
# Experimental Measurements of the eXplicit Control Protocol (XCP)

Aaron Falk, Ted Faber, Eric Coe,
Aman Kapoor, & Bob Braden

*USC Information Sciences Institute*

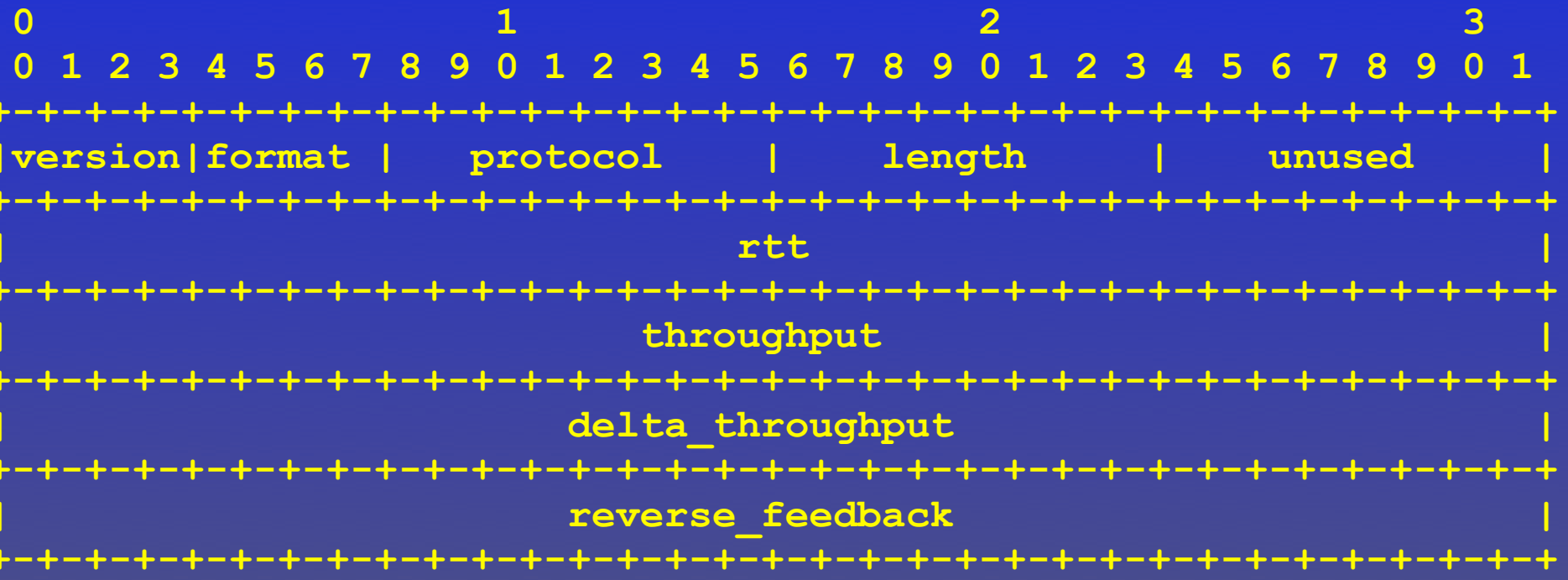
# Outline

- What is XCP?
- What is ISI doing?
- Implementation Details
- Experimental Results
- Next steps

# XCP in a Nutshell

- XCP is a new congestion control protocol developed by Dina Katabi
- End-systems tell routers what throughput they'd like to send at
- Routers make a per-flow allocation, inspect incoming packets, and reduce the throughput request to match the allocation (if necessary)

# The Congestion Header



# What's So Cool About XCP?

- In simulation...
  - XCP fills the bottleneck pipe much more rapidly than Van Jacobson congestion control (VJCC)
  - XCP rapidly converges to fair allocation of bottleneck bandwidth
  - XCP gets better bottleneck link utilization than VJCC for large BDP flows

# What's So Cool About XCP?

- XCP maintains tiny queues
- XCP is more stable than VJCC at long RTTs
- Future/other functionality:
  - Unfair allocations (e.g., QoS, low priority)
  - CC for other protocols (e.g., realtime)

# ISI's XCP Development

- Our objective: Take XCP from theory to reality
- To get there:
  - Build & test a kernel implementation
  - Evaluate the cooler aspects:
    - *Rapid convergence*
    - *Good performance over large BDP, RTT*
  - Write a protocol specification, mature the protocol
  - Move ns-2 simulation code into distribution
  - Work with the community (researchers, vendors, operators, IETF)
  - Develop deployment strategies

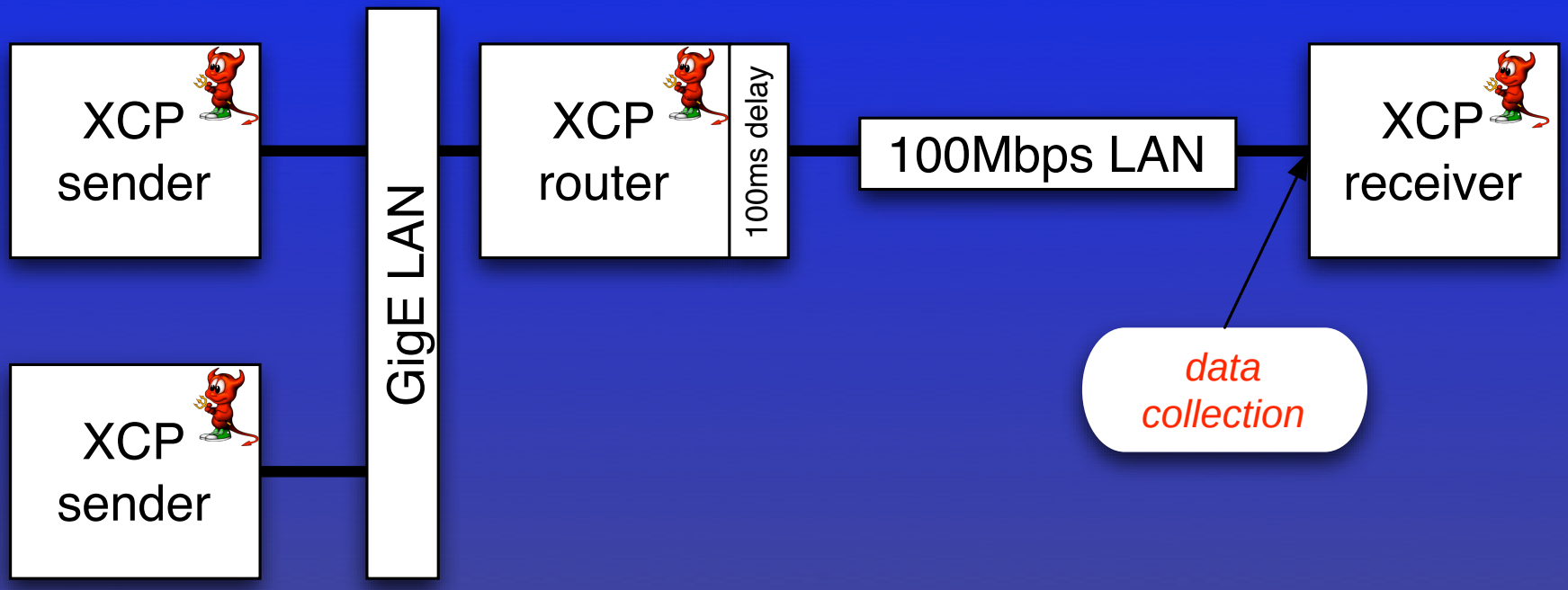
# Our XCP Prototype

- Congestion header is placed between TCP and IP (layer 3.5)
- Application opens socket to protocol 'XCP' to get TCP using XCP congestion control
- Router operates XCP on output queue

# Implementation Details

- End System
  - FreeBSD ver. 4.8 kernel implementation
  - XCP code modifies TCP cwnd value
  - (using cwnd in header now, switching to throughput soon)
- Router
  - FreeBSD ver. 4.8 kernel implementation
  - Dummynet used to provide separate queues for TCP and XCP packets
  - All integer math (requires lots of scaling)
  - Many router parameters stashed into debug extensions to congestion header

