

# Trickles:

## A Stateless Protocol Stack

Alan Shieh

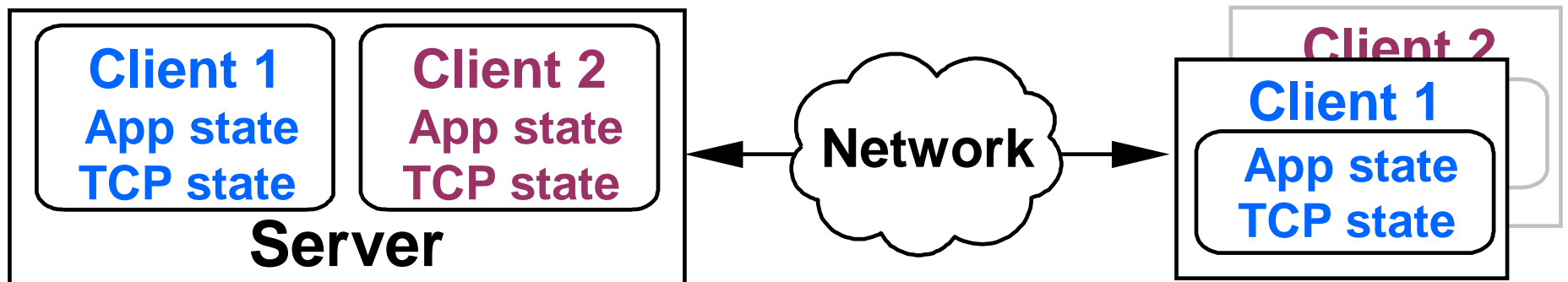
Andrew Myers

Emin Gün Sirer

Cornell University

# State placement is important

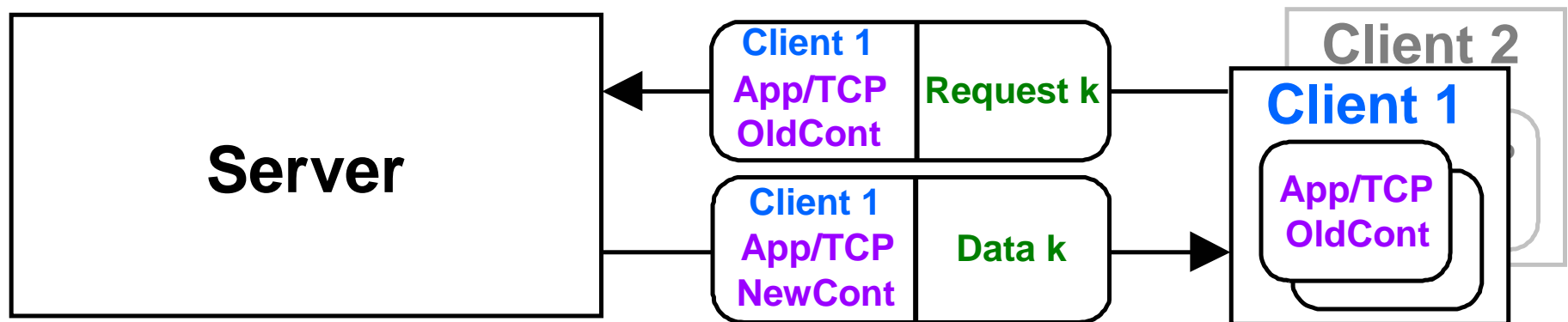
- In typical network stacks, both servers and clients hold per-connection state
  - Requires resources
  - Limits scalability
  - Increases vulnerability to DoS attack
  - Poses barriers to migration



