Methodologies for Generating HTTP Streaming Video Workloads to Evaluate Web Server Performance

Jim Summers, Tim Brecht, Derek Eager,

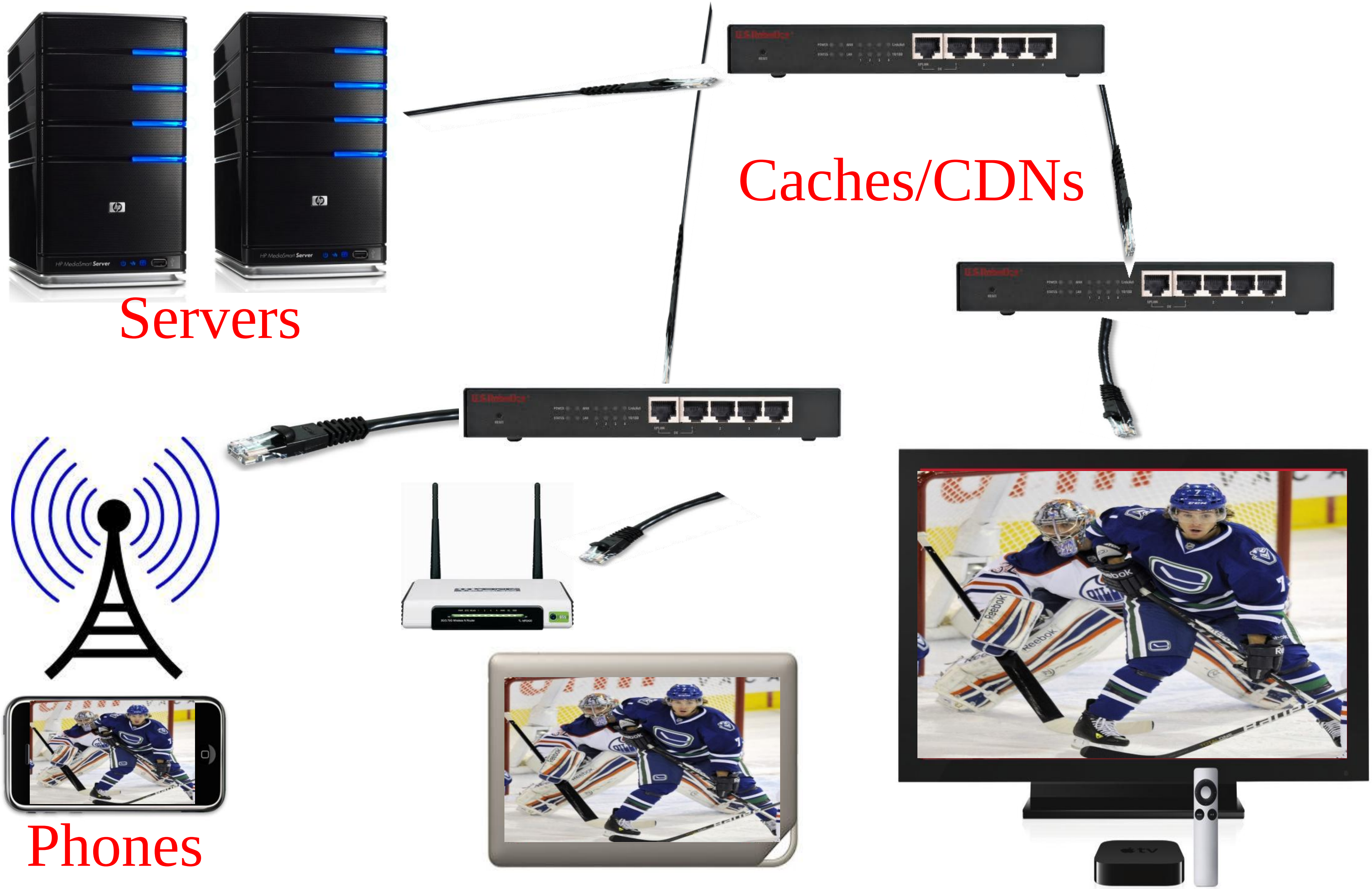
and Bernard Wong

University of Waterloo

University of Saskatchewan

HTTP Streaming Video

HTTP Ecosystem



Servers

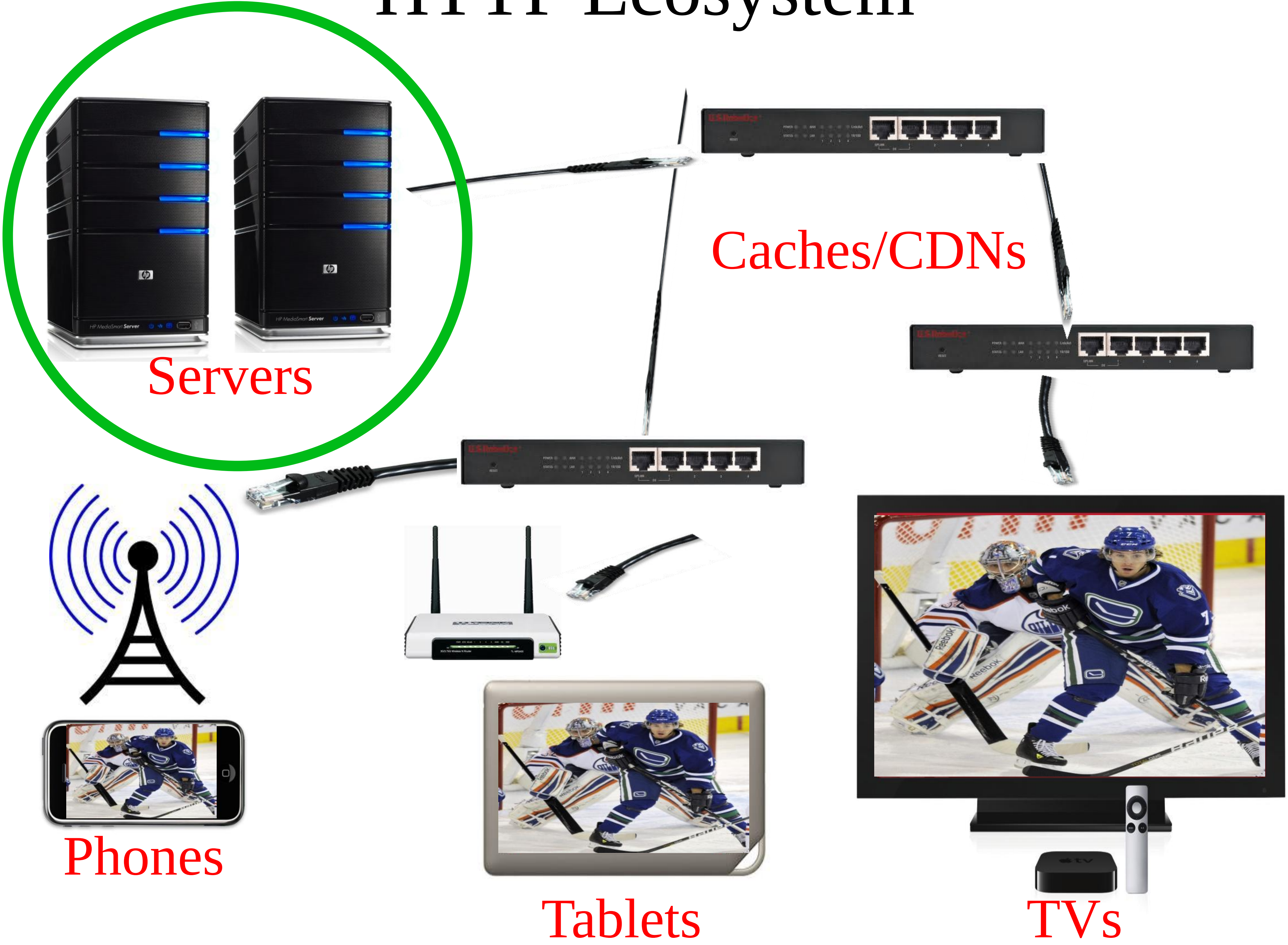
Caches/CDNs

Phones

Tablets

TVs