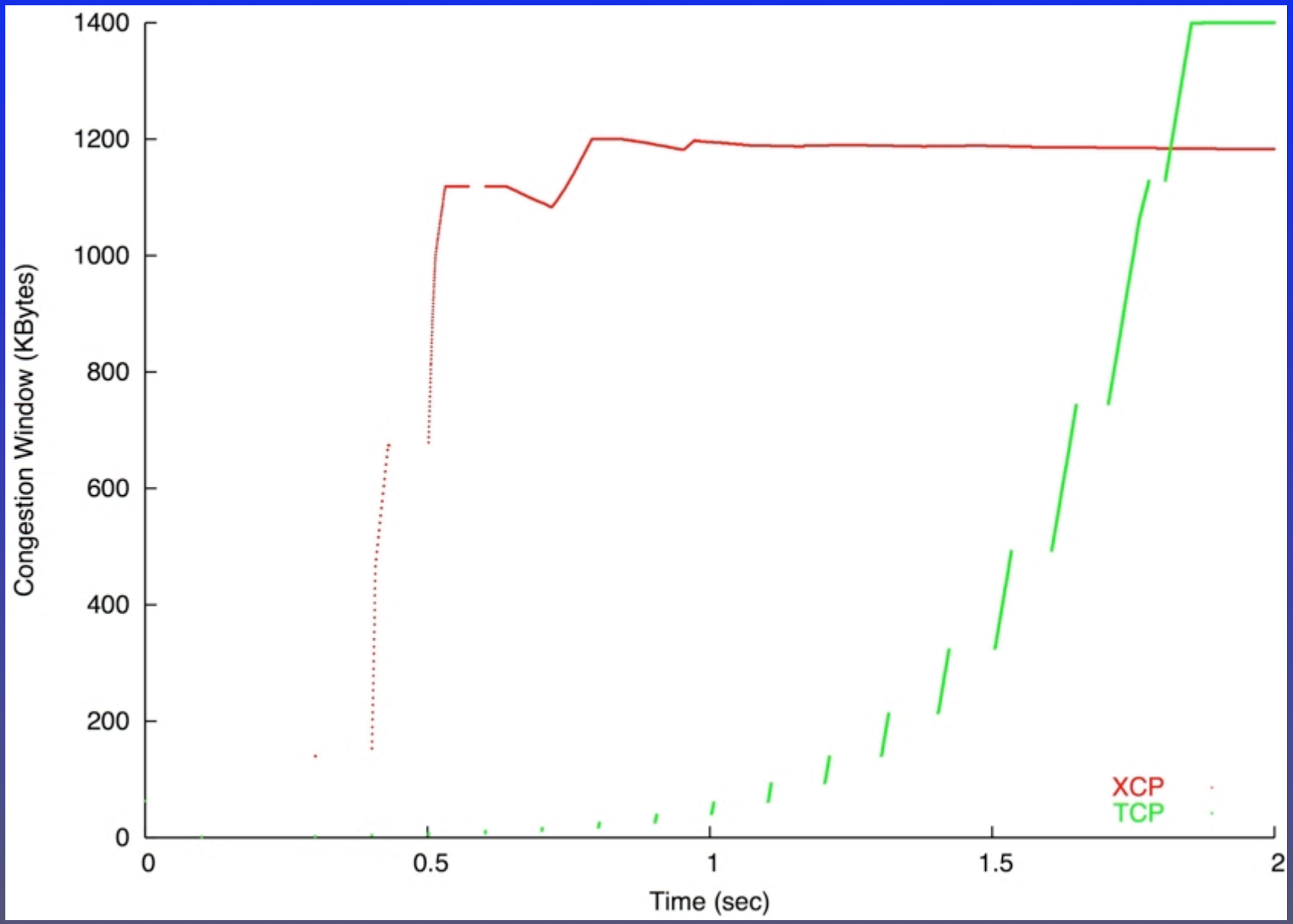
# Testbed Topology



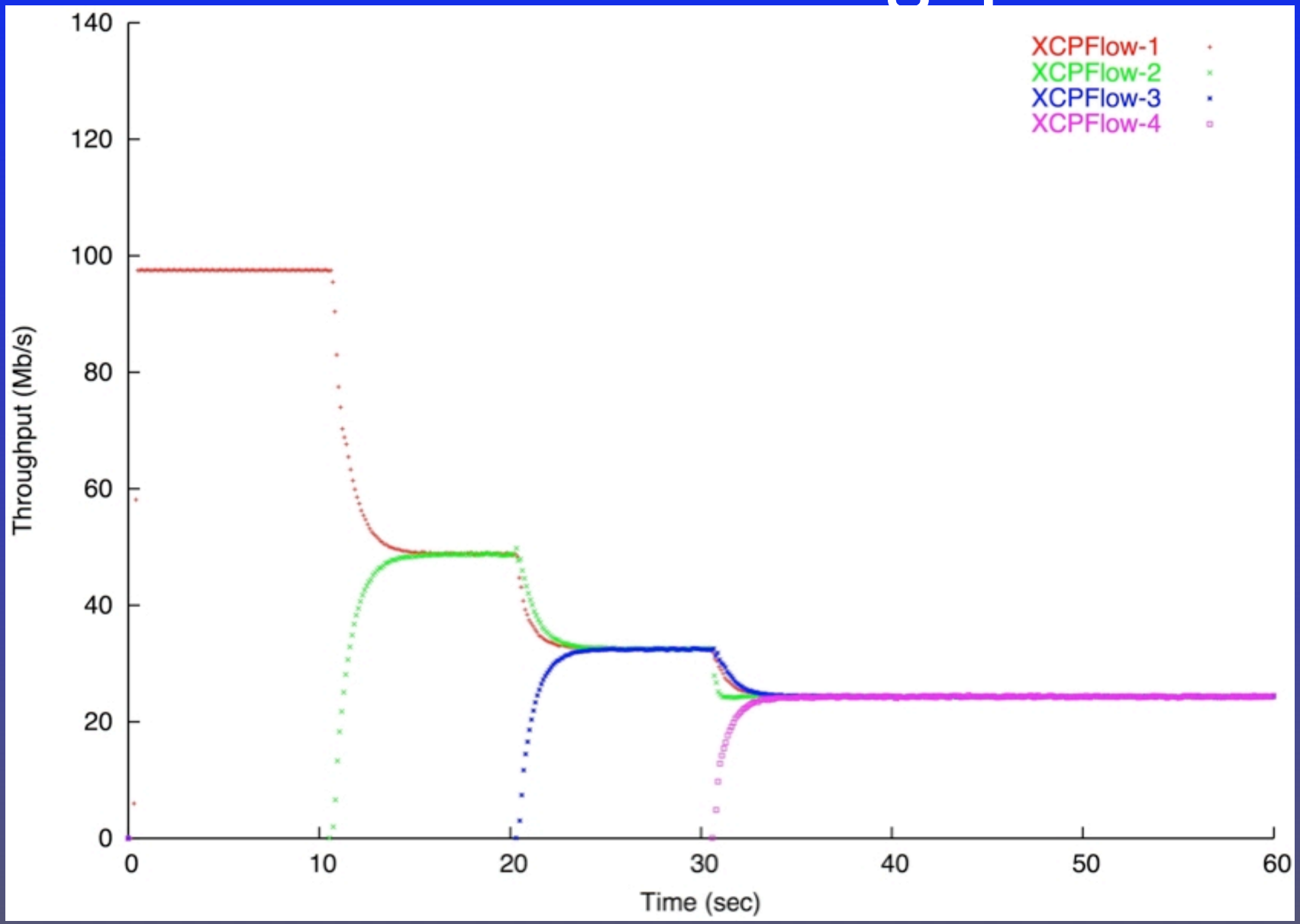

- PCs: 2.8 GHz, dual PCI-X, FreeBSD4.8, 512MB
- Dummynet 100ms delay on ACK flow
- Router buffering: 5kpkts in, 20kpkts out

