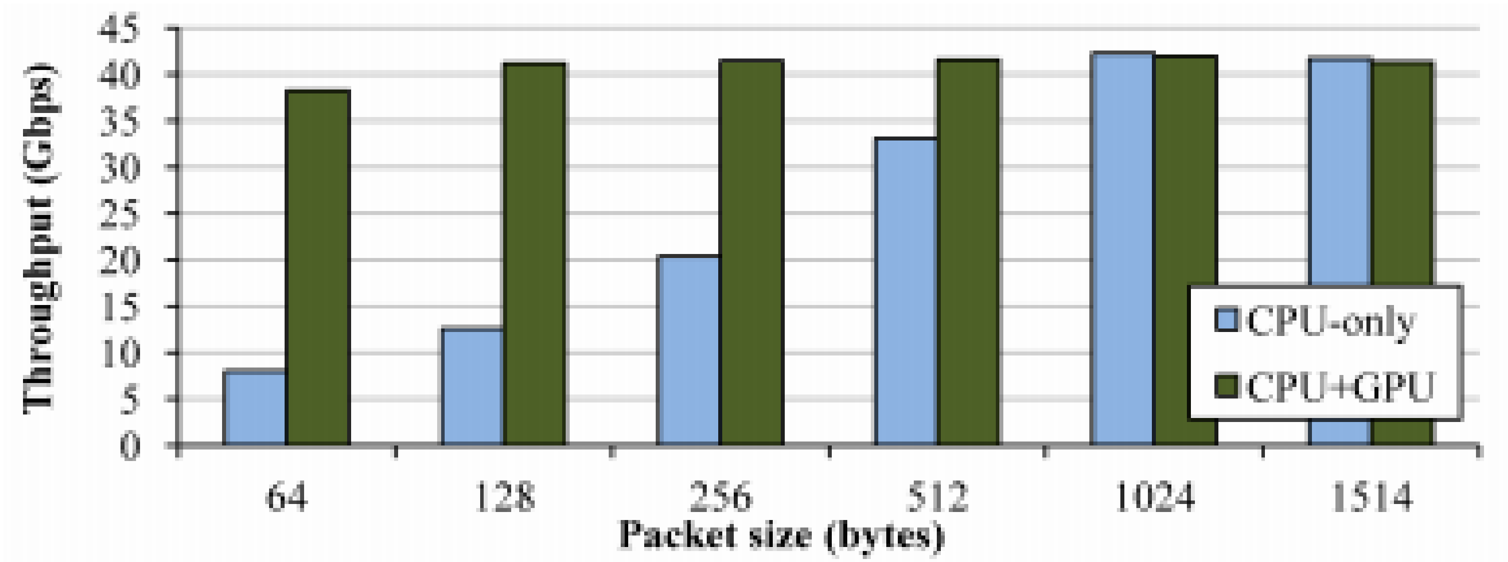
Performance: IPv4 Forwarding

- Algorithm: DIR-24-8-BASIC
  - It requires one memory access per packet for most cases, by storing next-hop entries for every possible 24-bit prefix.
- Pre-shade :
  - Require slow path => Linux TCP/IP stack
  - Else, Update TTL and checksum.



