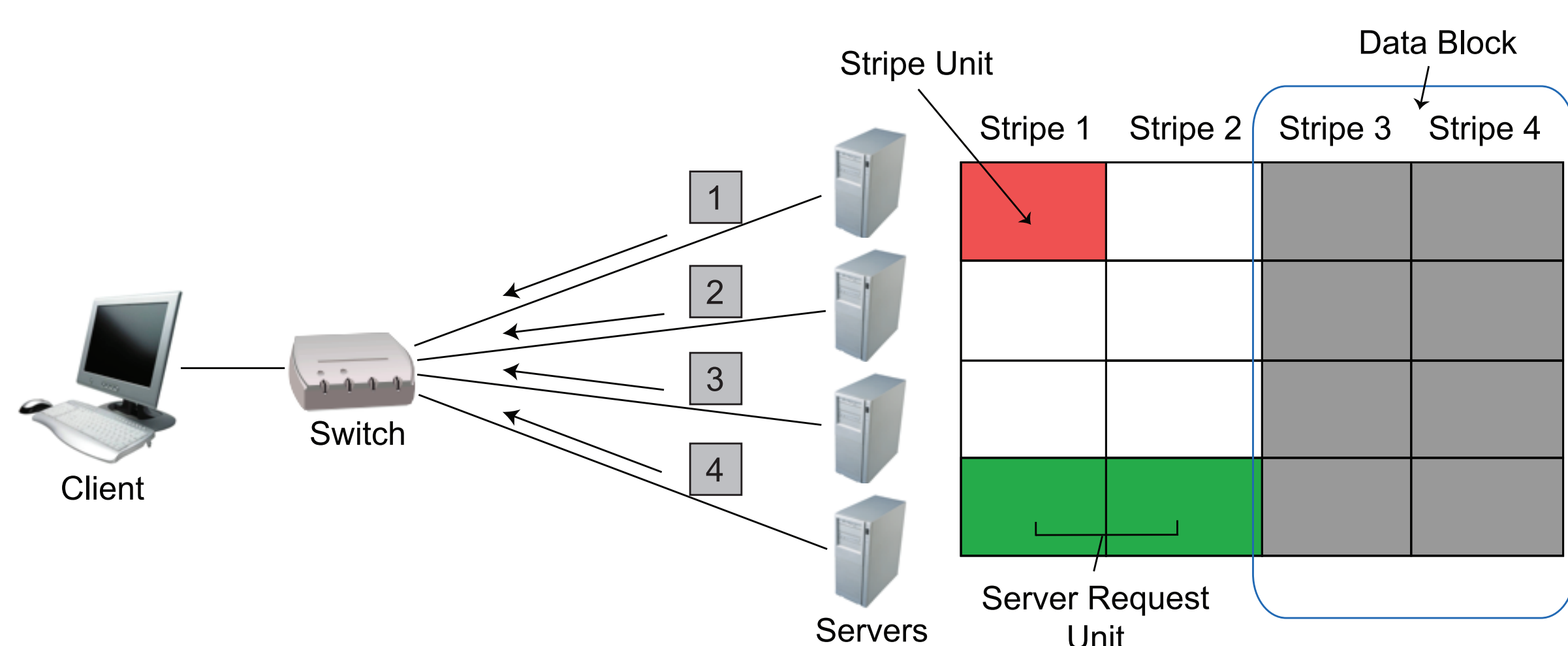


TCP Throughput Collapse in Cluster-based Storage

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

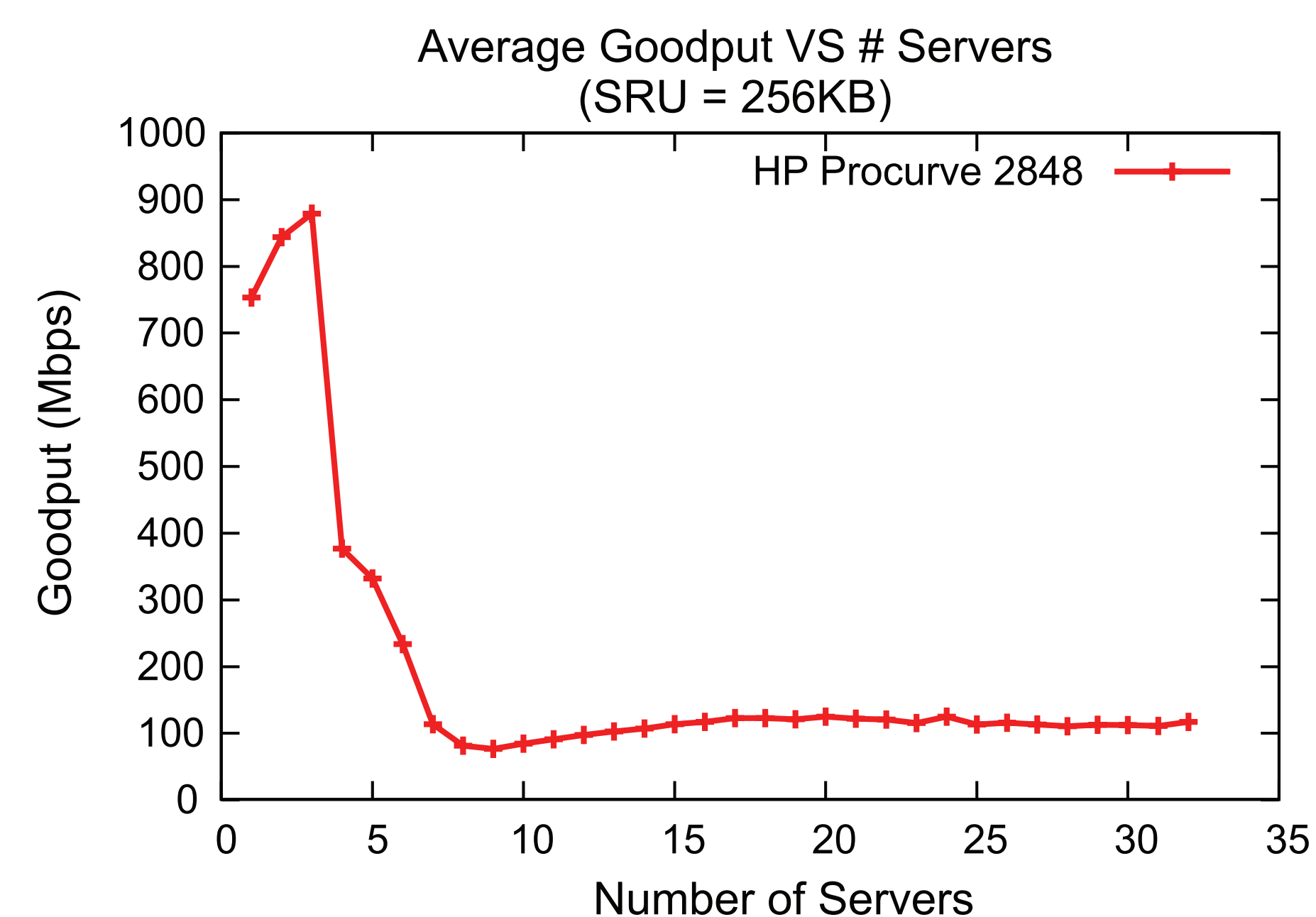
Storage Network Environment

- Cluster-based storage using TCP/IP over ethernet
- Data striped across many servers
- Client & servers separated by one or more switches
- Client requests a “block” of data and waits
 - A block is composed of one or more stripes
 - Each storage device serves its own stripe units
 - When all block data is received, client begins next request



Incast Problem

- Client can experience very poor TCP throughput
 - Data from too many servers overflows switch buffers
 - Buffer overflow causes significant packet loss
 - Goodput as low as 1-10% of client link capacity!

