On Application-level Approaches to Avoiding TCP Throughput Collapse in Cluster-based Storage Systems

Elie Krevat
Vijay Vasudevan, Amar Phanishayee,
David Andersen, Greg Ganger,
Garth Gibson, Srini Seshan

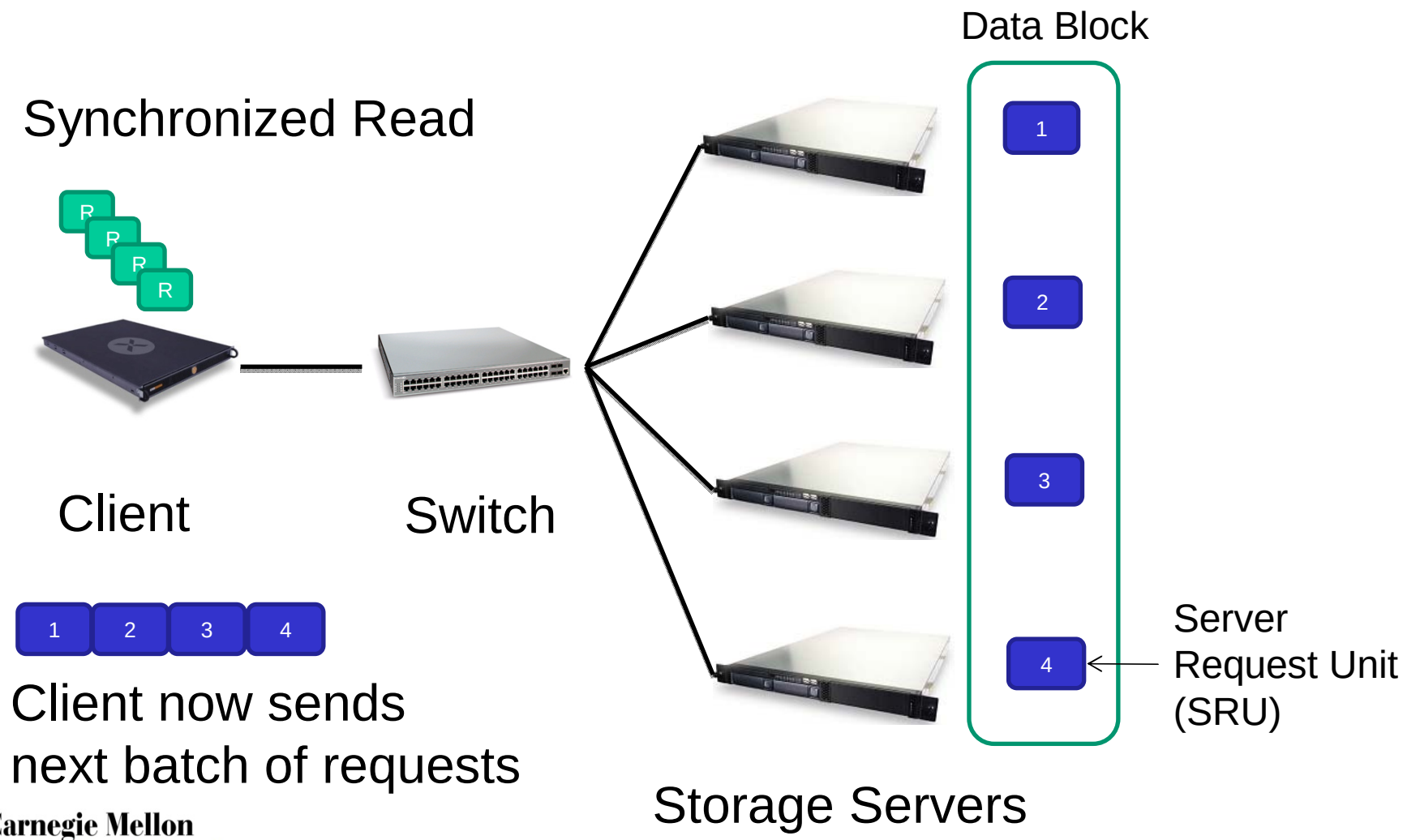
PARALLEL DATA LABORATORY
Carnegie Mellon University

Carnegie Mellon
Parallel Data Laboratory

Outline

- Motivation: TCP throughput collapse
- TCP- and Ethernet-level improvements
- Possible application-level solutions
- Conclusions

