Cloud Control with Distributed Rate Limiting

Barath Raghavan, Kashi Vishwanath,
Sriram Ramabhadran, Kenneth Yocum, and Alex C. Snoeren

University of California, San Diego
Centralized network services

- Hosting with a single physical presence
  - However, clients are across the Internet
Running on a cloud

- Resources and clients are across the world
- Services combine these distributed resources

1 Gbps
Key challenge

We want to control distributed resources as if they were centralized
Ideal: Emulate a single limiter

- Make distributed **feel** centralized
  - Packets should experience same limiter behavior
Distributed Rate Limiting (DRL)

Achieve functionally equivalent behavior to a central limiter

1. Global Token Bucket
2. Global Random Drop
3. Flow Proportional Share

Packet-level (general)                  Flow-level (TCP specific)
Distributed Rate Limiting tradeoffs

Accuracy
(how close to $K$ Mbps is delivered, flow rate fairness)

+ 

Responsiveness
(how quickly demand shifts are accommodated)

Vs.

Communication Efficiency
(how much and often rate limiters must communicate)
DRL Architecture

Packet arrival

Estimate interval timer

Enforce limit

Estimate local demand

Global demand

Set allocation

Limiter 1

Gossip

Limiter 2

Gossip

Limiter 3

Gossip

Limiter 4
Token Buckets

Token bucket, fill rate $K$ Mbps
Building a Global Token Bucket

Limiter 1

Limiter 2

Demand info (bytes/sec)
Baseline experiment

Single token bucket

10 TCP flows

S → D

Limiter 1

3 TCP flows

S → D

Limiter 2

7 TCP flows

S → D
Global Token Bucket (GTB)
(50ms estimate interval)

Problem: GTB requires near-instantaneous arrival info
Global Random Drop (GRD)

Limiters send, collect global rate info from others

5 Mbps (limit)
4 Mbps (global arrival rate)

Case 1: Below global limit, forward packet
Global Random Drop (GRD)

6 Mbps (global arrival rate)
5 Mbps (limit)

Same at all limiters

Case 2: Above global limit, drop with probability:

\[
\frac{\text{Excess}}{\text{Global arrival rate}} = \frac{1}{6}
\]
GRD in baseline experiment
(50ms estimate interval)

Delivers flow behavior similar to a central limiter
GRD with flow join
(50ms estimate interval)

Flow 1 joins at limiter 1
Flow 2 joins at limiter
Flow 3 joins at limiter 3
Flow Proportional Share (FPS)

Limiter 1

3 TCP flows

S → Limiter 1 → D

Limiter 2

7 TCP flows

S → Limiter 2 → D
Flow Proportional Share (FPS)

Goal: Provide inter-flow fairness for TCP flows

Local token-bucket enforcement

“3 flows”

“7 flows”

Limiter 1

Limiter 2

...
Estimating TCP demand

Limiter 1

S

1 TCP flow

S

1 TCP flow

D

Limiter 2

3 TCP flows

S

D
Estimating TCP demand

**Local** token rate (limit) = 10 Mbps

Flow A = 5 Mbps

Flow B = 5 Mbps

Flow count = 2 flows
Estimating TCP demand

1 TCP flow

Limiter 1

D

3 TCP flows

Limiter 2

D
Estimating **skewed** TCP demand

**Local** token rate (limit) = 10 Mbps

Flow A = 2 Mbps

Flow B = 8 Mbps

Bottlenecked elsewhere

Flow count ≠ demand

Key insight: Use a TCP flow's rate to infer demand
Estimating *skewed* TCP demand

Local token rate (limit) = 10 Mbps

- Flow A = 2 Mbps
- Flow B = 8 Mbps

\[
\frac{\text{Local Limit}}{\text{Largest Flow’s Rate}} = \frac{10}{8} = 1.25 \text{ flows}
\]

Bottlenecked elsewhere
Flow Proportional Share (FPS)

Global limit = 10 Mbps

Limiter 1
- 1.25 flows

Limiter 2
- 2.50 flows

Set local token rate = \[
\frac{10 \text{ Mbps} \times 1.25}{1.25 + 2.50}
\]

= 3.33 Mbps
Under-utilized limiters

Set local limit equal to actual usage (limiter returns to full utilization)
Flow Proportional Share (FPS)

(500ms estimate interval)
Additional issues

• What if a limiter has no flows and one arrives?
• What about bottlenecked traffic?
• What about varied RTT flows?
• What about short-lived vs. long-lived flows?

• Experimental evaluation in the paper
  – Evaluated on a testbed and over Planetlab
Cloud control on Planetlab

- Apache Web servers on 10 Planetlab nodes
- 5 Mbps aggregate limit
- Shift load over time from 10 nodes to 4 nodes
Static rate limiting

Demands at 10 apache servers on Planetlab

Wasted capacity
Demand shifts to just 4 nodes

Rate (Kbps)

Time (sec)
FPS (top) vs. Static limiting (bottom)
Conclusions

• Protocol agnostic limiting (extra cost)
  – Requires shorter estimate intervals
• Fine-grained packet arrival info not required
  – For TCP, flow-level granularity is sufficient
• Many avenues left to explore
  – Inter-service limits, other resources (e.g. CPU)
Questions!