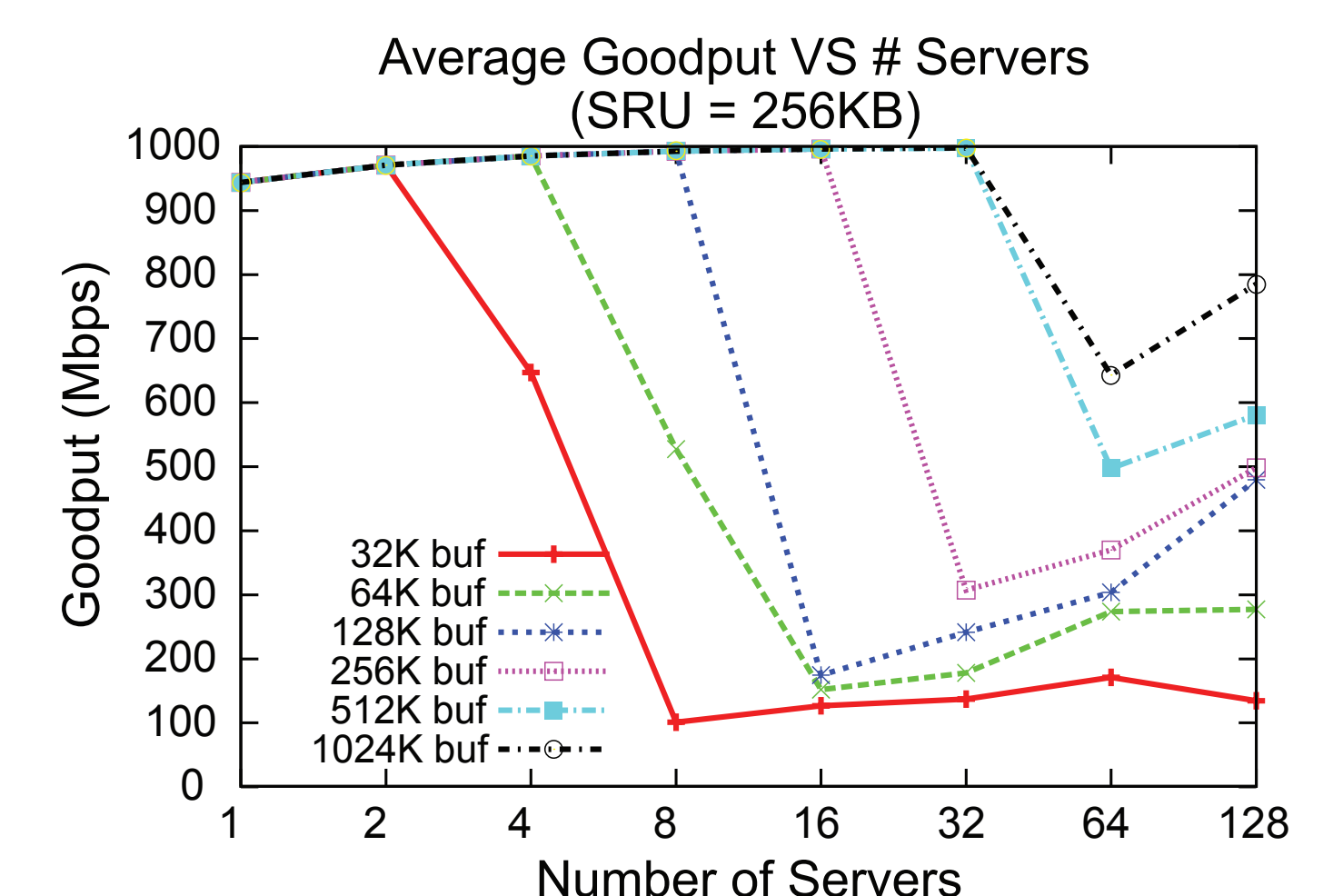


Goal: Understanding Incast

- Conditions affecting problem are not well understood
- Approach 1: Reproduce problem in real Ethernet setups
 - Exists on different switches and topologies
 - Experiment with Ethernet flow control
- Approach 2: Study network and TCP "fixes" in simulation
 - Vary TCP parameters and protocols
 - Vary switch buffer sizes, SRU sizes, and number of servers