Cluster-based Storage Systems



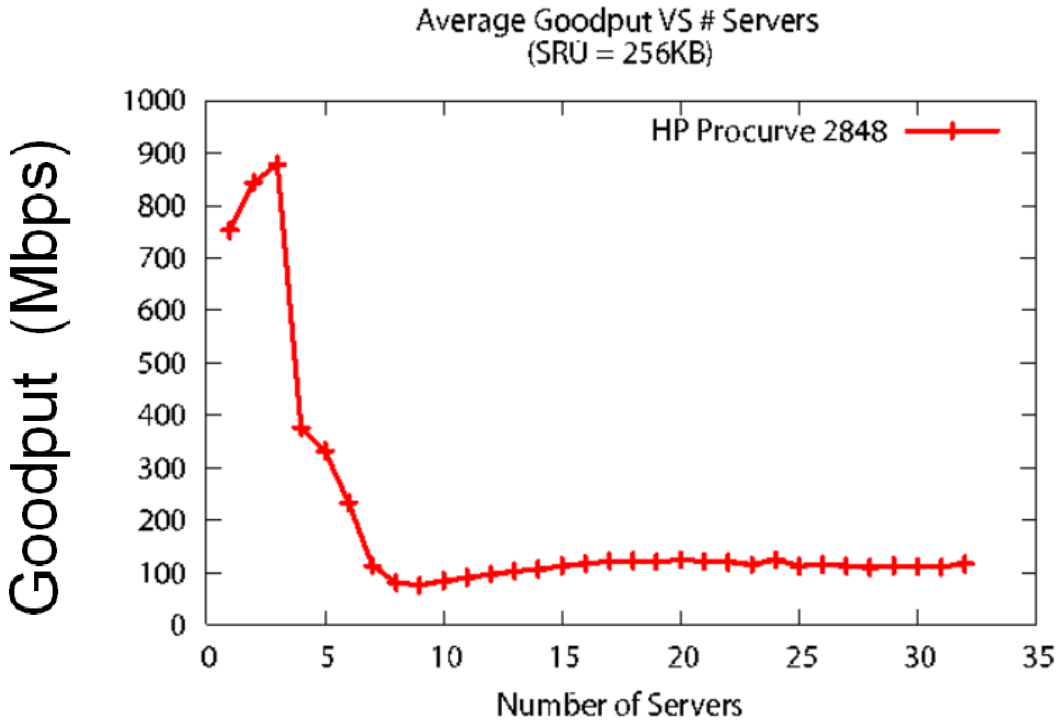
Storage Networking Options

- FibreChannel, InfiniBand
  - ✓ Specialized high throughput networks
  - ✘ Expensive
- Commodity Ethernet networks
  - ✓ Low cost
  - ✓ Shared infrastructure
  - ✘ TCP throughput collapse (with synchronized reads)

Storage Cluster Experimental Setup

- Client performs synchronized reads
  - Fix SRU size
  - Increase # of servers
- Servers respond with cached data
- Measure goodput (i.e., app throughput)