HTTP Ecosystem



Servers

Caches/CDNs

Phones

Tablets

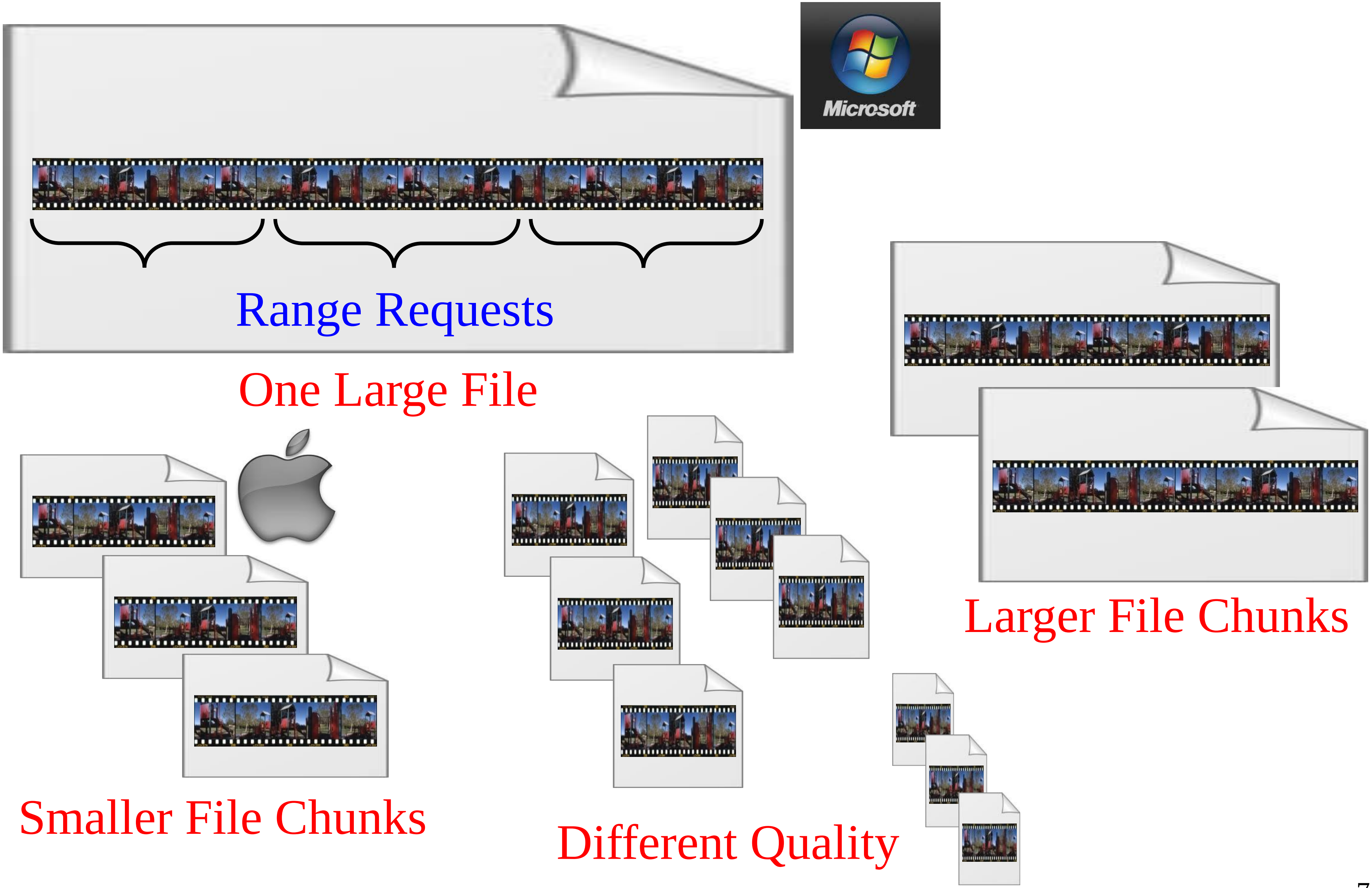
TVs

Buffering …

Video and Client Characteristics

- Video is buffered
    - Start at full speed
    - Remainder at rate of consumption
- Clients usually do not watch until the end
- Change quality of video
- Pause, skip forward or back
- Long-tail distribution of content