# Experimental Results

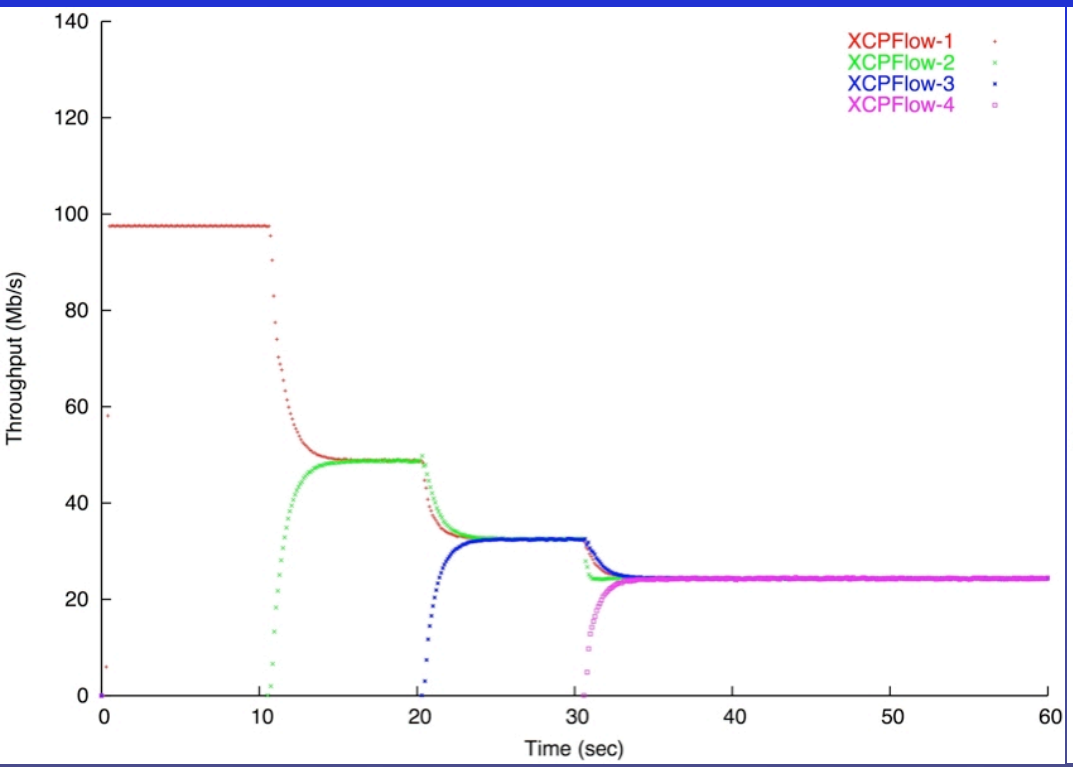
# XCP vs. TCP startup behavior

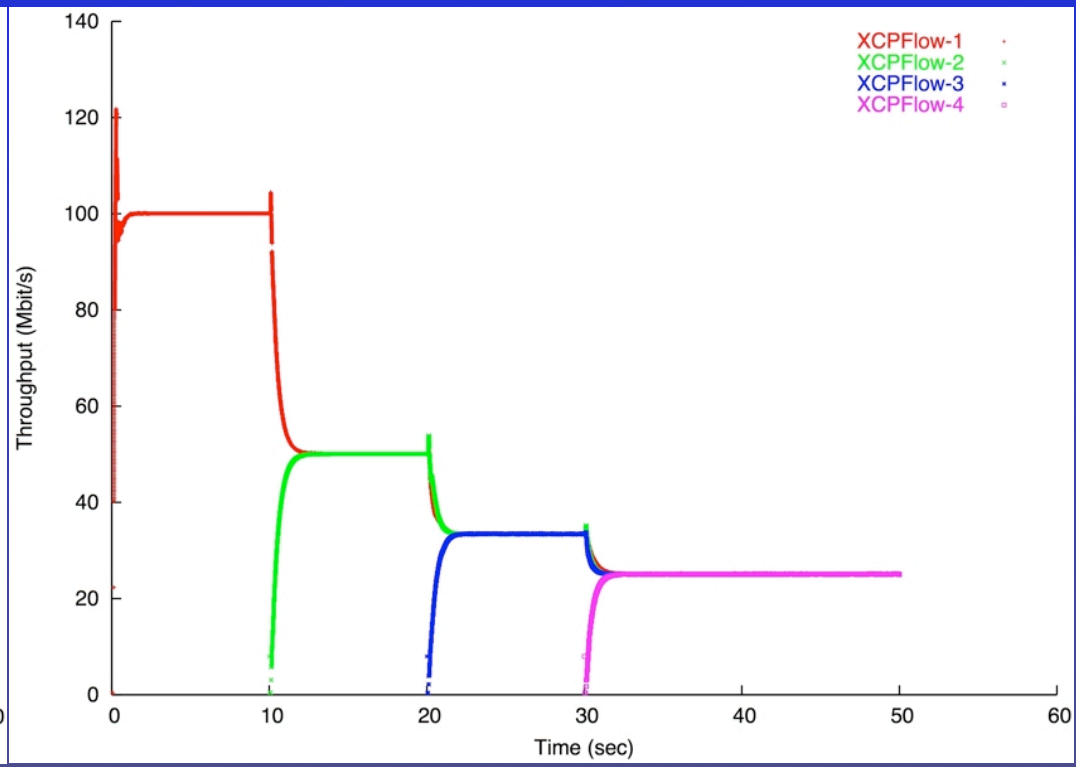


# Four XCP Flows Fairly Share Bottleneck Throughput



# XCP Throughput Compared to Simulation Results

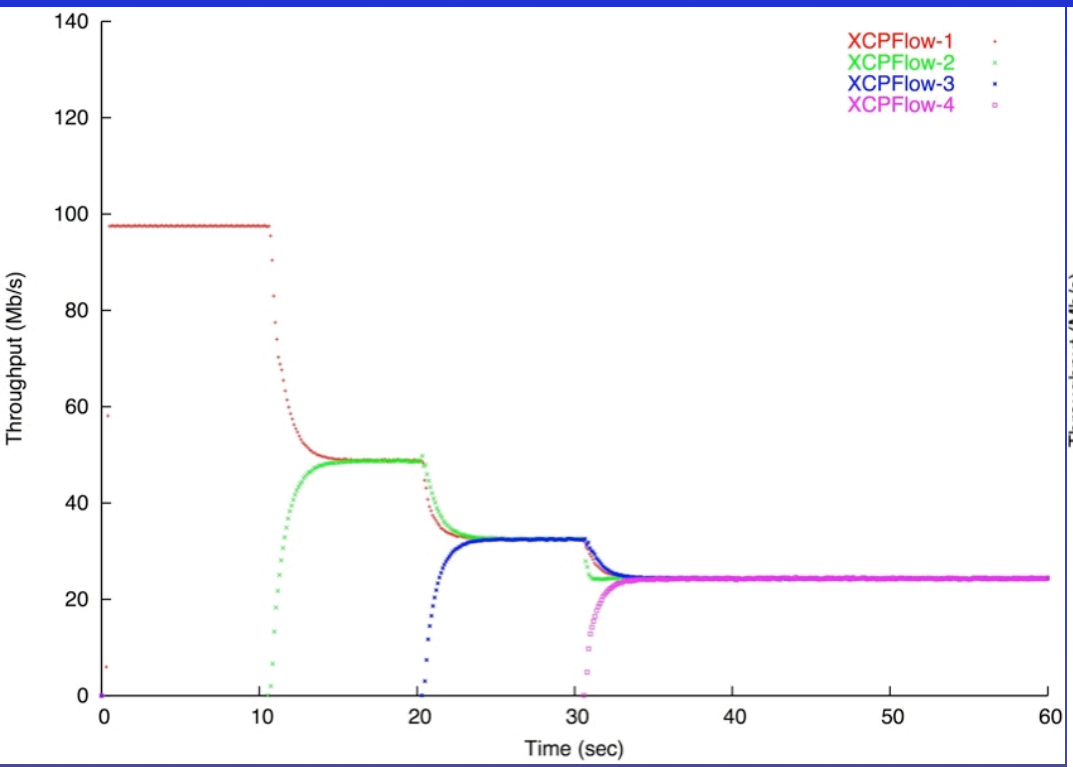


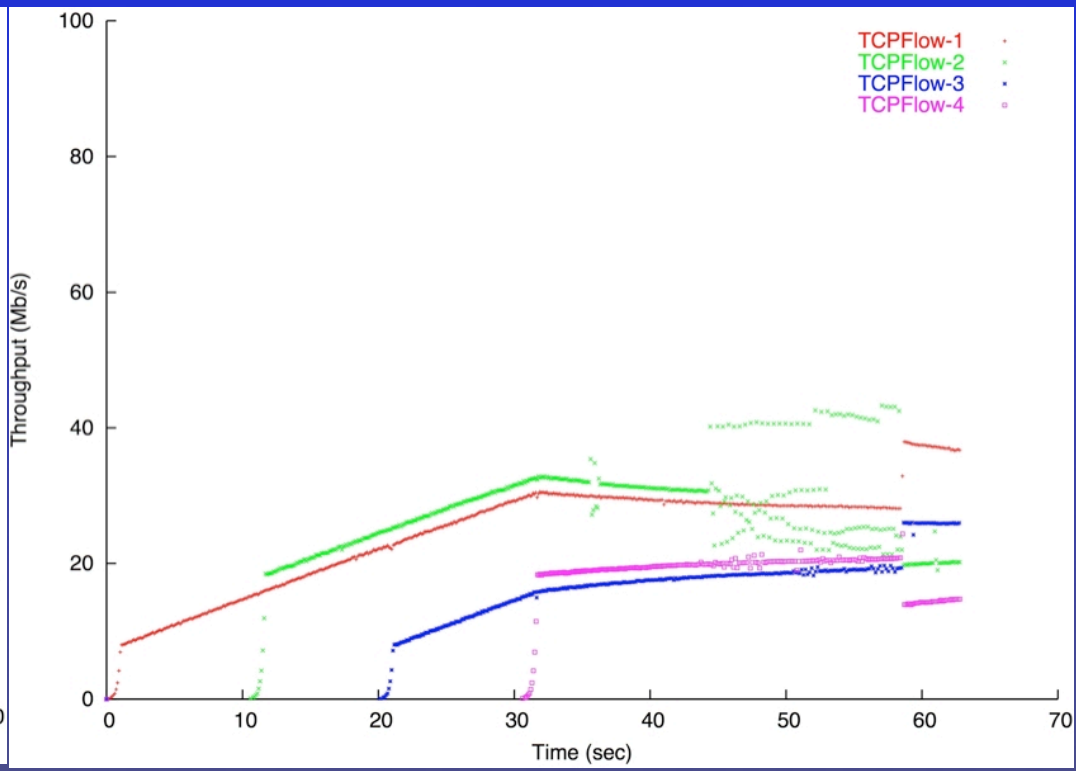

XCP Measured

XCP Simulated

# XCP Throughput Compared to TCP

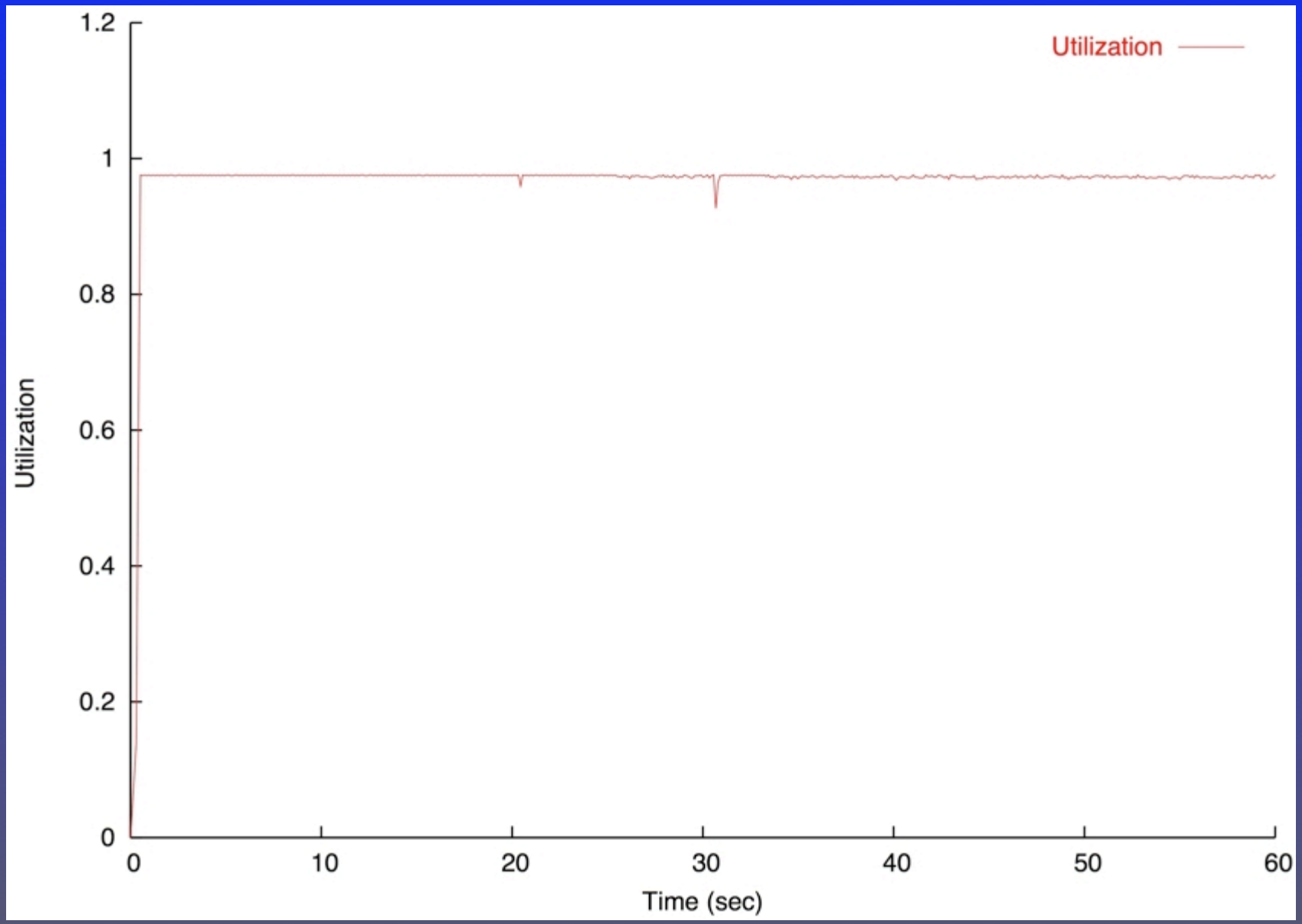



XCP Measured

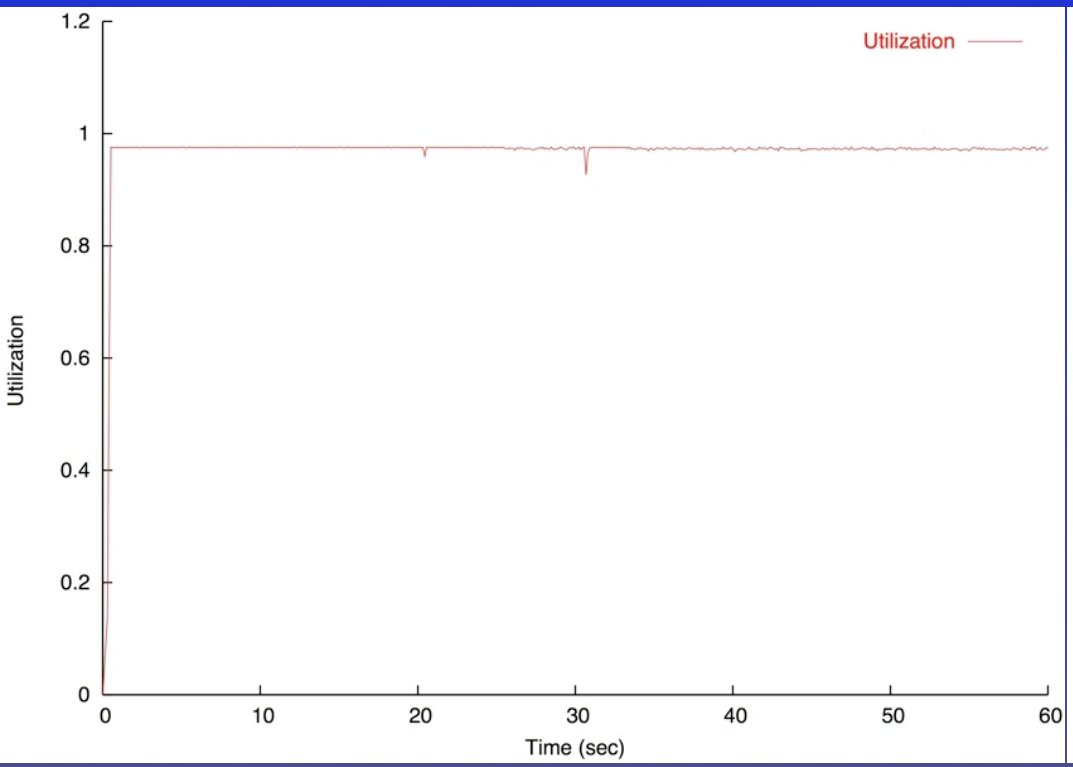


