Network Policy Enforcement with Commodity Multiqueue NICs for Multi-Tenant Data Centers

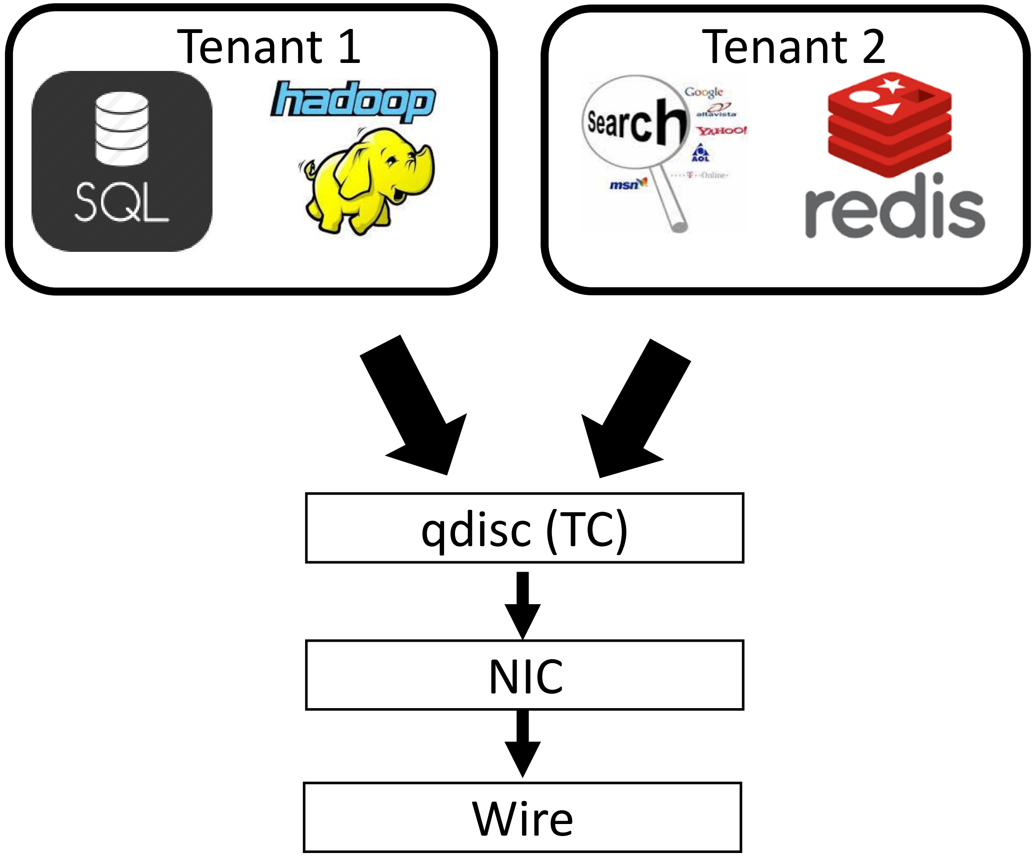
IEEE Internet of Things Journal, April 2022.

Gyuyeong Kim and Wonjun Lee

KOREA UNIVERSITY

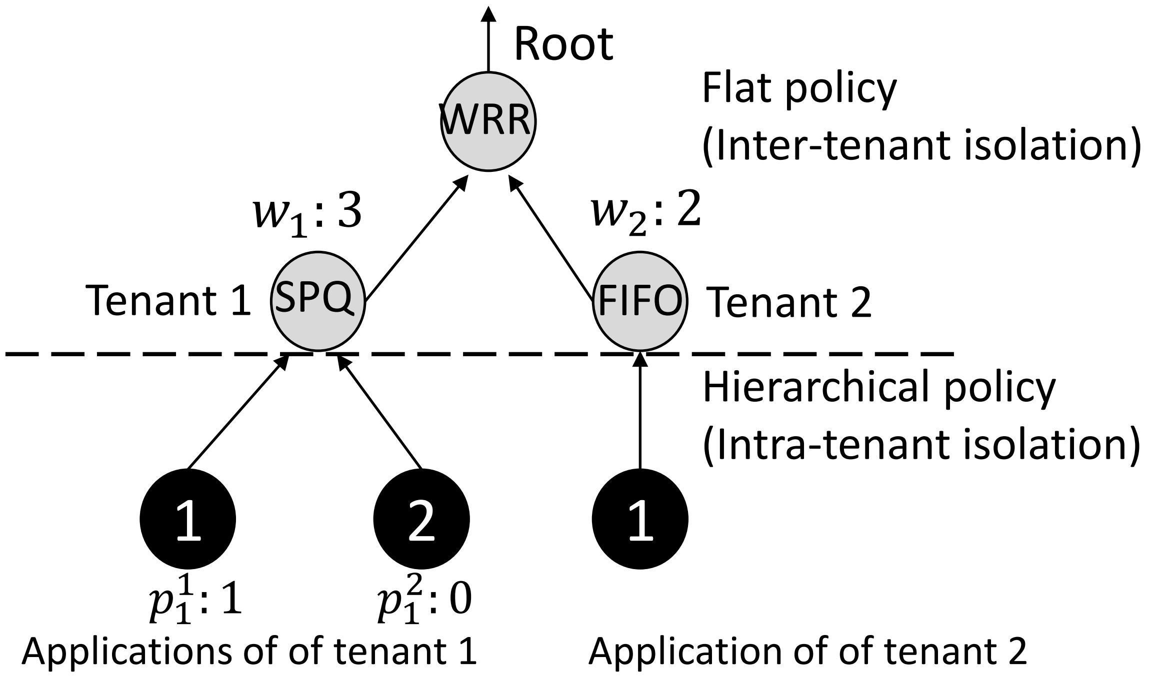
Motivation – End-Hosts in Multi-Tenant DCNs