Storage and Request Options



Range Requests

One Large File

Larger File Chunks

Smaller File Chunks

Different Quality

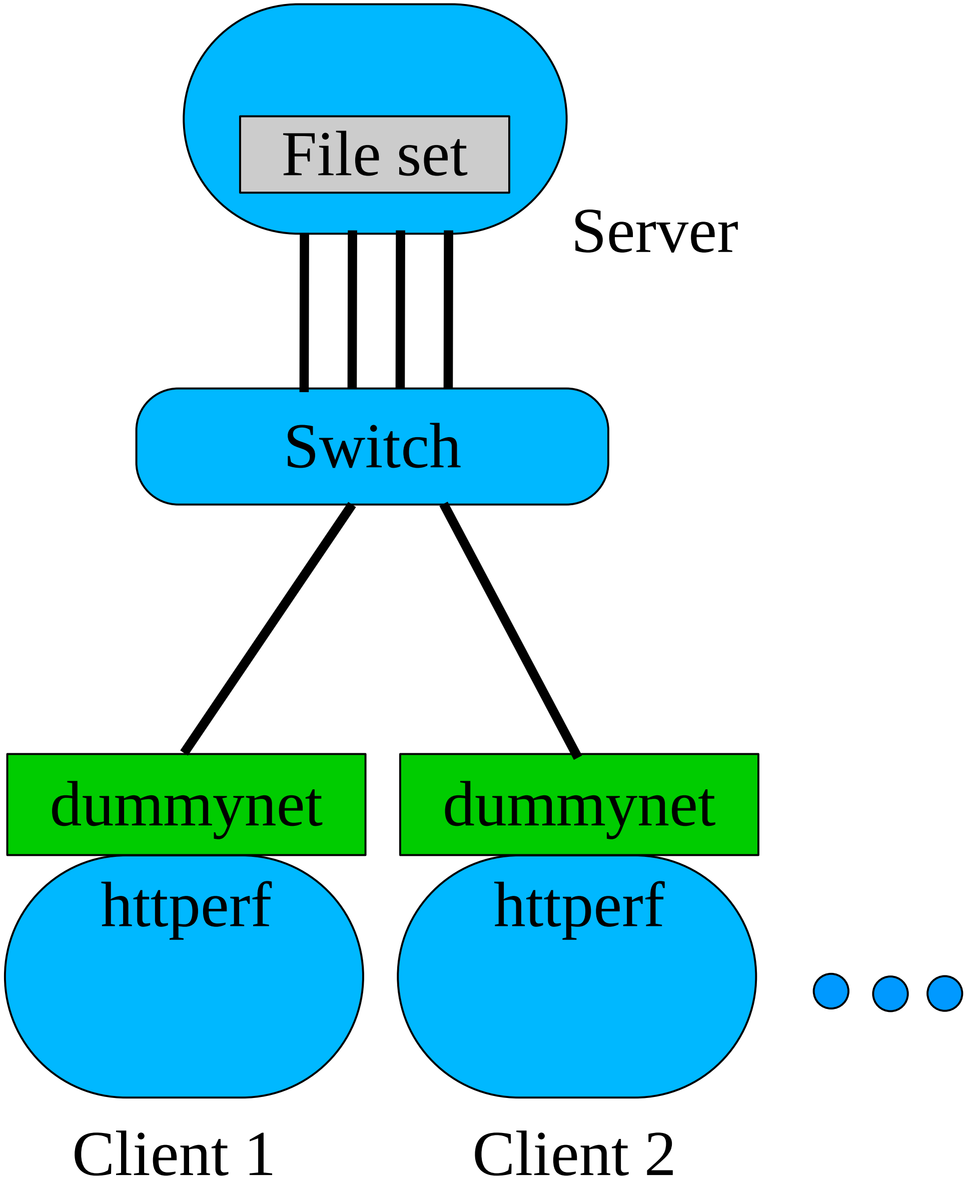
Methodology Goals

- Flexible
  - Many types of videos and users
- Representative
  - Based on workload measurements and studies
  - Limited client network access
- Practical
  - Experiments repeatable