- State problems are inherent to TCP+Sockets

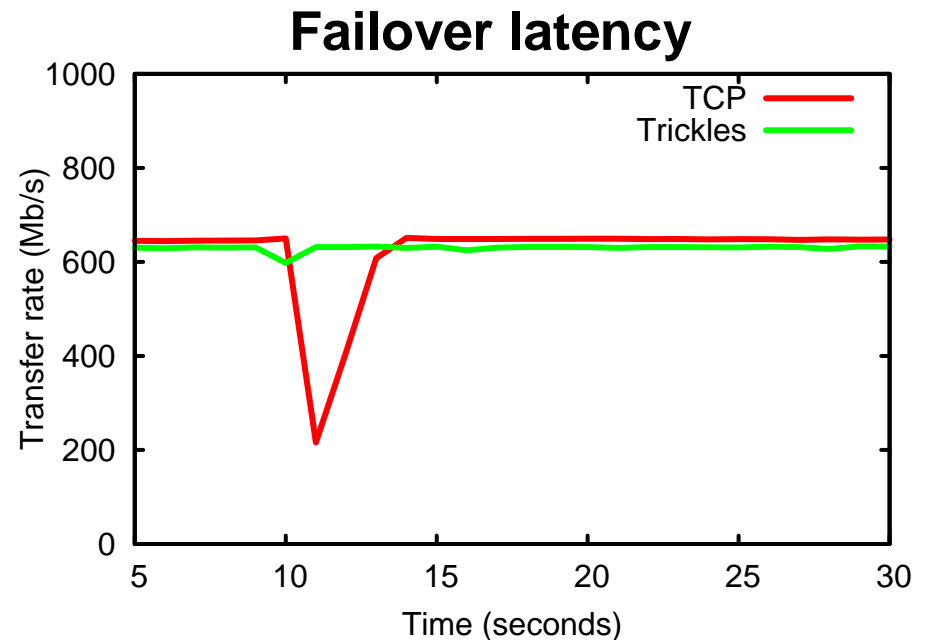
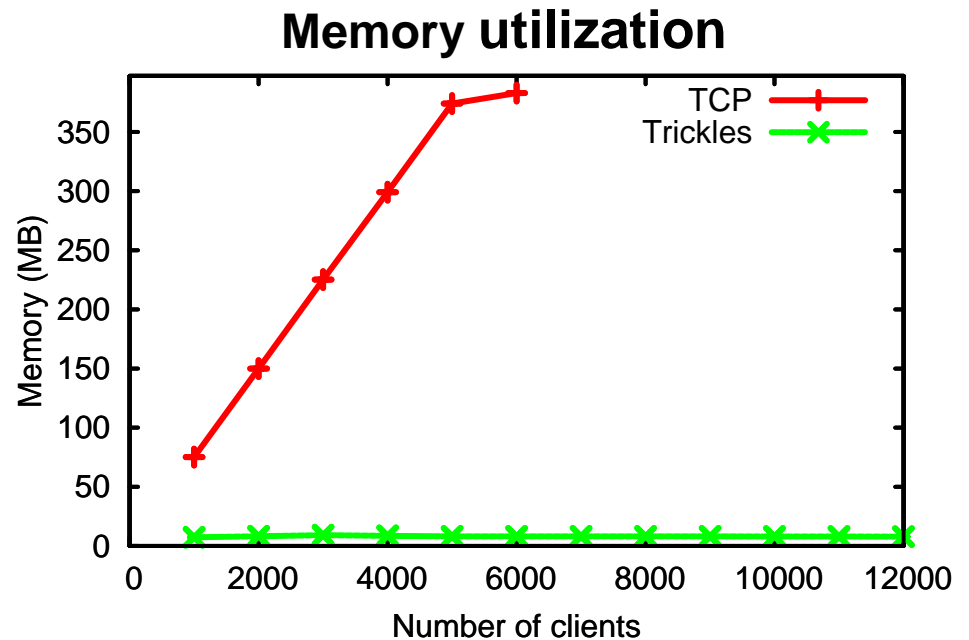
# Trickles approach

- **Make one end stateless**
- Migrate all state to client, recreate state at server via **continuations**
  - Encapsulate server state
  - Piggyback on request and data packets
  - Secure continuations with tamper-resilient MAC
- Any server replica can service any request