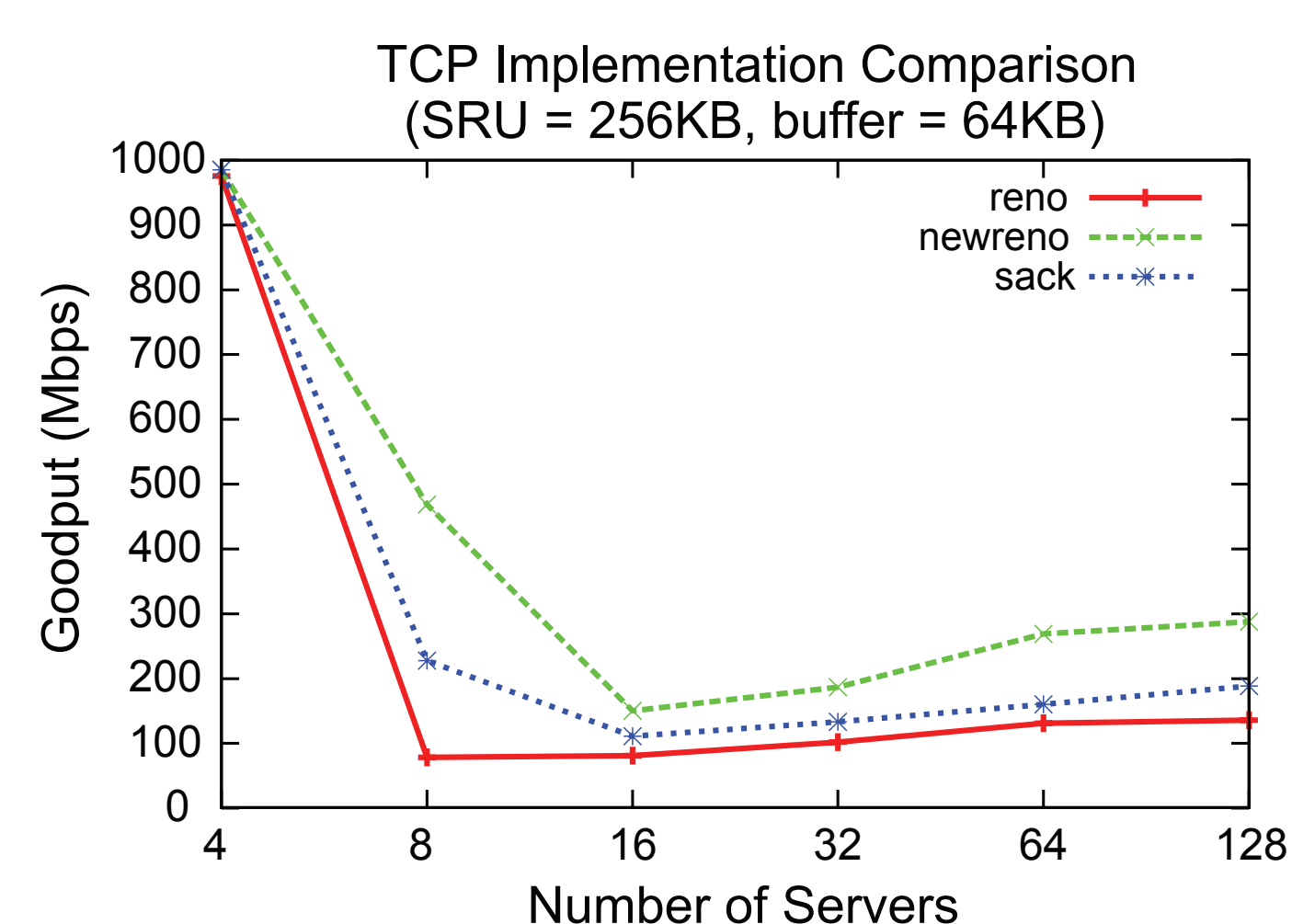
Varying Switch Buffer Size

- Larger switch buffers delay onset of incast
- Switches with large buffers are expensive