  - Reasonable execution time
  - In a lab

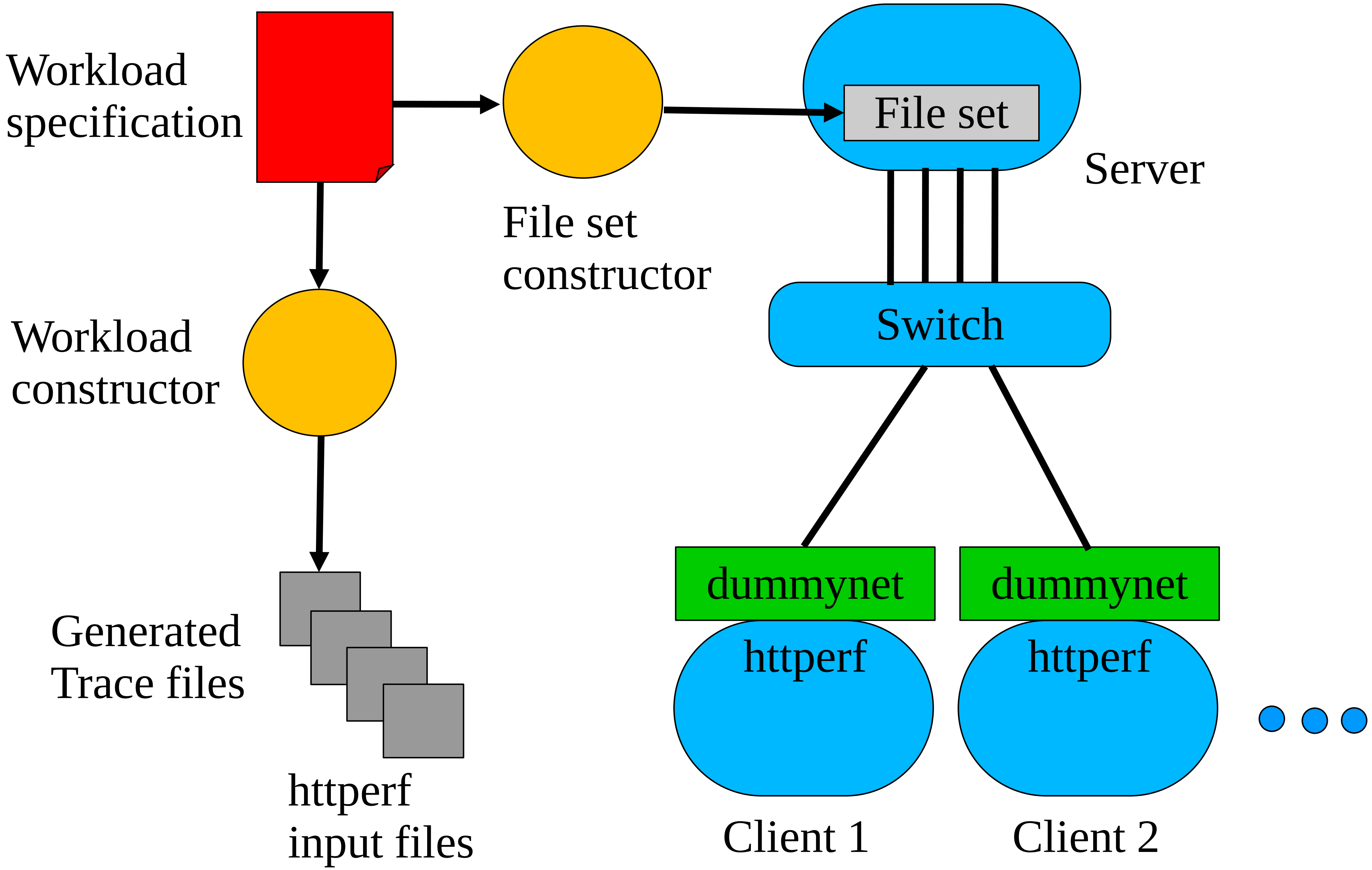
Related Work

- Benchmarks and Workload Generators
  - YouTube Workload generation [Abhari et al, ’10]
  - SPECweb2009
  - BenchLab [Cecchet et al, WebApps ’11]
- Measurement Studies
  - YouTube Everywhere [Finamore et al, IMC ’11]
- Client Testing
  - DASH Dataset [Lederer et al, MMSys'12]