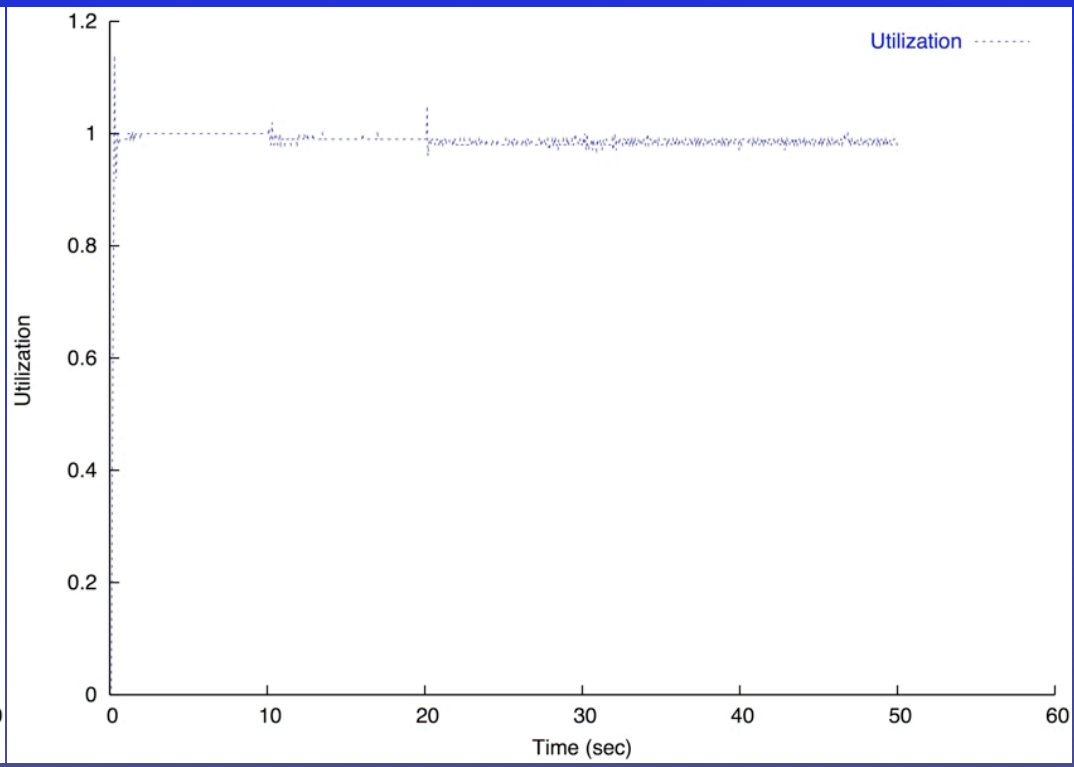

TCP Measured

# Link Utilization Is Maintained As New Flows Arrive



# Link Utilization Compared to Simulation Results

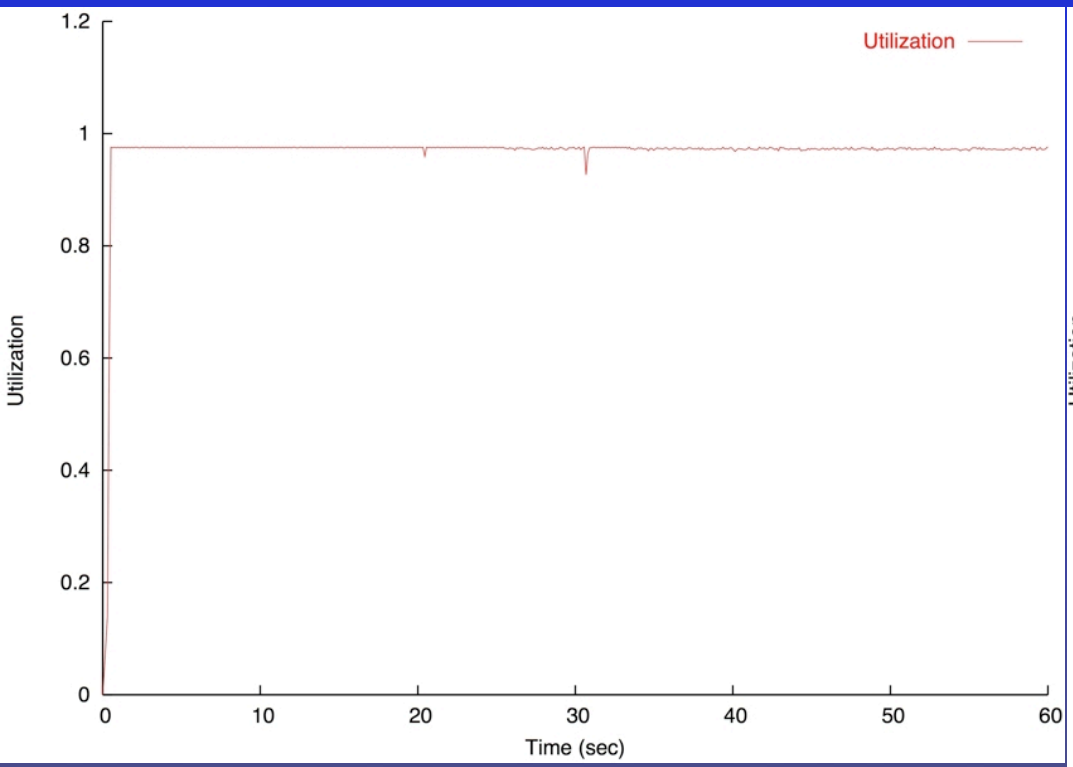


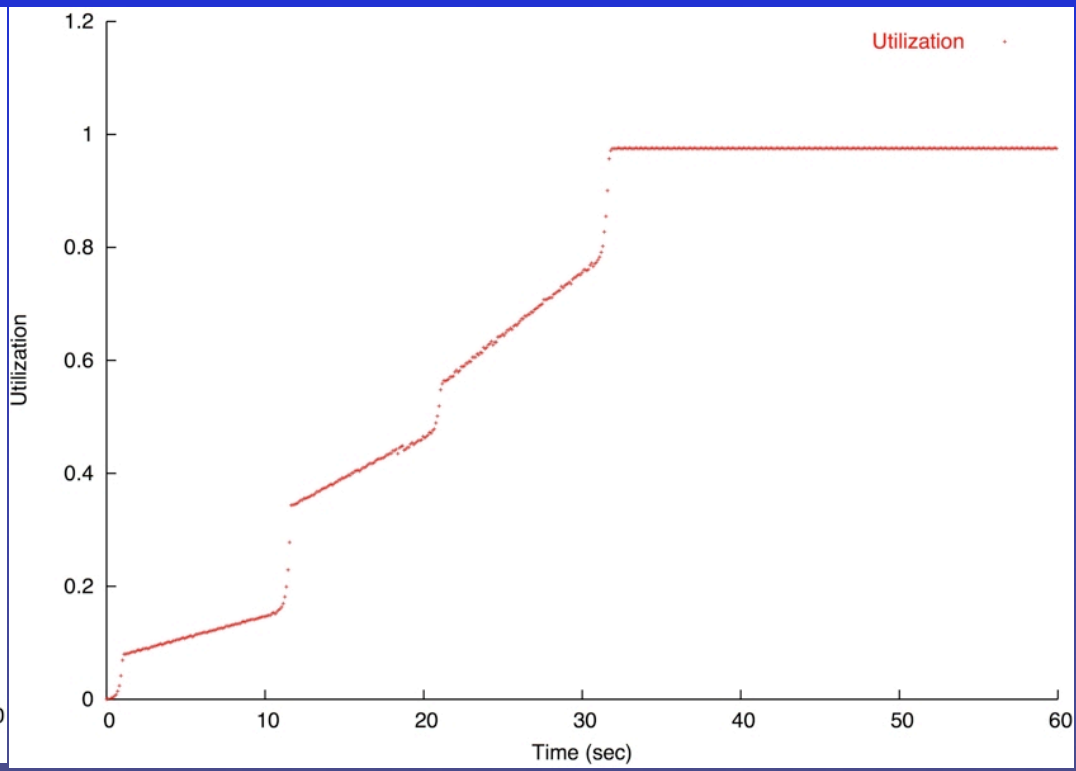




XCP Measured

XCP Simulated

# Link Utilization Compared to TCP

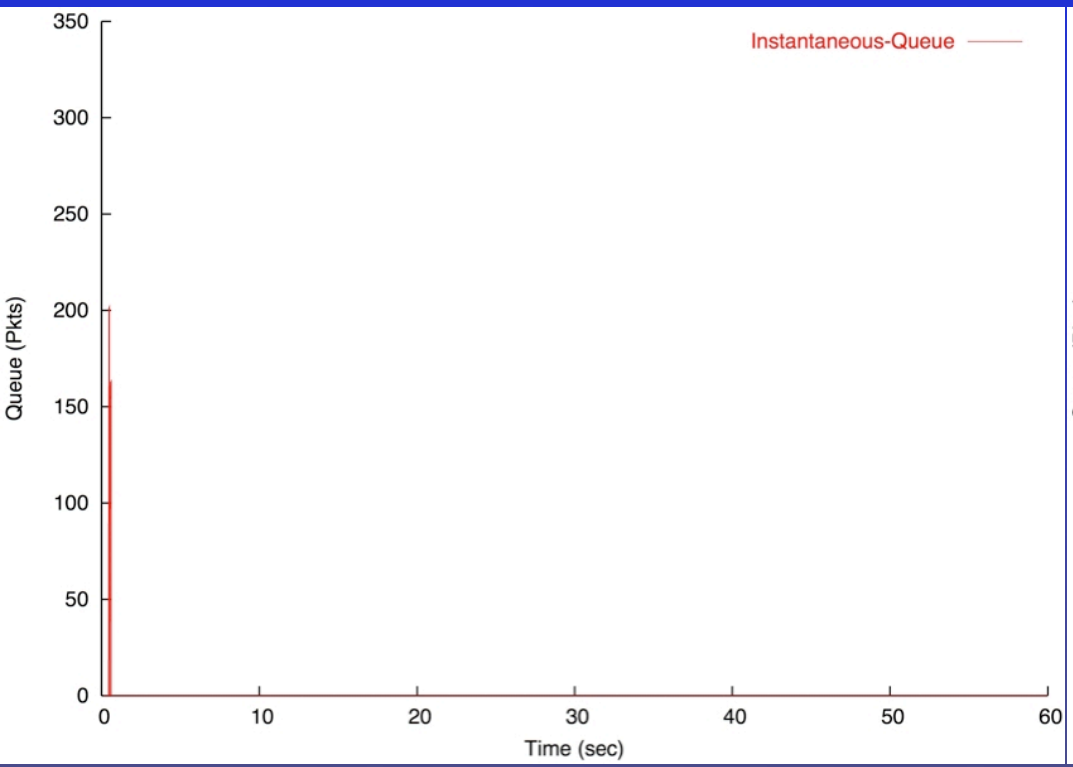


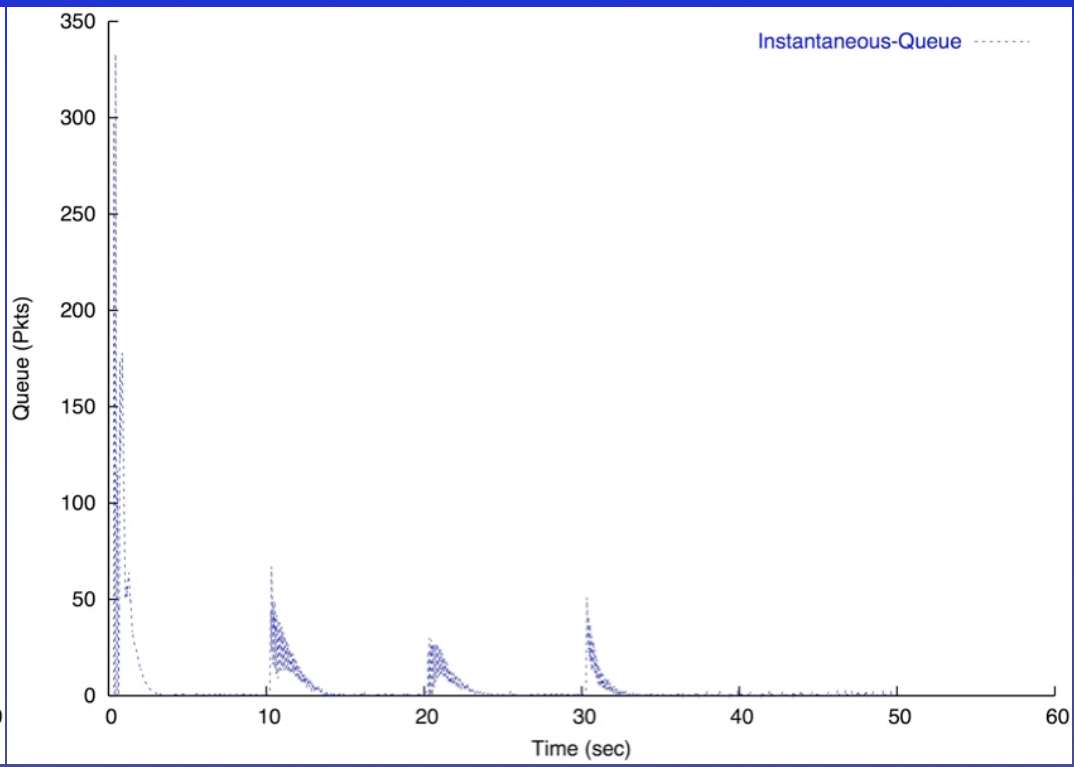

XCP Measured

TCP Measured

# Queues Stay Small As New Flows Arrive

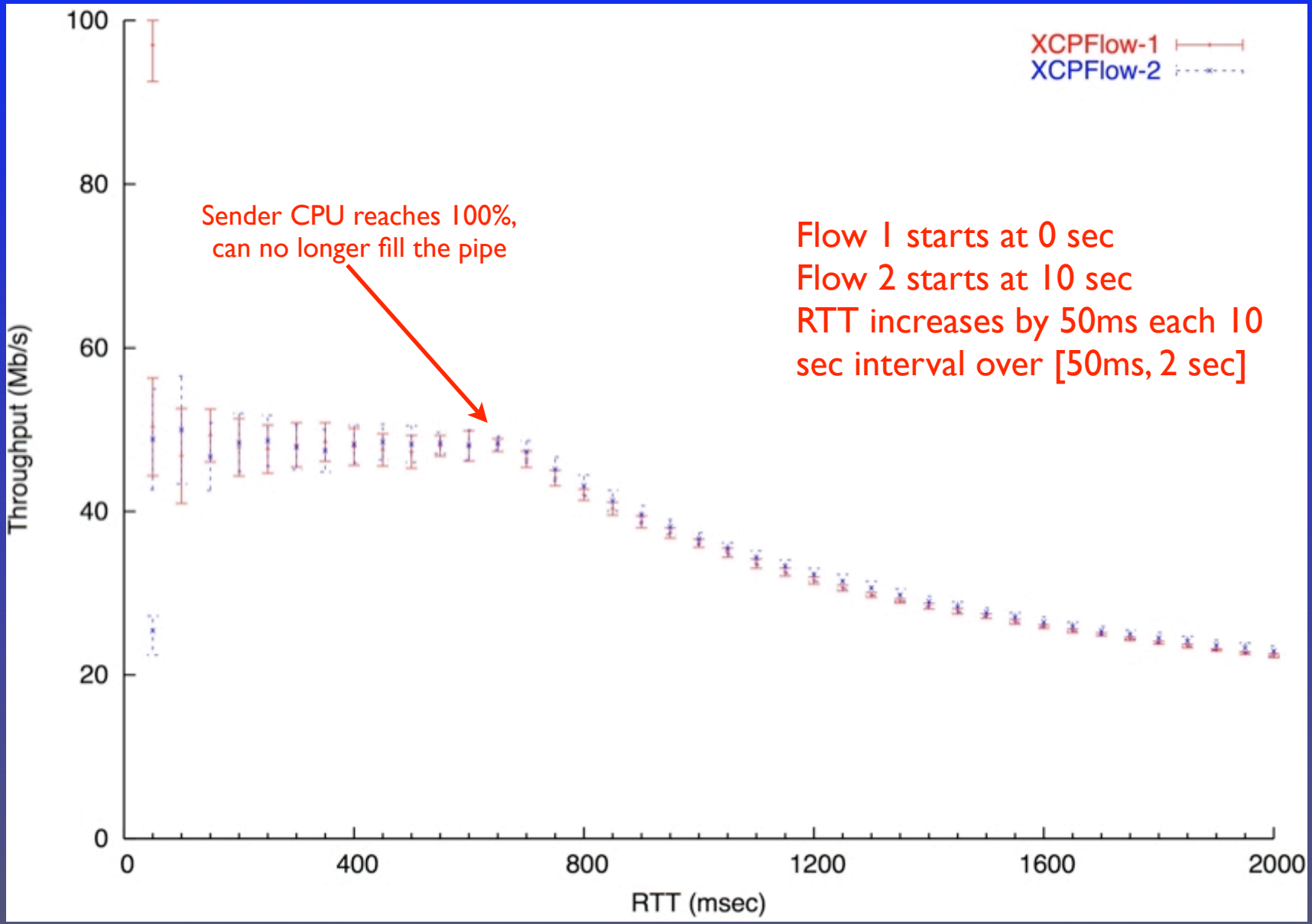