- End-hosts are the first place where tenants collide in multi-tenant DCNs



Motivation – Network Policy in End-Hosts

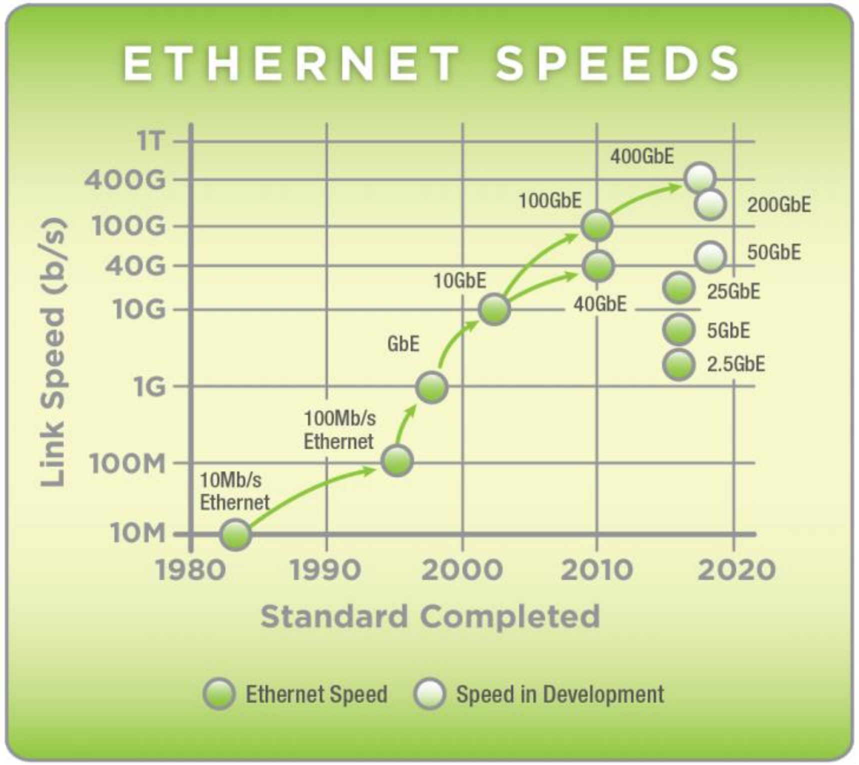
- **Inter-tenant isolation:** a flat policy specifies weights among tenants
- **Intra-tenant isolation:** a hierarchical policy generally specifies priority among applications within a tenant



Motivation – Increase of NIC Line-Rates

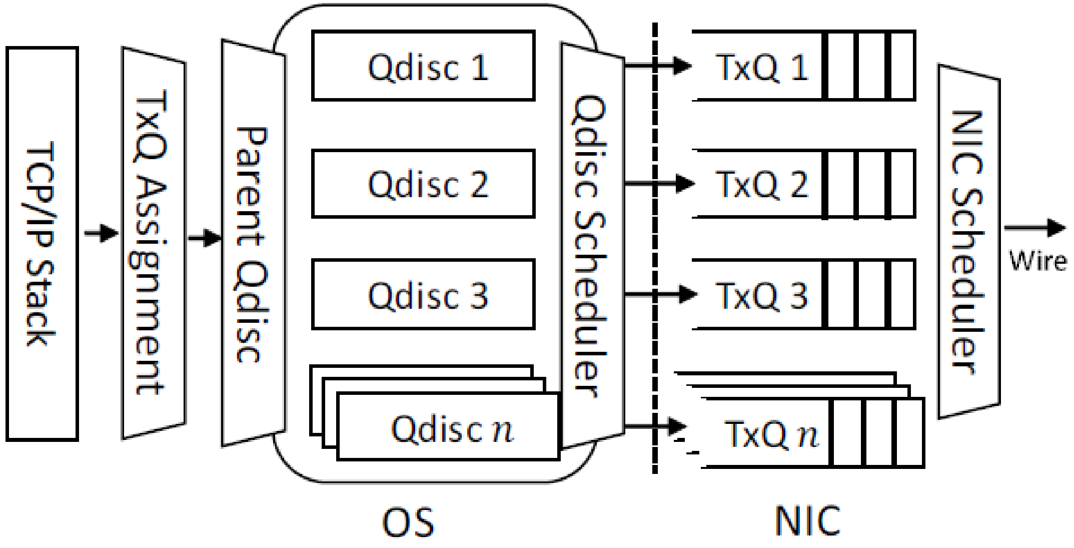
- 10Gbps NICs are commonly deployed in today's data centers
- NICs of over 40Gbps are already on the market