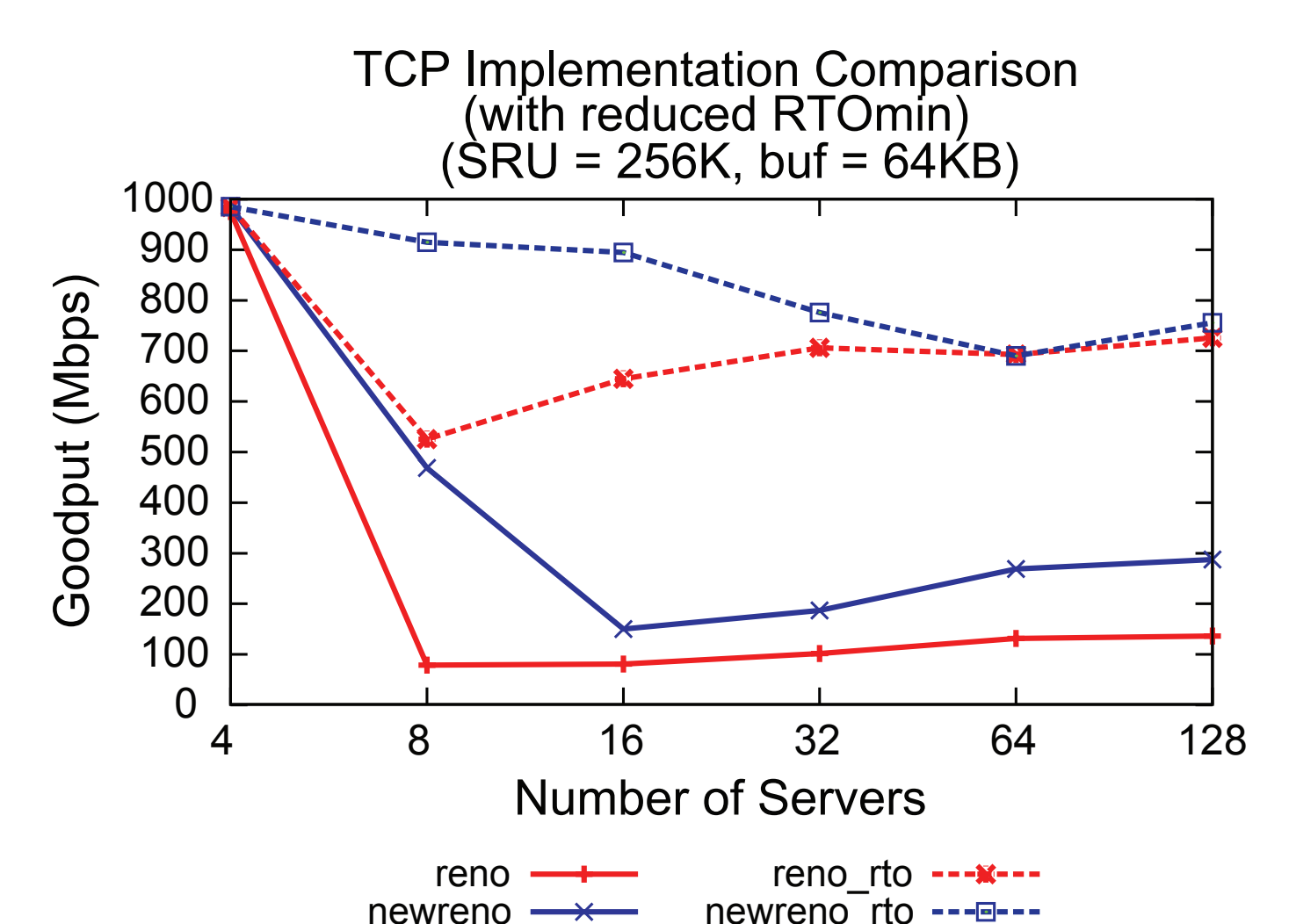
Alternative TCP Implementations

- NewReno: fewer timeouts, better goodput than Reno
- Still cannot avoid eventual goodput collapse