XCP Measured

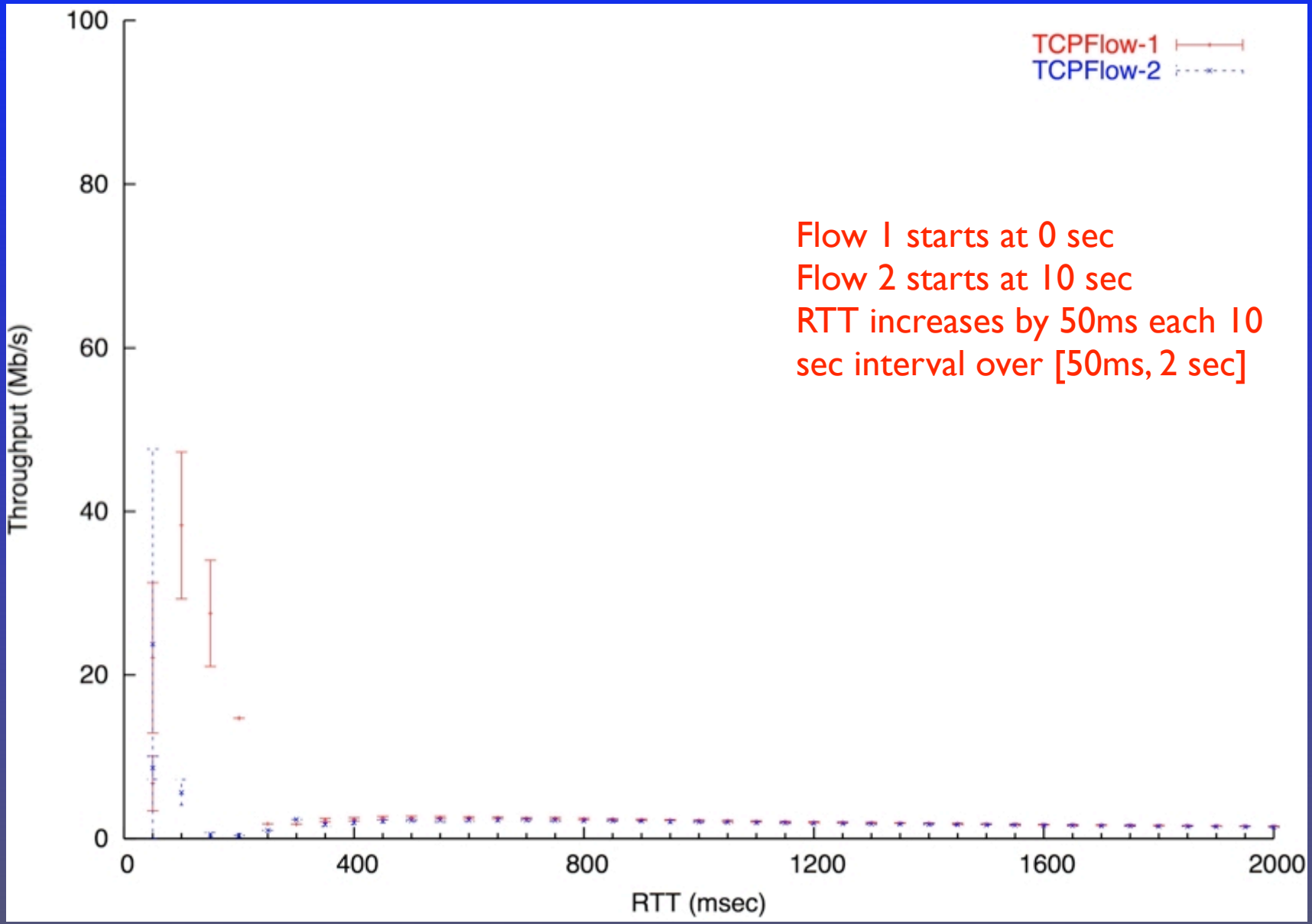



XCP Simulated

# XCP is Stable as RTT Increases...



# TCP Doesn't Do As Well




Flow 1 starts at 0 sec
Flow 2 starts at 10 sec
RTT increases by 50ms each 10 sec interval over [50ms, 2 sec]

# Methodology

- Utilization & Throughput
  - tcpdump at receiver on all packets
  - Sum packet size over 100ms intervals
- Throughput vs. RTT
  - Sum packet size over interval set to current RTT
  - Hold RTT for 10 sec