Intel X520-DA2 10G NIC



Motivation – Multi-Queue (MQ) NICs

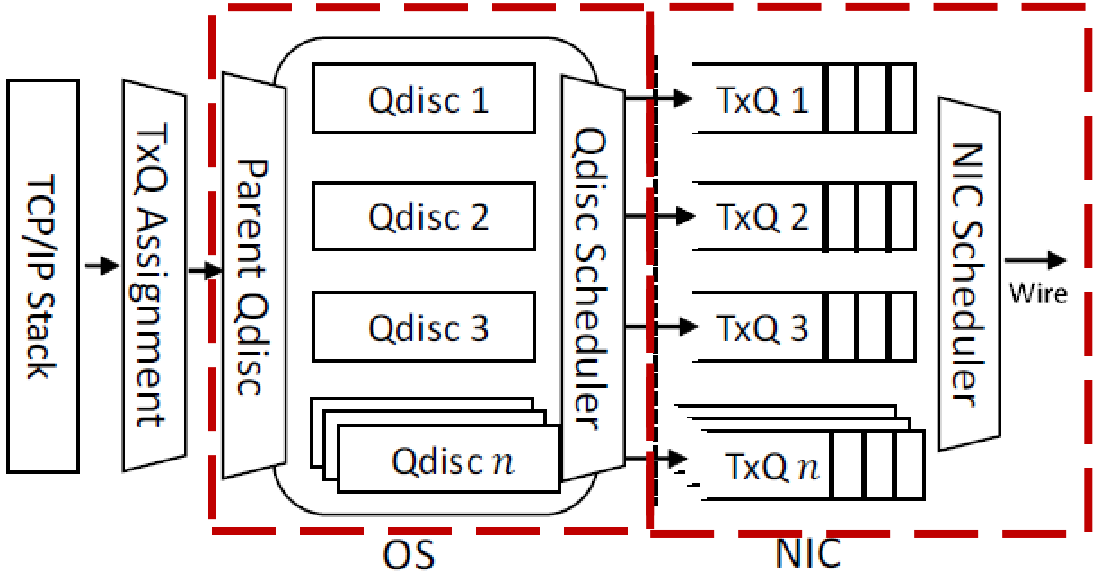
- 10Gbps and beyond NICs support multiple hardware queues
  - Enables parallelized packet transmission for multi-core systems
    - Higher CPU efficiency than single-queue NICs
    - Scales across tens of CPU cores
    - e.g. Intel X520-DA2 supports 64 queues
  - Essential to achieve line-rate for 10Gbps and beyond NICs



NetLab
Network and Security Research Lab.

Motivation – Packet Scheduling with MQ-NICs

- NIC becomes the ultimate policy enforcement point
  - With single-queue NICs, the qdisc is responsible for policy enforcement



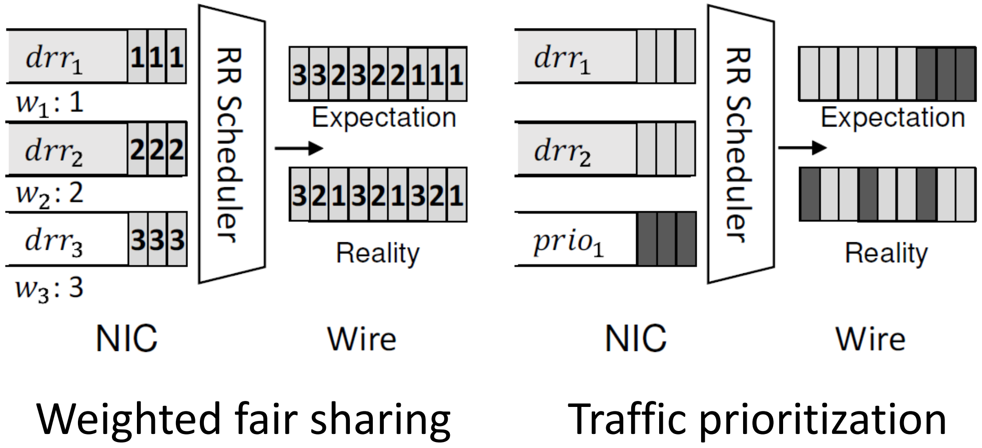
Traditional policy enforcement point

Policy enforcement point with MQ-NICs

NetLab Network and Security Research Lab.