Environment



Overview of the Methodology



Workload specification

File set constructor

File set

Server

Switch

Workload constructor

Generated Trace files

httperf input files

dummynet

httperf

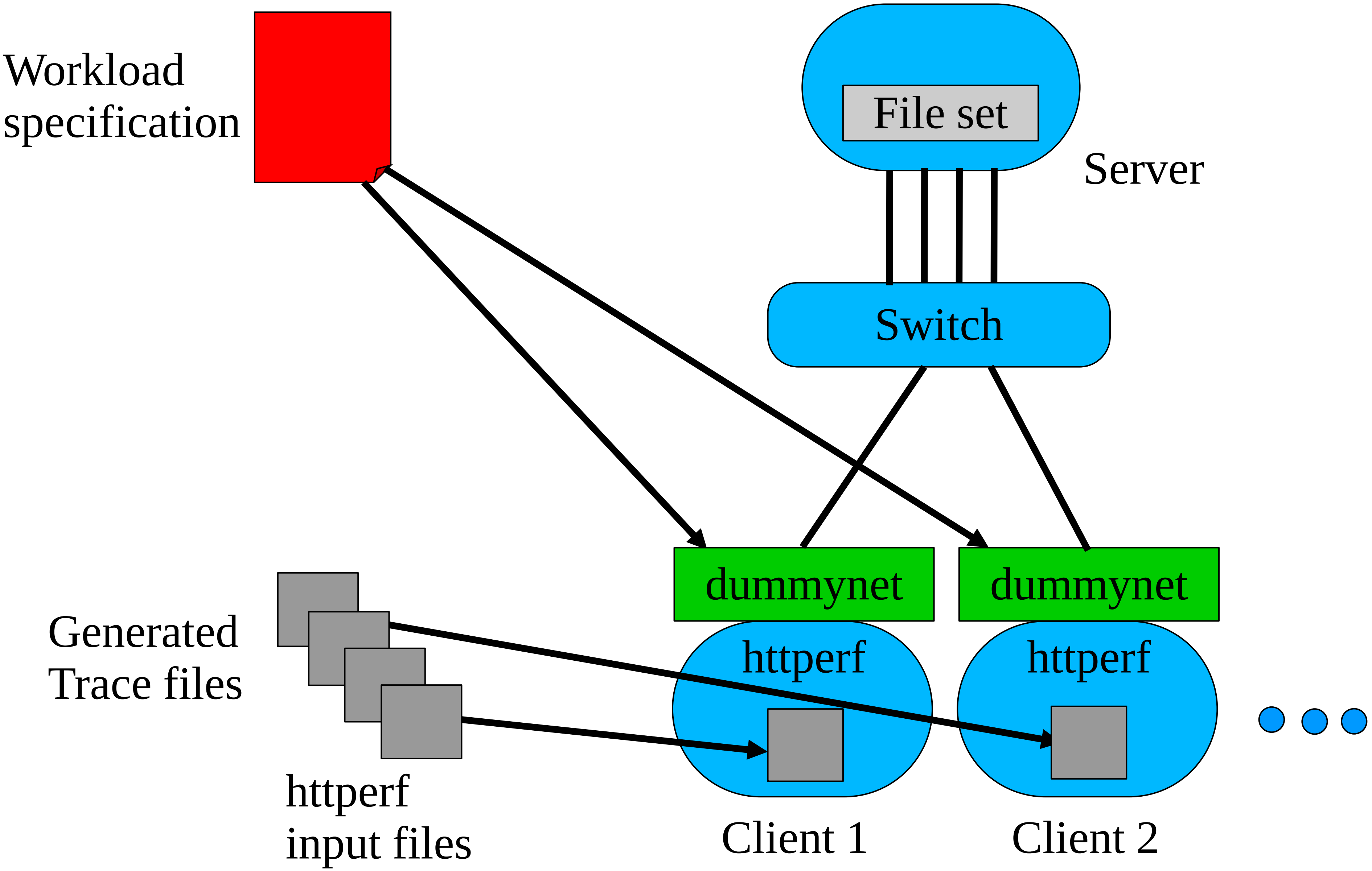
dummynet

httperf

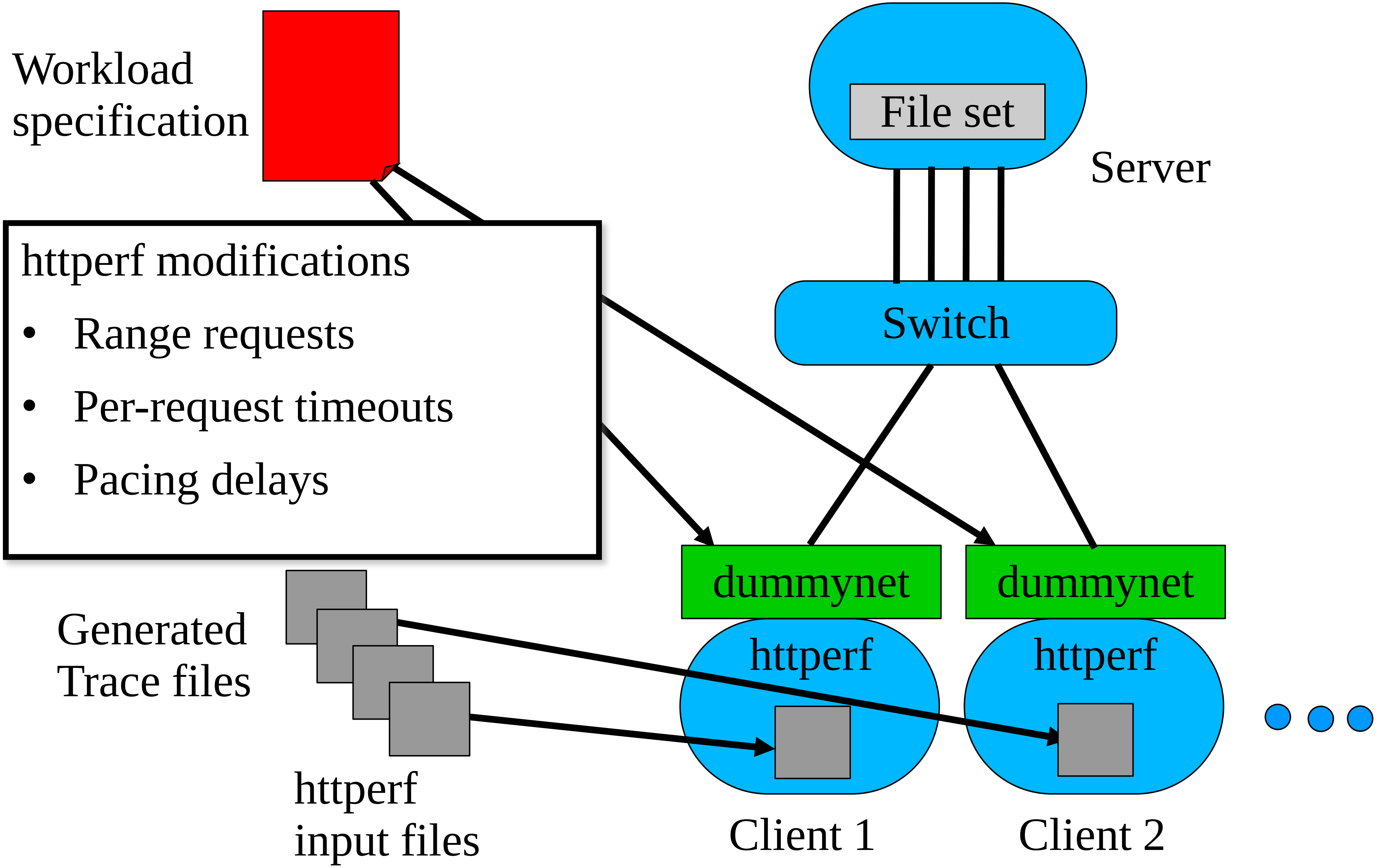
Client 1

Client 2

Running Experiments



Running Experiments



Workload specification