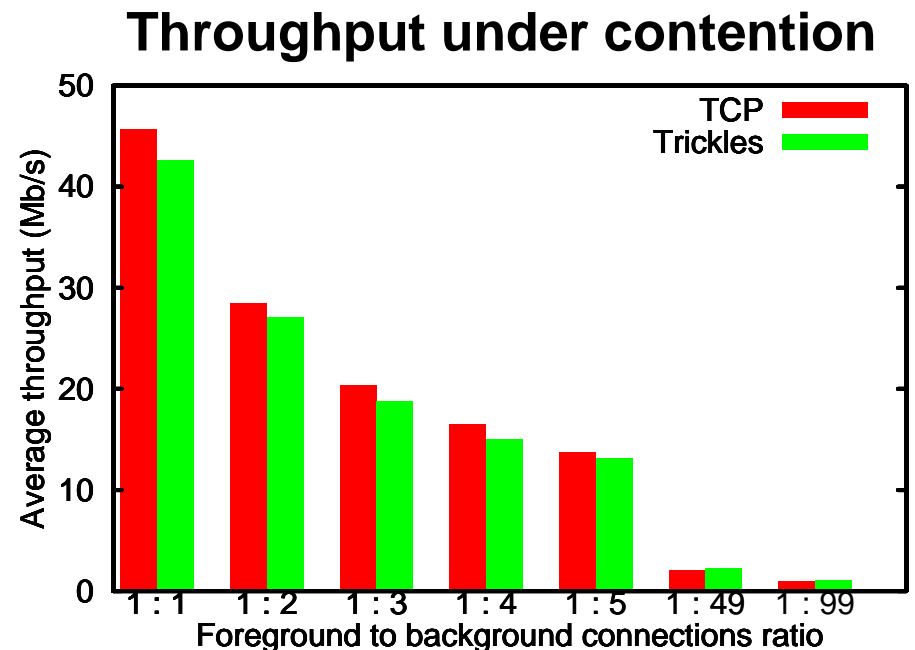
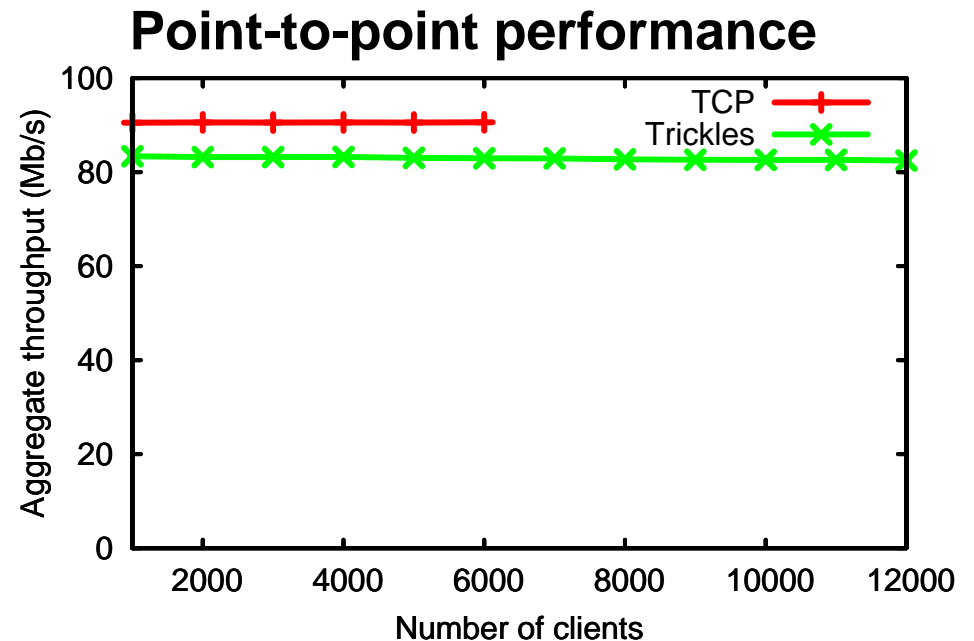
# Trickles properties

- Entire server protocol stack is **stateless**
  - Replaces inherently stateful TCP+Sockets
- Trickles enables:
  - **Transparent failover**
  - **Load balancing**
  - **Anycast services**



# Trickles properties

- **Linux implementation**
  - 15,000 LOC
  - Deployed on PlanetLab
- Comparable performance
- **Backwards-compatible & TCP-friendly**
  - Same wire format
  - Same client interface
  - Uses TCP flow control



# Summary

- Trickle enables stateless connection oriented services
  - Scale well
  - Resist DoS attacks
  - Require less resources
- Makes possible qualitatively different kinds of services
  - Geographically-distributed anycast services with long-lived connections