Motivation – Policy Violation with MQ-NICs

- MQ-NICs support only a round-robin (RR) scheduler
  - Cannot enforce network policy at all due to hardware constraint
  - Rich packet scheduling in the qdisc cannot be preserved in the NIC



	Line-rate throughput	Policy enforcement
Single-Queue NIC	X	◯
Multi-Queue NIC	◯	X

NetLab Network and Security Research Lab.

Motivation – How to solve the dilemma?

- Straightforward solution: designing new NIC hardware!
  - Loom [NSDI'19]: ASIC with programmable packet schedulers
    - Only supports 2048 flows at line-rate
  - FlexNIC [ASPLOS'16], PANIC [HotNets'18]: programmable NIC hardware
  - ASIC with rich built-in packet schedulers
- Fundamental limitations of hardware solutions
  - **Burdensome replacement costs:** tens of thousands of NICs in the data center
  - **A lot of time to deploy in practice:** several years for commercialization and manufacturing

[Loom] B. Stephens, A. Akella, and M. Swift, "Loom: Flexible and efficient NIC packet scheduling," in *Proc. of USENIX NSDI*, 2019.

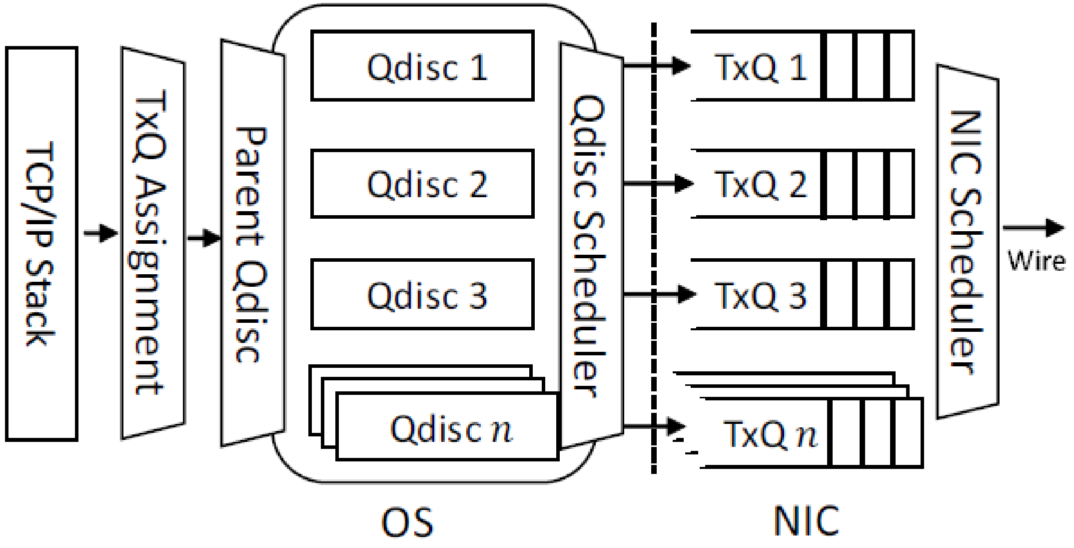
Motivation - Problem and Requirements

- **Q:** how to enforce network policy in end-hosts with commodity MQ-NICs?

- **Inter-tenant isolation:** should be able to share bandwidth fairly among tenants with different weights
- **Intra-tenant isolation:** should be able to prioritize latency-sensitive traffic within a tenant
- **Commodity NIC support:** should be work with commodity MQ-NICs