httperf modifications

- Range requests
- Per-request timeouts
- Pacing delays

Generated Trace files

httperf input files

File set

Server

Switch

dummynet

dummynet

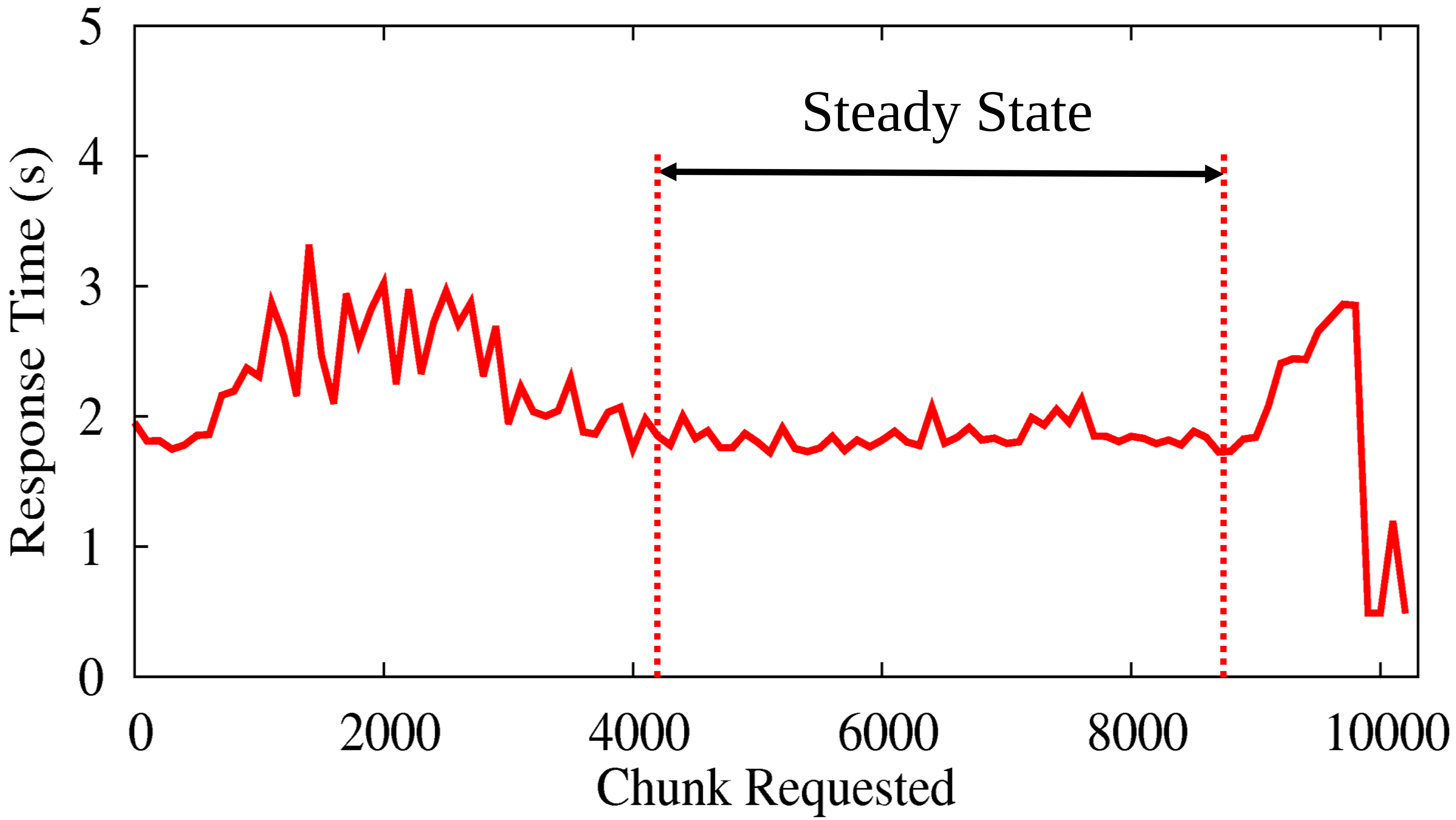