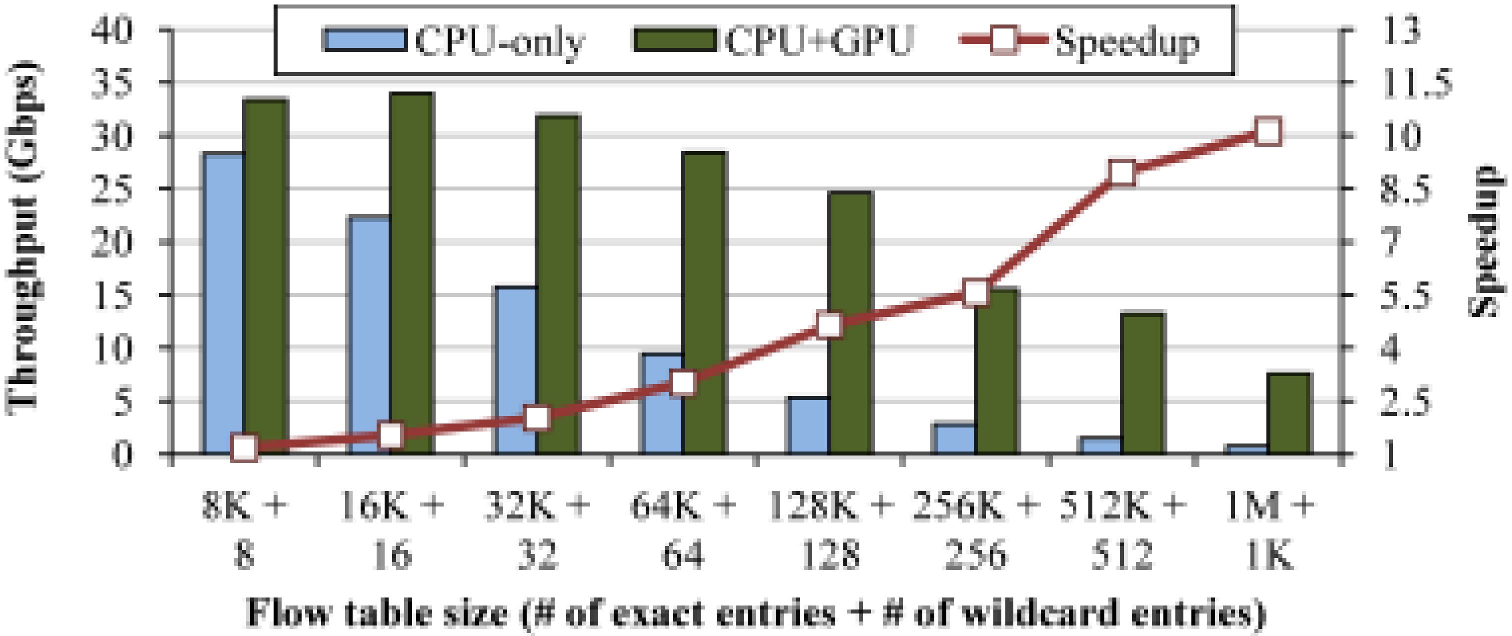
Performance: IPv6 Forwarding

- Same idea of IPv4, more memory access



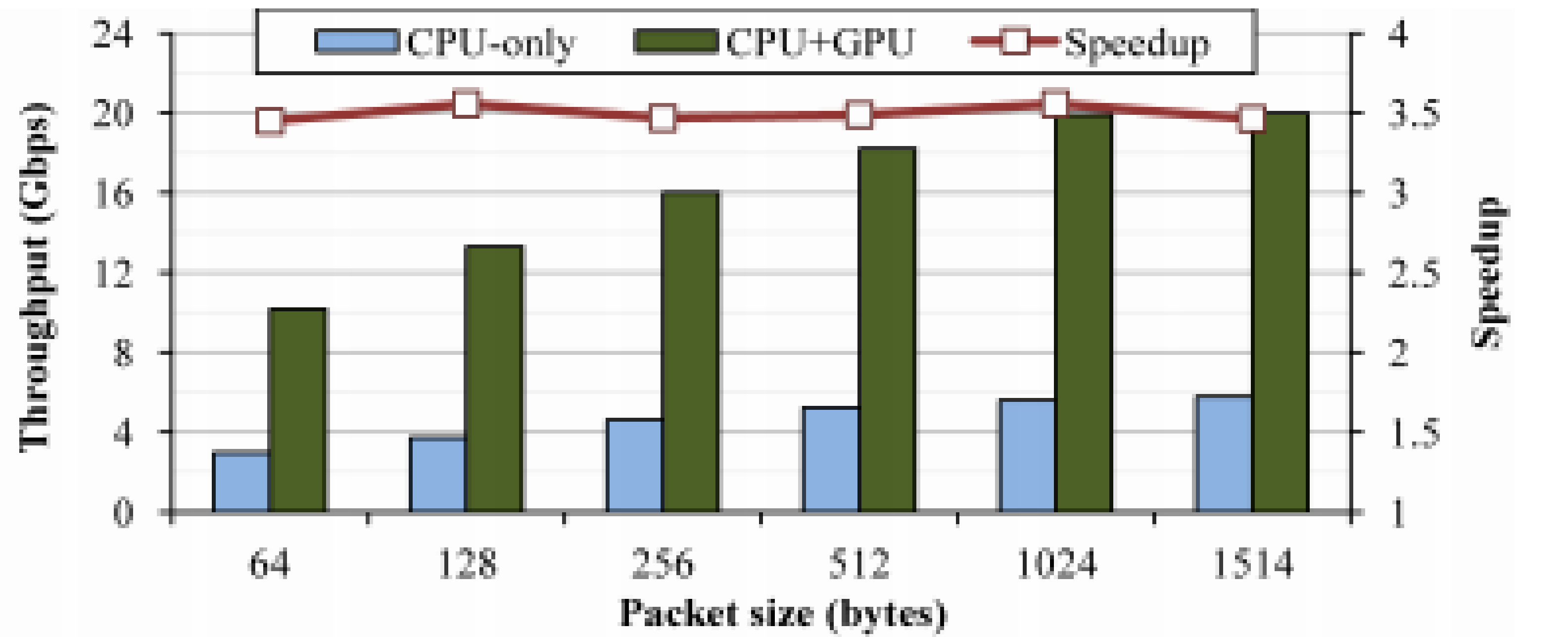
Performance: OpenFlow

- OpenFlow is a framework that runs experimental protocol s over existing networks. Packets are processed on a flow basis.
- The OpenFlow switch is responsible for packet forwarding driven by flow tables.



Performance: IPsec