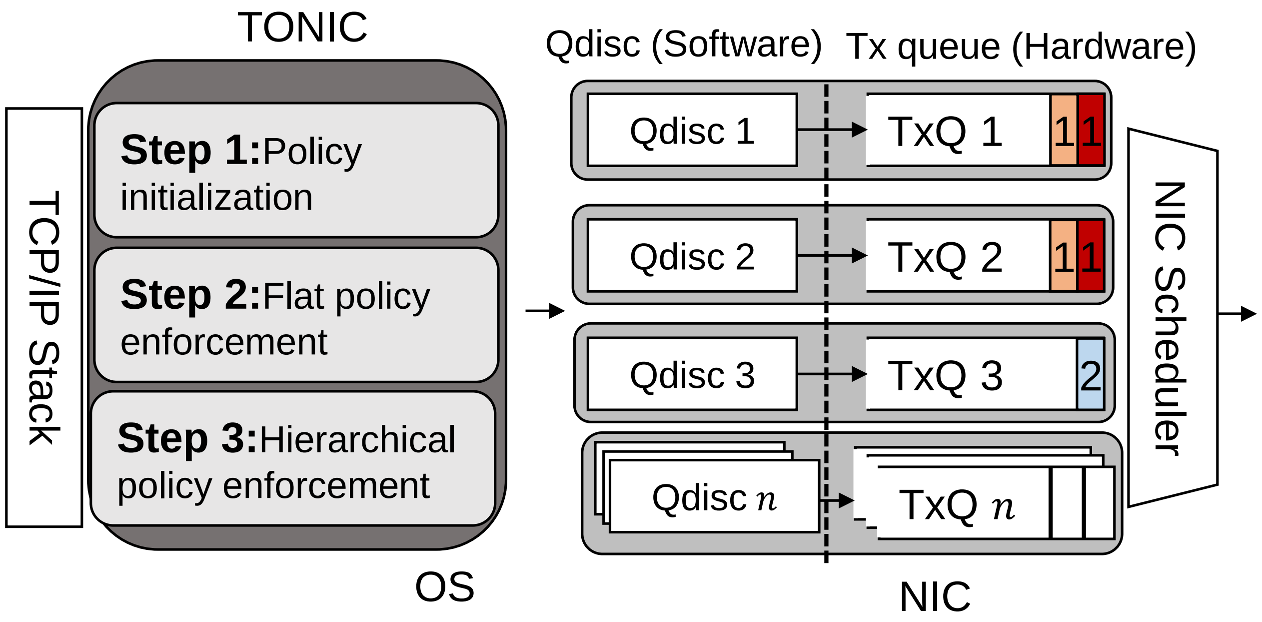
Design – Key Idea of TONIC

- Key insight: NIC packet enqueueing decisions happen in the OS, not in the NIC
  - May be able to approximate various packet schedulers indirectly
- TONIC dynamically enqueues packets in software to <u>manipulate the packet dequeueing sequence of the hardware scheduler</u>



Design – Overview

- **Step1 :** internal array initialization and CPU/IRQ affinity configuration
- **Step2:** ensures inter-tenant isolation by leveraging multiple Tx queues
- **Step3:** ensures intra-tenant isolation by head buffering



Design – Policy Initialization

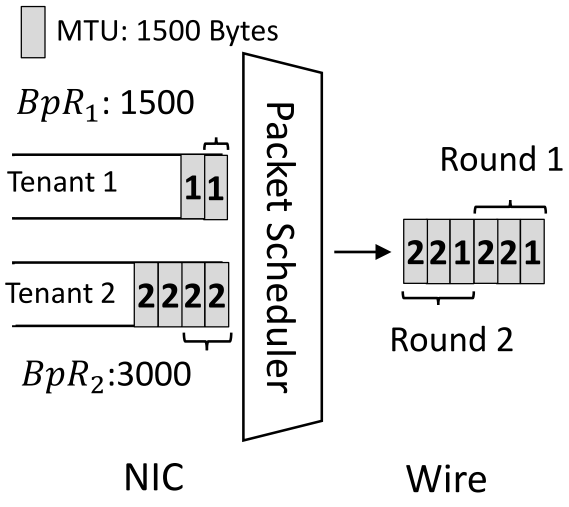
- With SLA, operators know network policy information including:
  - # of CPU cores
  - Tenant weights
  - Port numbers of high priority applications (if specified)

- TONIC maintains the following 4 arrays
  - Tenant mapping array: tenant-core mapping using `sender_cpu` metadata
  - Tenant weight array: the index range of tenant queues in ascending order
  - Tenant start index array: the first queue index for each tenant
  - Port number array: port numbers of applications should be prioritized

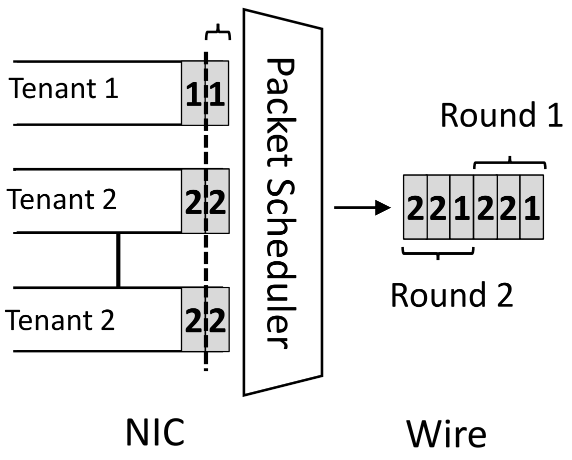
Design – Flat Policy Enforcement

- Leverages the mismatch between # of queues and tenants in an end-host
  - MQ-NICs support many queues (e.g. 64 queues in Intel 82599 NICs)
  - Applications require tens of containers with many CPU cores
  - Each tenant runs several applications in end-hosts
- Tenant weights are expressed by the number of queues
  - Updates `queue_mapping` metadata