# Next Steps

- Continue evaluating performance, stability, and fairness under more widely varying conditions
  - Mixing flow RTT, bandwidth
  - Networks with multiple routers
  - Scenarios with link errors, moving bottlenecks

# Next Steps

- Examine heterogeneous networks
  - XCP & TCP co-existance
  - Performance with non-XCP routers
  - Performance with layer 2 queues

# Next Steps

- Develop a more general router model
- Resolve some protocol issues
  - IPsec, MPLS, header formats
- Experiment with deployment scenarios
  - E.g., running XCP in a cloud

# Summary

- Early measurements match simulated results
- XCP fairly allocates bottleneck bandwidth to multiple flows
- XCP dynamically reallocates bottleneck bandwidth as flows arrive and depart
- XCP remains stable as RTT varies by 4000%

# People

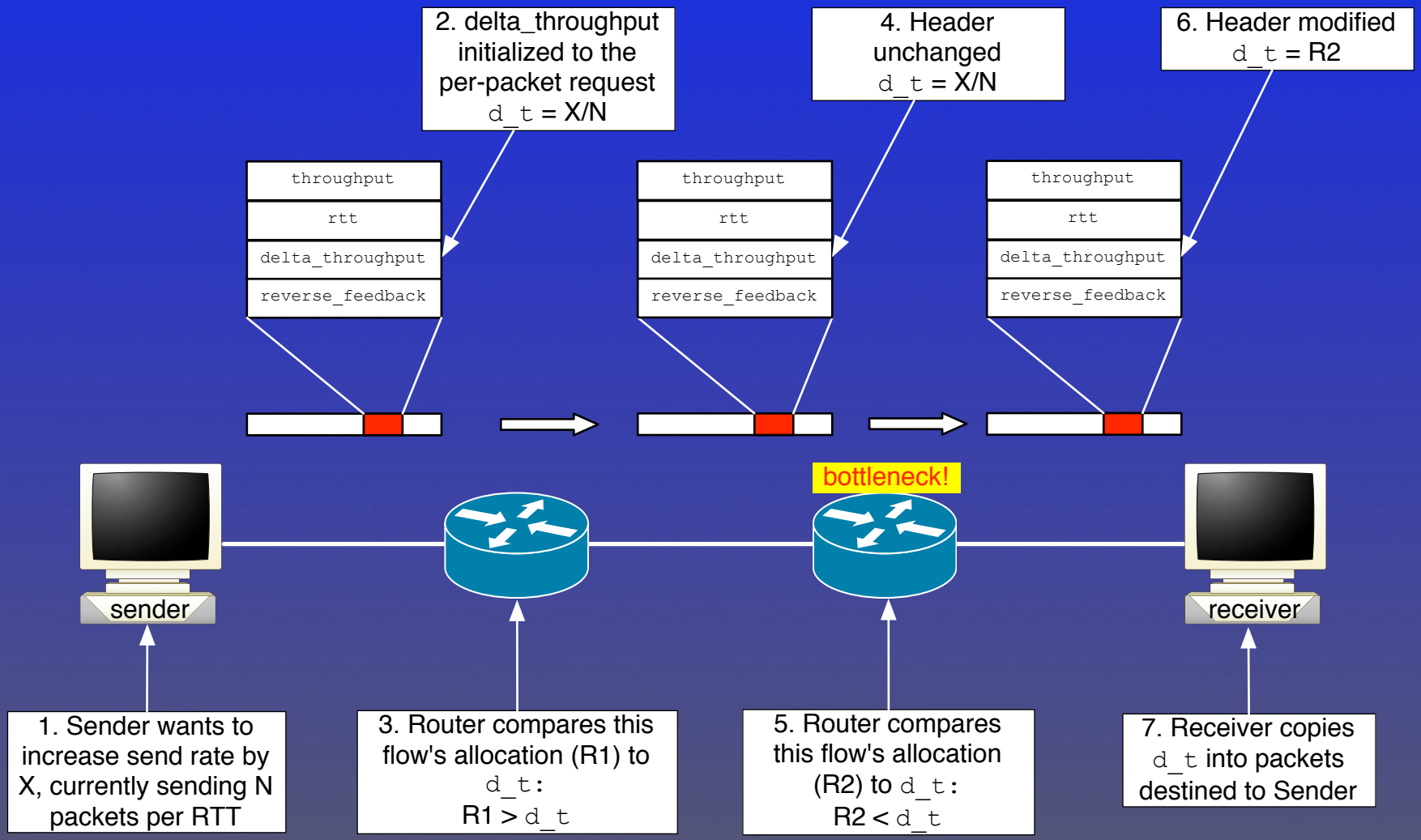
- Aaron Falk, ISI, project lead
- Ted Faber, ISI
- Bob Braden, ISI
- Eric Coe, student
- Aman Kapoor, student
- Dina Katabi, MIT
- John Wroclawski, MIT