- IPsec is widely used to secure VPN tunnels or for secure communication between two end hosts.
- Cryptographic operations used in IPsec are highly compute-intensive



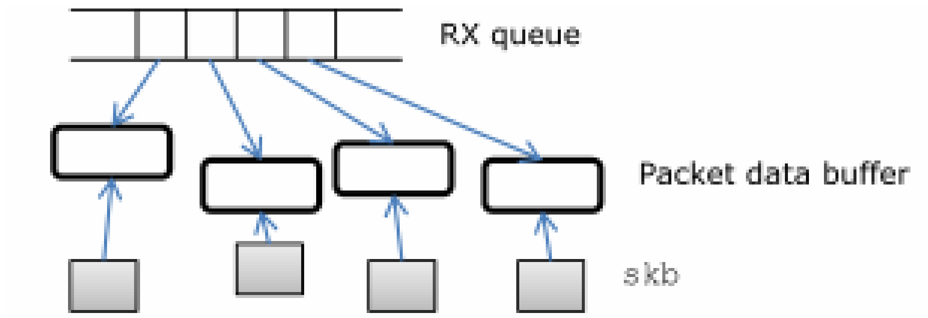
Configuration of the System

- Problem:
  1. Linux Network Stack Inefficiency.
  2. NUMA (None uniform memory access)
  3. Dual-IOH Problem

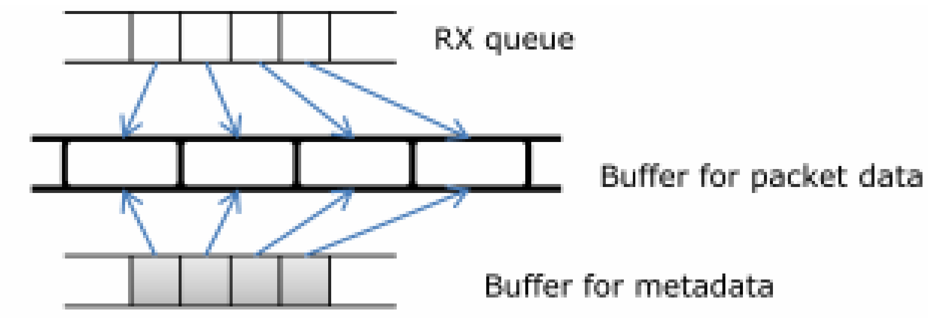
- Solutions:
  1. Better Driver, use Huge Packet Buffer
  2. NUMA aware driver
  3. In research