BpR: Bytes per Round



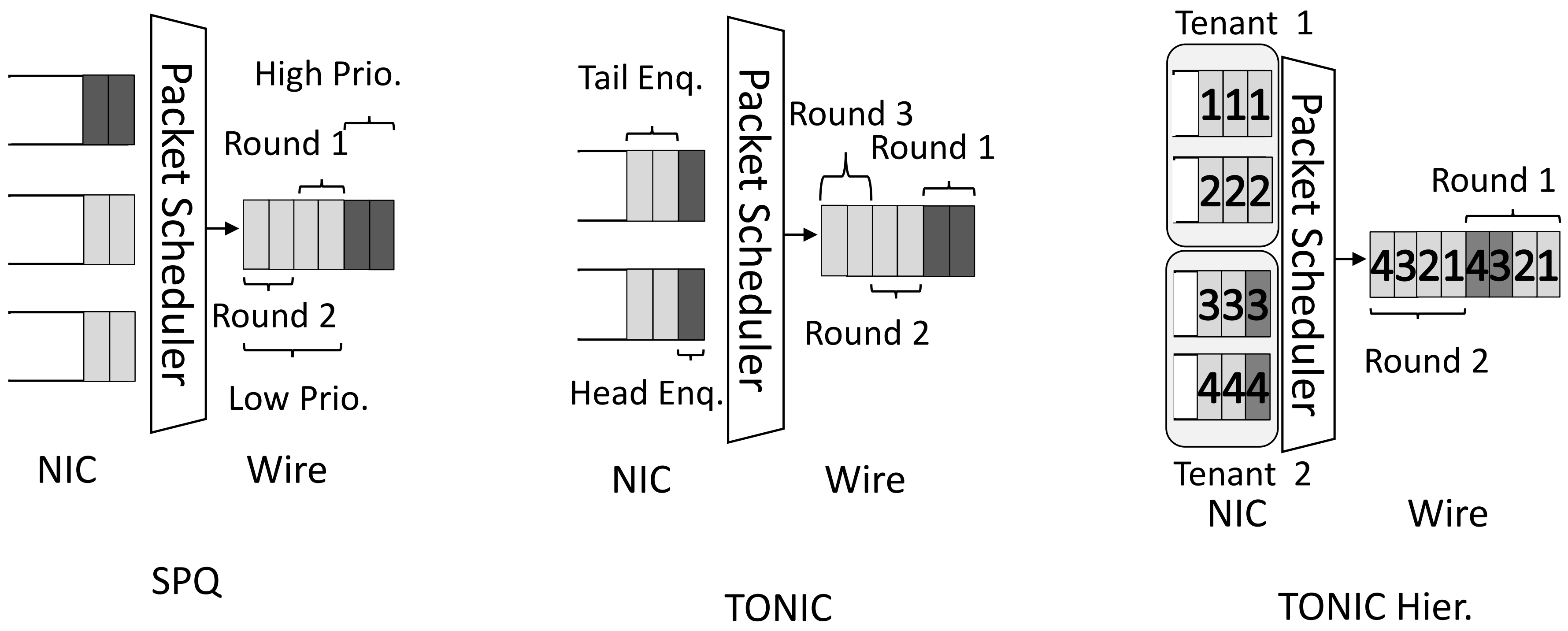
WRR



TONIC

Design – Hierarchical Policy Enforcement

- Leverages a double-ended queue structure of the qdisc
  - TONIC buffers the packet to the head of qdisc if the application has high priority
  - TONIC sets `priority` metadata to 1, and the parent qdisc performs head buffering



Implementation

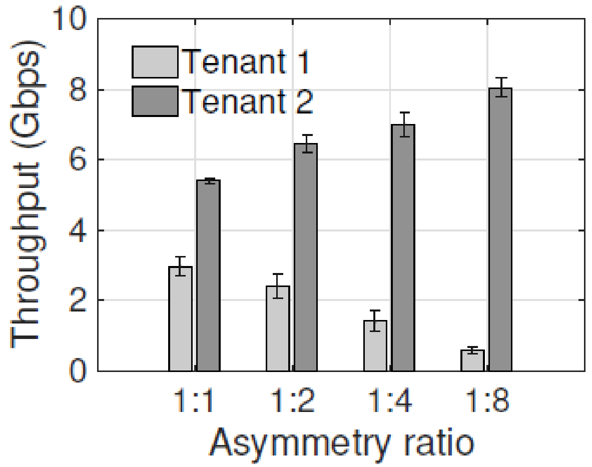
- Implemented as NETFILTER module in Linux kernel 4.4
  - Shim layer between the network stack and the qdisc layer
  - Updates `queue_mapping` and `priority` metadata
  - Disabled XPS since it overwrites `queue_mapping`
  - Modified `multiq` qdisc module to perform head buffering
- Each tenant is isolated by Linux cgroups