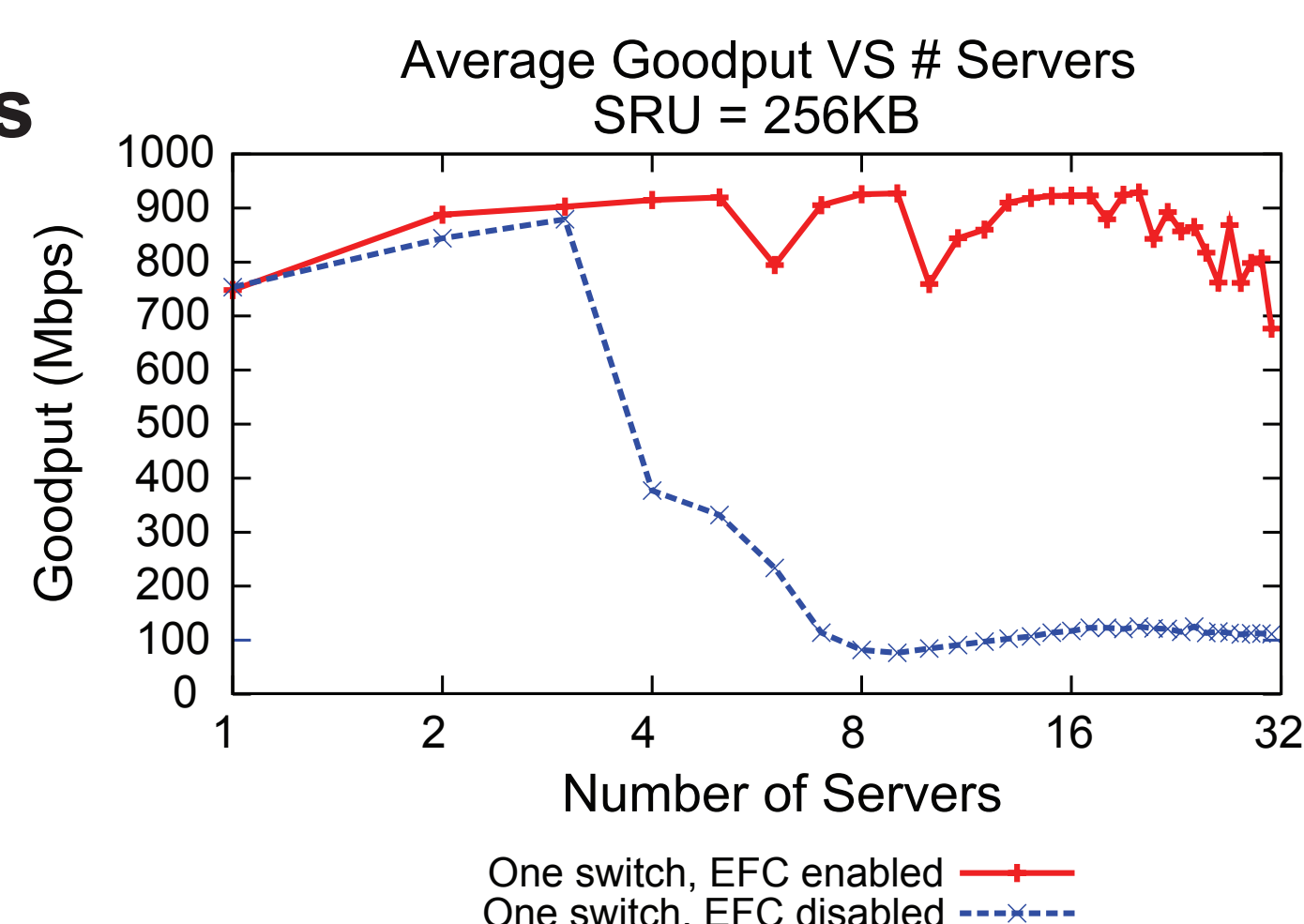
Reducing Timeout Penalty

- Reduce minimum retransmission timeout
200ms → 200μs
- Improves goodput
- Downside: safety/generalality



Ethernet Flow Control

- Effective for single switch, not multi-switched networks
- Downsides:
 - TCP flow agnostic
 - Inconsistent vendor implementations



Conclusions and Ongoing Work

- TCP timeouts cause throughput collapse during synchronized parallel data transfers
- No clear TCP- or network-level solution
 - Some only delay onset
 - Some have significant downsides
- Next step: explore application-level approaches
 - Limit number of servers or throttle transfers
 - Stagger client requests or skew responses
 - Globally schedule data transfers