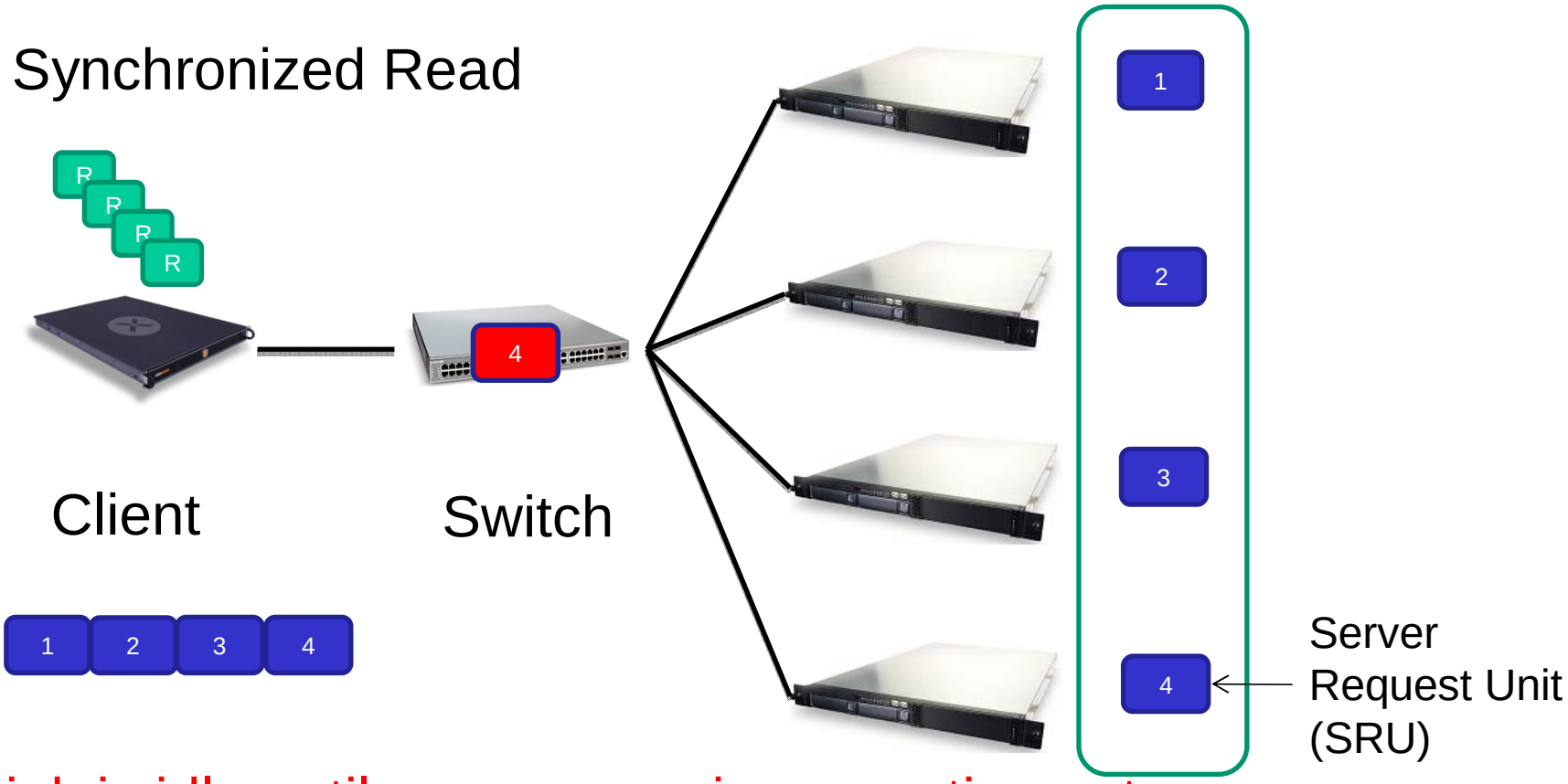
TCP Throughput Collapse: *Incast*



Number of servers

- [Nagle04] called this *Incast*
- Cause of throughput collapse: TCP timeouts

Link idle time due to timeouts



Link is idle until server experiences a timeout

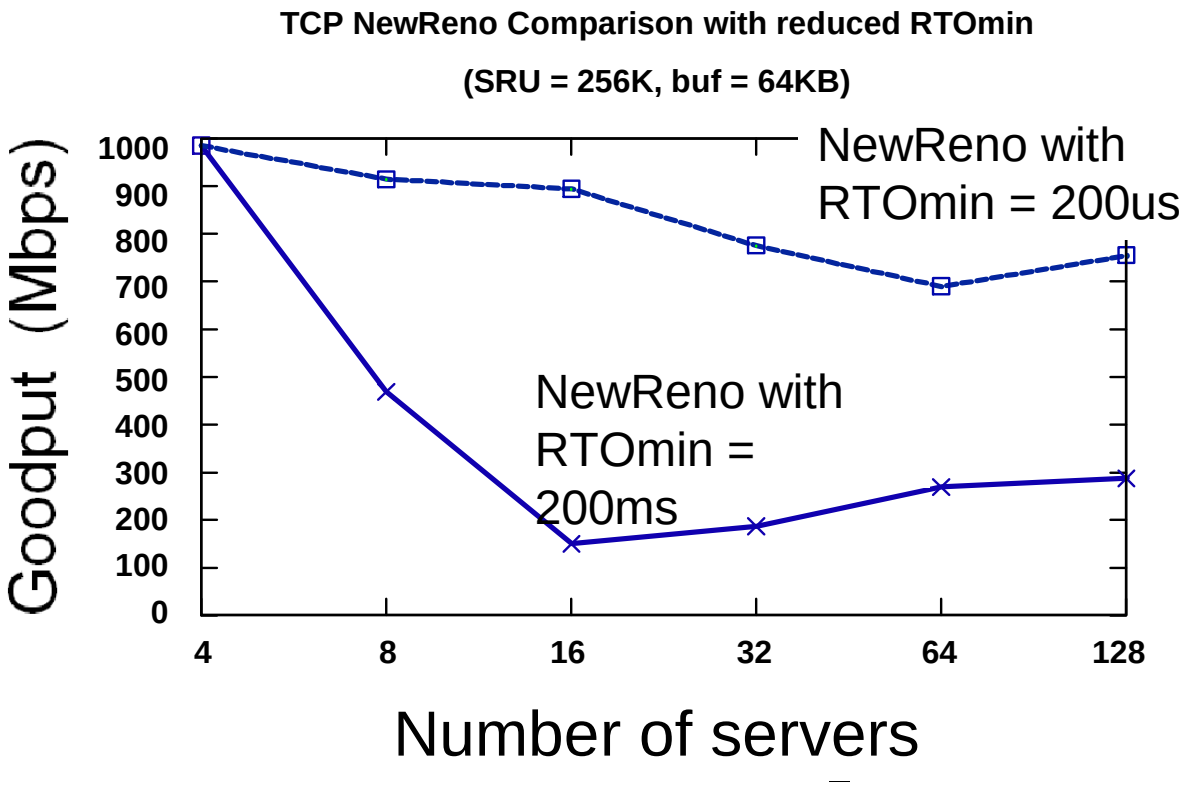
Outline

- Motivation: TCP throughput collapse
- ➢ TCP- and Ethernet-level improvements
- Possible application-level solutions
- Conclusions

Reducing the penalty of timeouts

- Reduce penalty by reducing TCP ***R***etransmission ***T***ime***O***ut period (RTO)
- RTOmin to guard against premature timeouts
  - Default = 200ms
  - Orders of magnitude greater than RTT (100 us)

Reducing RTOmin in simulation



Issues with Reducing RTOmin

- Still show 30% decrease for 64 servers
- Implementation Hurdles:
  - Soft Timers (μs granularity)
  - Safety and generality
    - Servers talk to other clients over wide area
    - Premature timeouts

More TCP Improvements

- Increasing SRU size means less link idle time
  - ✓ MB-sized SRUs effective
  - ✘ More pinned space in kernel memory
- Larger switch buffers can mitigate *Incast*
  - ✓ Doubling buffer space supports double the servers
  - ✘ Expensive switches
- More results at FAST ’08

Ethernet Improvements?

- Ethernet Flow Control helps, with problems
  - ✓ Very good performance on one switch
  - ✘ Adverse effects on other flows
- New Ethernet protocols/standards
  - Congestion management
    - Rate-limiting behavior
  - Granular per-channel flow control
    - "Pause" packets won't block entire link
  - ✓ Provide "no-drop" congestion response
  - ✘ May take years before added to switches

Outline

- Motivation: TCP throughput collapse
- TCP- and Ethernet-level improvements
- ➢ Possible application-level solutions
- Conclusions

Application-level Solutions

- "Application" is storage cluster software