Evaluation

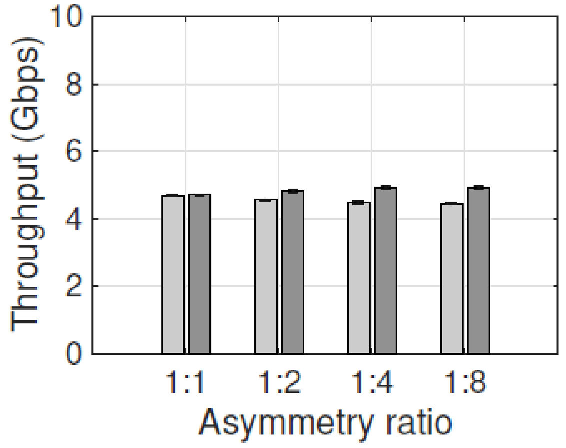
- Testbed setup
  - Two servers connected to a 10Gbps switch
  - Servers are with Intel X520-DA2 82599 10Gbps NIC supporting 64 hardware queues
  - Enabled TSO and LRO to reduce CPU overhead
- Compared scheme
  - **XPS:** the state-of-the-art MQ-NIC solution in the current Linux kernel
    - Matches each CPU core with each Tx queue to support parallel packet processing
    - Packets are buffered into the matched Tx queue of the CPU core generated the packet

Evaluation – Equal Fair Sharing

- Two tenants with the equal weights
- Tenant 1 has 8 flows while tenant 2 has {8,16,32,64} flows



XPS



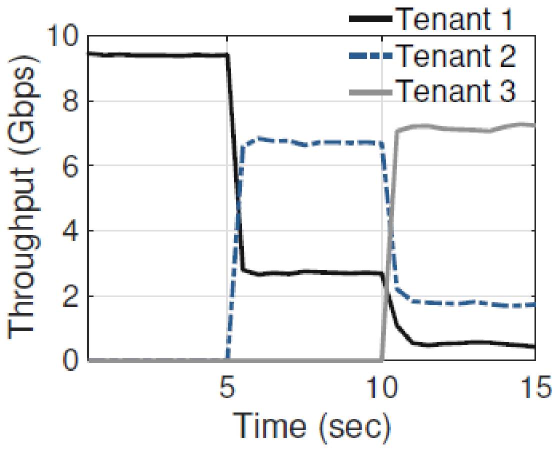
TONIC

TONIC ensures almost equal sharing regardless of the number of flows per tenant

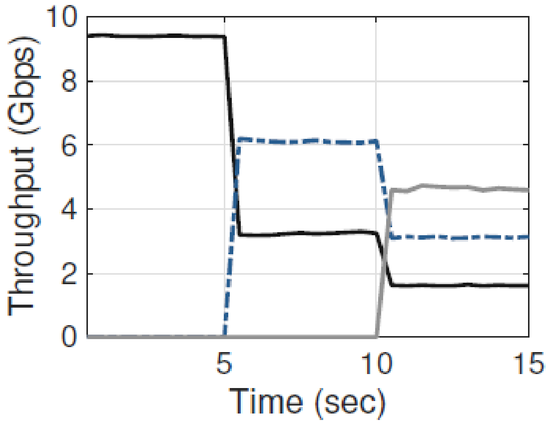
NetLab Network and Security Research Lab.

Evaluation – Weighted Fair Sharing

- Three tenants with weights of 1:2:3
- Each tenant has {8,16,32} flows, respectively