Network Stack Inefficiency

1. Frequent allocation/deallocation memory
2. skb too large (208 bytes)



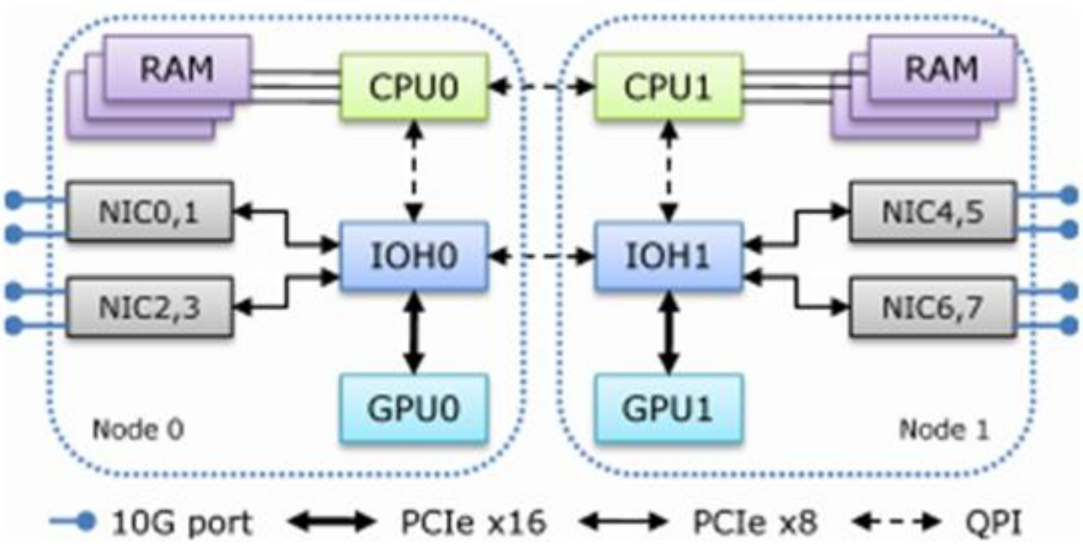
(a) Linux packet buffer allocation



(b) Huge packet buffer allocation

NUMA

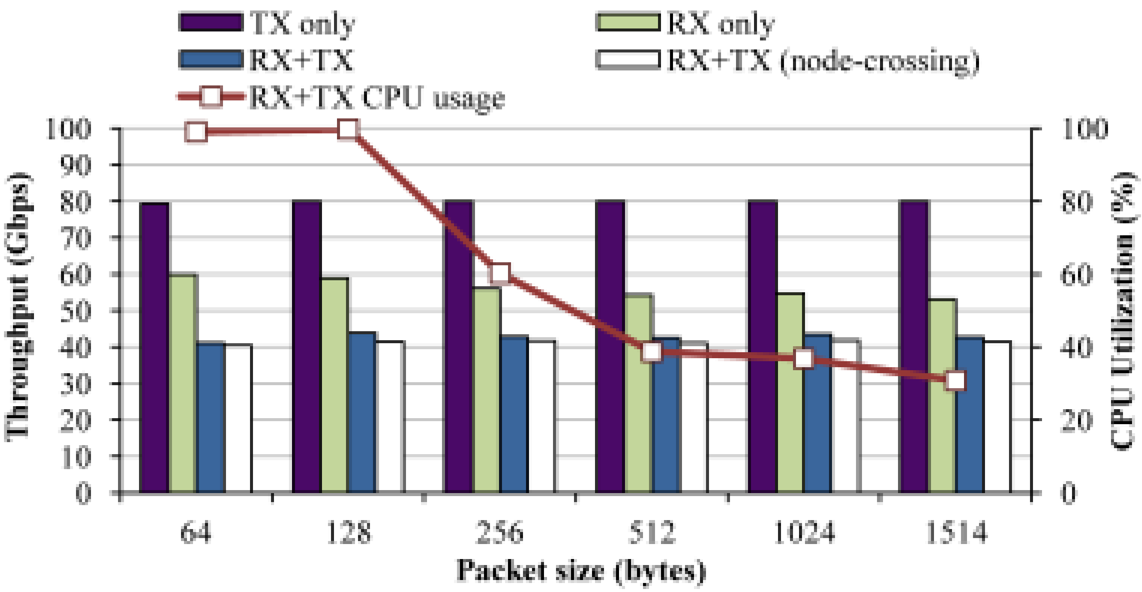
- None Uniform Memory Access due to RSS.



- Solution : Reconfigure RSS to we configure RSS to distribute packets only to those CPU cores in the same node as the NICs

Dual-IOH Problem

- Asymmetry on Data transfer rate.



- Cause: Unknown!!