  - Has more knowledge of all requests
- Used in practice to avoid Incast
- Can combine with TCP improvements

Increasing Request Sizes

- Make larger requests
  - From simulation, larger SRU sizes are better
  - But can't be too large!
    - Memory pressure problems
    - Latency and fairness issues

Limiting Number of Servers

- Restrict number of synchronously communicating servers
  - Try to stay in “sweet spot”
- Panasas uses “RAID groups”
  - Group size of specific range
  - Load balance over many groups

Throttling Data Transfers

- Client can throttle servers' send rates
  - Advertise smaller TCP receive buffer
- Problems with static throttle rate
  - May underutilize link across few servers
  - Doesn't generalize to multiple requests

Staggering Data Transfers

- Stagger server responses to limit interference
- Option 1: Controlled by client
  - Client requests from subset of servers
  - Maintain window of requests
- Option 2: Controlled at servers
  - Servers skew responses
  - Random or deterministic
  - Prefetch data during delay
- Natural staggering effect in real systems

Global Scheduling

- Servers respond based on all traffic to client
- Maintain global pool of *SRU tokens*
- Servers must obtain SRU token to send data
- Limited #/tokens based on “sweet spot”
- Many interesting ways to distribute tokens
  - Can use separate *token authority*
    - Resides at client or distributed
    - Load balance requests

Outline

- Motivation: TCP throughput collapse
- TCP- and Ethernet-level improvements
- Possible application-level solutions
- ➢ Conclusions

Conclusions

- Synchronized Reads + TCP timeouts → Throughput Collapse
- TCP- and Ethernet-level improvements
  - Not a complete solution
- Potential for application-level solutions
  - Storage system has more knowledge and control
  - Can avoid overloading network