XPS

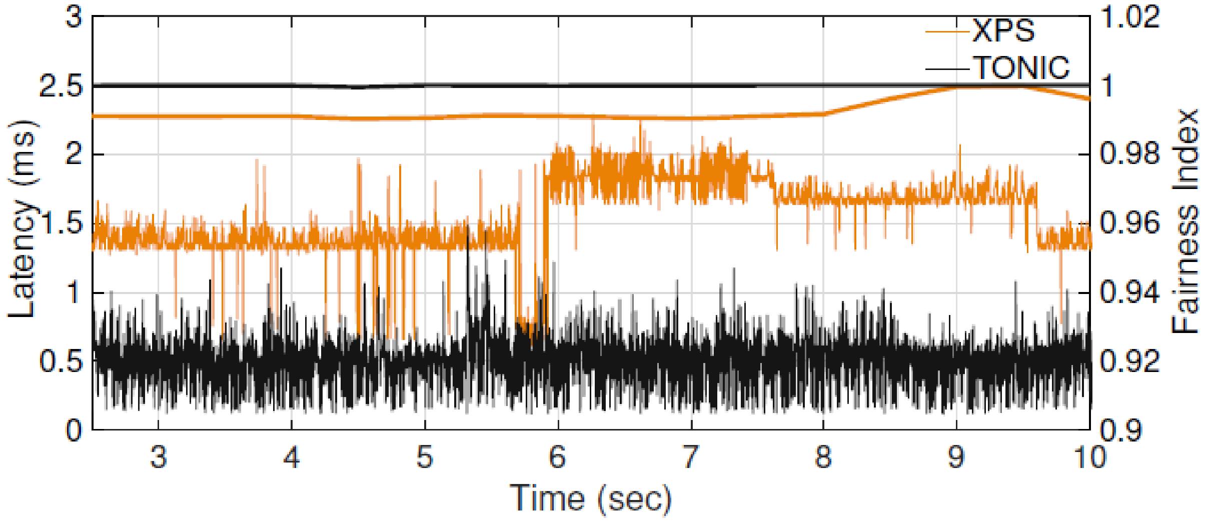


TONIC

TONIC respects the assigned weights regardless of different number of flows

Evaluation – Traffic Prioritization

- Two tenants with the equal weight
- Tenant 1 runs `iperf` only while tenant 2 runs `iperf` and `sockperf`
  - `sockperf` has higher priority than `iperf` within tenant 2



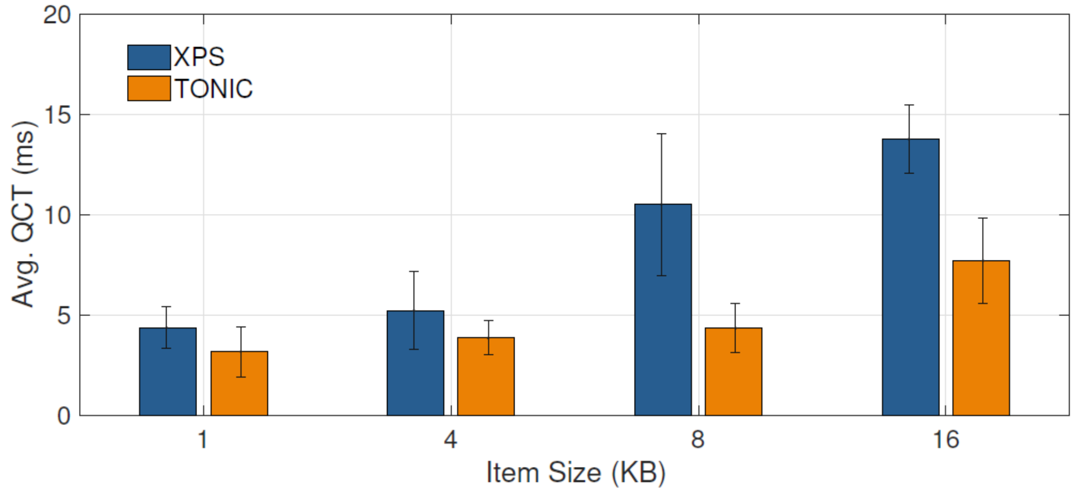
Packet latency of tenant 2 and throughput fairness

TONIC preserves inter-tenant isolation and achieves intra-tenant isolation as well

NetLab Network and Security Research Lab.

Evaluation – Results with KVS

- The same settings with the previous experiment except that tenant 2 uses memcached KVS instead of `sockperf`



The average QCT of memcached with different item sizes

TONIC can improve the performance of real DC applications by enforcing network policy with commodity MQ-NICs

Network and Security Research Lab.

Summary of TONIC

- **Problem:** how to enforce network policy in end-hosts with commodity MQ-NICs?
- **Key idea:** approximates various packet scheduling algorithms by manipulating the packet dequeueing sequence of hardware schedulers through dynamic packet enqueueing decisions in software
- **TONIC**: an end-host packet scheduling solution that enables network policy enforcement with commodity MQ-NICs
  - Expresses tenant weights as the number of assigned Tx queues
  - Prioritizes high priority traffic through head buffering in the qdisc
- **Results**
  - Preserves equal sharing and weighted fair sharing regardless of # of flows
  - Ensures traffic prioritization within a tenant while maintaining inter-tenant fairness
  - Memcached experiments suggest that TONIC can enhance the performance of real DC applications

NetLab
Network and Security Research Lab.