# XCP Project Info

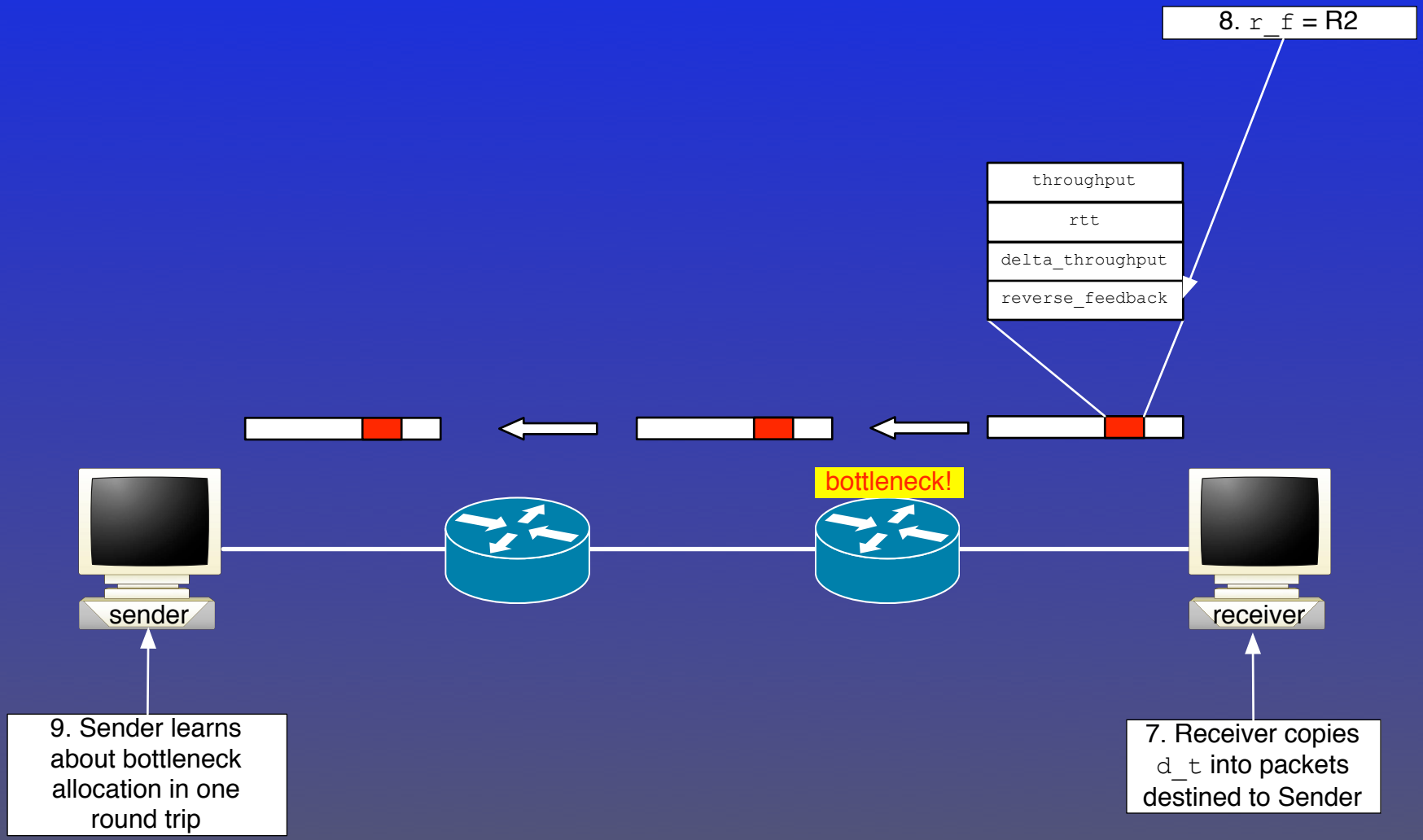
- http://www.isi.edu/isi-xcp
  - source code
  - draft specification
  - mailing list information

# Backup Slides

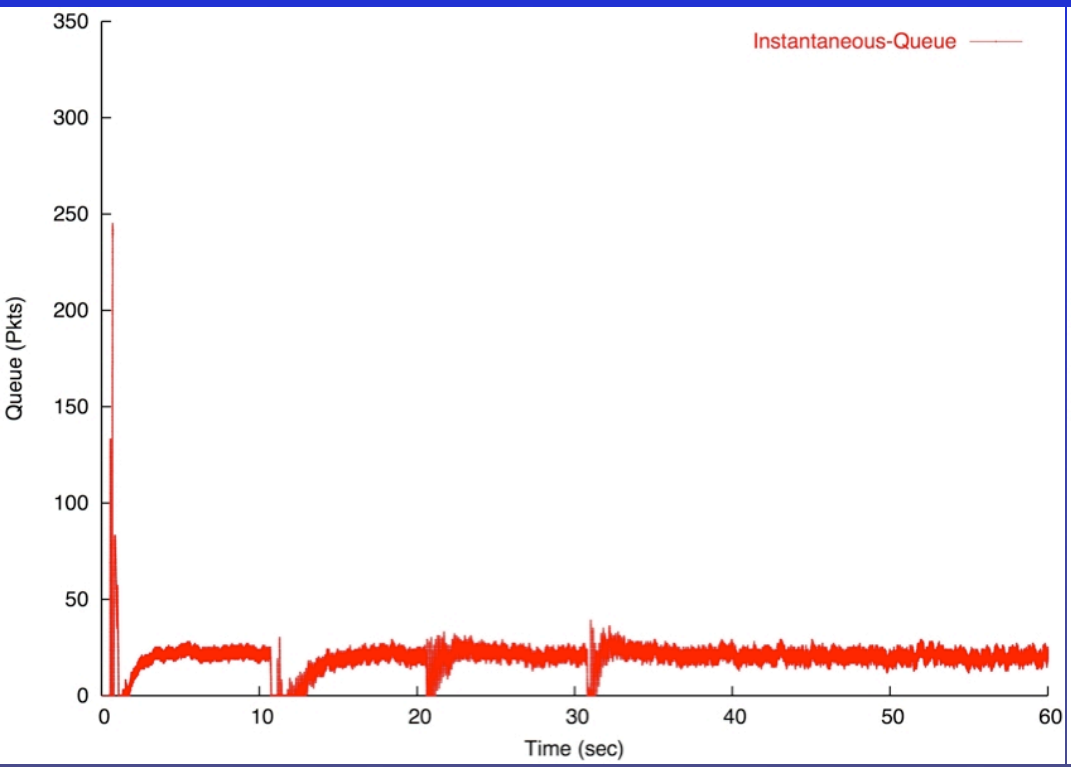
# XCP Feedback Loop

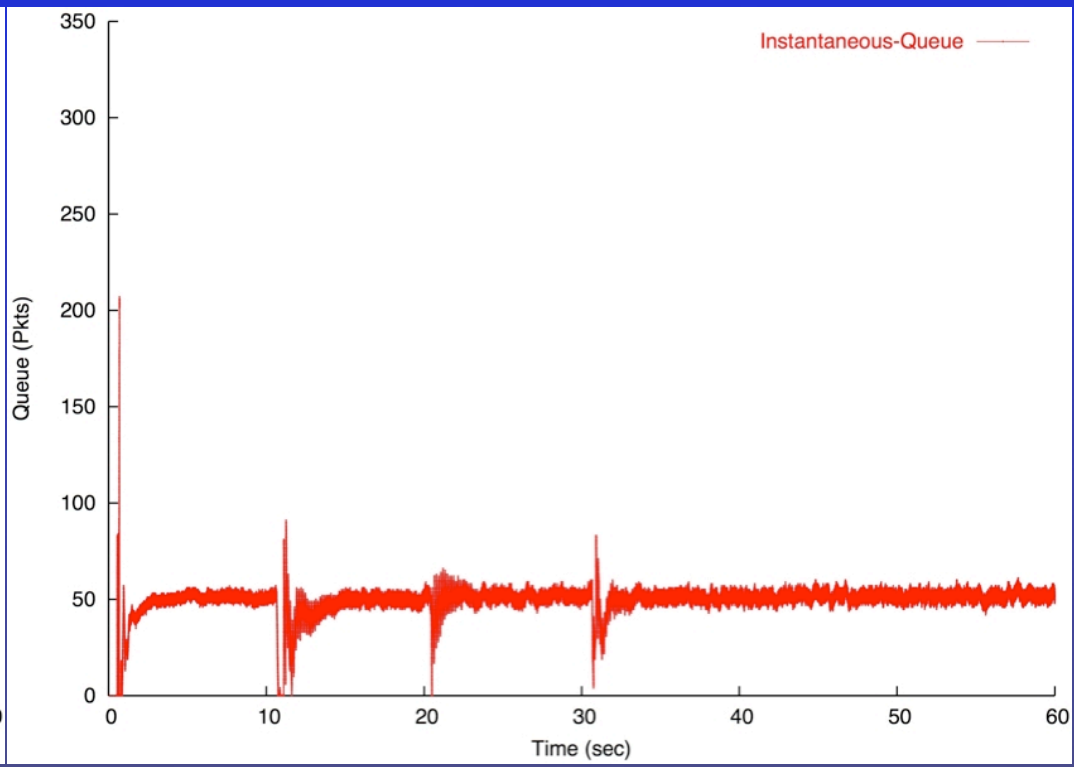


# XCP Feedback Loop



# Router Queues Remain Stable When Capacity is Over-Estimated




Router capacity set to 98.5Mbps

Router capacity set to 100Mbps