Automated and Scalable QoS Control
- For Network Convergence

Wonho Kim (Princeton Univ.) Puneet Sharma, Jeongkeun Lee, Sujata Banerjee, Jean Tourrilhes, Sung-Ju Lee, and Praveen Yalagandula (HP Labs)
Motivation

• Why do we care about QoS control?
  – Network convergence
  – Multi-tenancy networks

• **Automated** QoS control is needed
Network convergence

Different protocols, adapters, switches, and configuration
Network convergence

- Fewer switches, ports, adapters, cables
- Reduced power, equipment, cooling cost
- Simpler topology
- I/O consolidation
- Unified resource management

- Converged Enhanced Ethernet (CEE)
- Data Center Ethernet (DCE)
- Data Center Bridging (DCB)
- Fibre Channel over Ethernet (FCoE)
- Fibre Channel over CEE (FCoCEE)
Multi-tenancy networks

- Serve multiple customers with a single fabric
- Better utilization of network infrastructure
Performance isolation
Performance isolation
Performance isolation
Performance isolation

- Need virtual network slices
- Need fine-grained performance isolation

Virtualized Servers
Variable Workloads
Bugs, malicious attack
Goal

- Enables performance isolation with QoS control
Good news

• Most commodity switches have QoS knobs
  – rate limiter
  – priority queues
  – schedulers

• Single network domain
  – datacenters, enterprise networks, …
  – free from Layer-8 issues (billing, collaborations, …)
  – fine-grained control becomes feasible
Challenges

- Coarse-grained QoS knobs
  - designed for distributed management
  - class-based
  - no e2e performance

- Manual configuration
  - no standards for classifiers
  - error-prone
  - static (not adaptive)
Our Solution: OpenFlow QoS Controller

Virtual Slice 1, Virtual Slice n

Physical network fabric

QoS configuration (rate limiter, priority queues, …)

Manual & Static setting

Problem

QoS Controller

• Automated configuration
• Fine-grained flow management
• Adaptive to dynamic workloads
Overview of OpenFlow QoS controller

Flow
Src
rate limit
Dest

OpenFlow APIs

Adaptive Aggregator → QoS Controller → Network-wide Optimization

Topology, Nodes, Resource, Flows, ...
End-to-End performance models

Rate limiters
Queue mappings
Shortest Span First ...

Flow
Src
prio
prio
prio
prio
Dest
Adaptive aggregation

Flow specs

- flow 1: Customer DB
  - src IP: X.X.X.X, port: 9551
  - Type: IP (UDP), ...
  - Peak rate: 400 Mbps
  - Delay bound: 10 ms
  - Aggregate: False

- flow 2: Employee DB

- flow 3: Video

- flow 4: System backup

- flow 5: Backup
  - Aggregate: True

- flow 6: Log Archive

Slice specs

- Storage
  - Peak rate: 400 Mbps
  - Delay bound: 10 ms
  - Aggregate: False

- Video
  - Peak rate: 100 Mbps
  - Delay bound: 100 ms
  - Aggregate: False
Available QoS Knobs (Priority queue)
Available QoS Knobs (Rate limiter)
OpenFlow QoS APIs

Rate limiter

Priority queue mapping

- Extension of OpenFlow specification
- Expose QoS capability in switches
OpenFlow QoS APIs

• With OpenFlow flow control
  – fine-grained control of flows
  – automated flow management

• With OpenFlow QoS APIs
  – uniform control of QoS knobs
  – configure QoS for individual (or aggregate) flows
Admission Control

• **Input**
  - new flow arrival event
  - performance requirements (peak rate, e2e delay)
  - database for the current network state
  - end-to-end performance model

• **Output**
  - admission control result (accept/reject)
  - priority queue assignment, rate limiter settings
  - path selection
Admission Control

• Two conditions should be satisfied
  – satisfy $f$’s performance requirement
  – not violate existing flows in the networks
Difficulties in queue assignment

We should consider interactions between
- flows in a switch
- flows in multiple switches
Admission control heuristic

• Goal
  – increase the ratio of admitted flows
  – lower the complexities in queue allocation

• Shortest Span First (SSF)

• Basic ideas
  – estimate affordable options for a flow
  – try first switches more likely to reject flow
Highest level & Lowest level

- **Highest level**: not violate existing flows
- **Lowest level**: not violate the new flow
- **Span**: available options for $f$
Shortest Span First (SSF)

- **Step 1**: compute highest & lowest levels independently
Shortest Span First (SSF)

- Step 2: sort switches in order of the span
Shortest Span First (SSF)

- **Step 2**: sort switches in order of the span
Shortest Span First (SSF)

- Step 3: try highest level at each hop
  - try first a switch more likely to reject flow
Implementation

- QoS APIs implemented on
  - hardware switch (HP ProCurve 5406zl)
  - software switch (Open vSwitch)

- QoS Controller implemented on top of NOX
  - open-source OpenFlow controller
  - [http://noxrepo.org](http://noxrepo.org)

- QoS Controller web interface
Prototype

Host A → QoS Controller
Output 14 Priority 7

Host B → Switch A → QoS Controller
Switch C → QoS Controller

Switch A → Output 21 Priority 7
Switch B → Output 21 Priority 7

Host C → Switch B
Host D → Switch C

Peak rate 400 Mbps
Delay bound 10 ms
Evaluation

- Traffic generation
  - generate 3 guaranteed flows from emulated services (UDP)
  - generate cross traffic (UDP, TCP)

- Disable/Enable QoS controller

- Measured throughput and packet loss in testbeds
Throughput with UDP cross traffic

Generate cross traffic

Flow name | Route (queue assignment)
--- | ---
Customer DB | H3 – S3(8) – S1(8) – H1
Employee DB | H4 – S3(8) – S1(8) – H2
VoD | H3 – S3(7) – S1(7) – H1
System Backup | H4 – S3(1) – S1(1) – H2

QoS controller protects guaranteed flows in congestion
Packet loss with TCP cross traffic

QoS control is needed even when most traffic in network is TCP
Future works

• Evaluations
  – effectiveness of admission control heuristics (ratio of admitted flows)
  – compare with offline optimal assignment
  – simulations on a variety of datacenter networks (e.g., Hierarchical, FatTree, …)

• Deployment
  – extend deployment to large networks
  – test with mixture of services
Conclusion

- Single integrated network fabric is desirable

- We need fine-grained automated QoS control

- Contributions
  - Design & Implement OpenFlow QoS APIs
  - QoS controller: automated QoS control for network slicing
Thank you