httperf

httperf

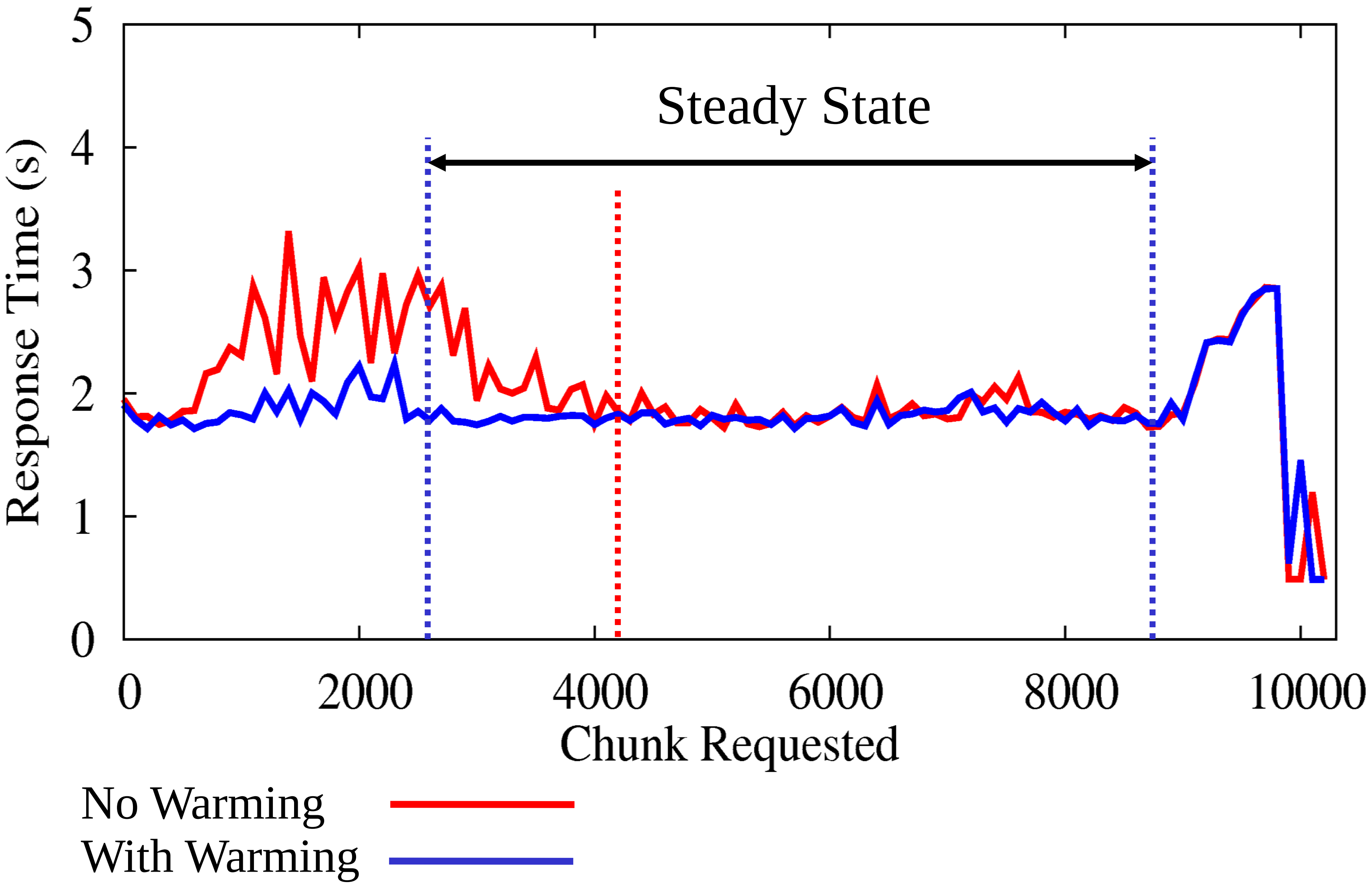
Client 1

Client 2

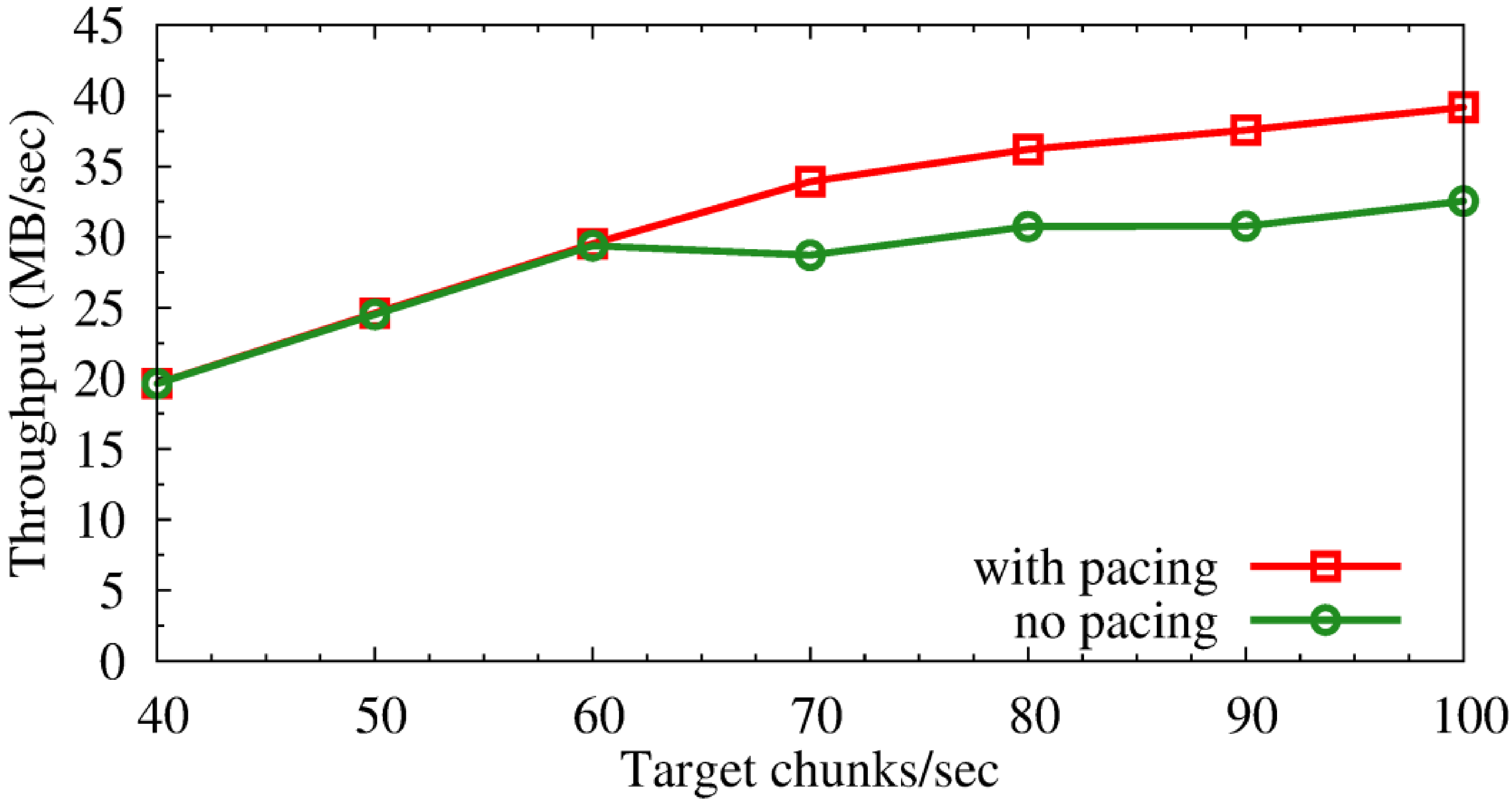
Experiment Progress



Experiment Progress



Effect of Pacing



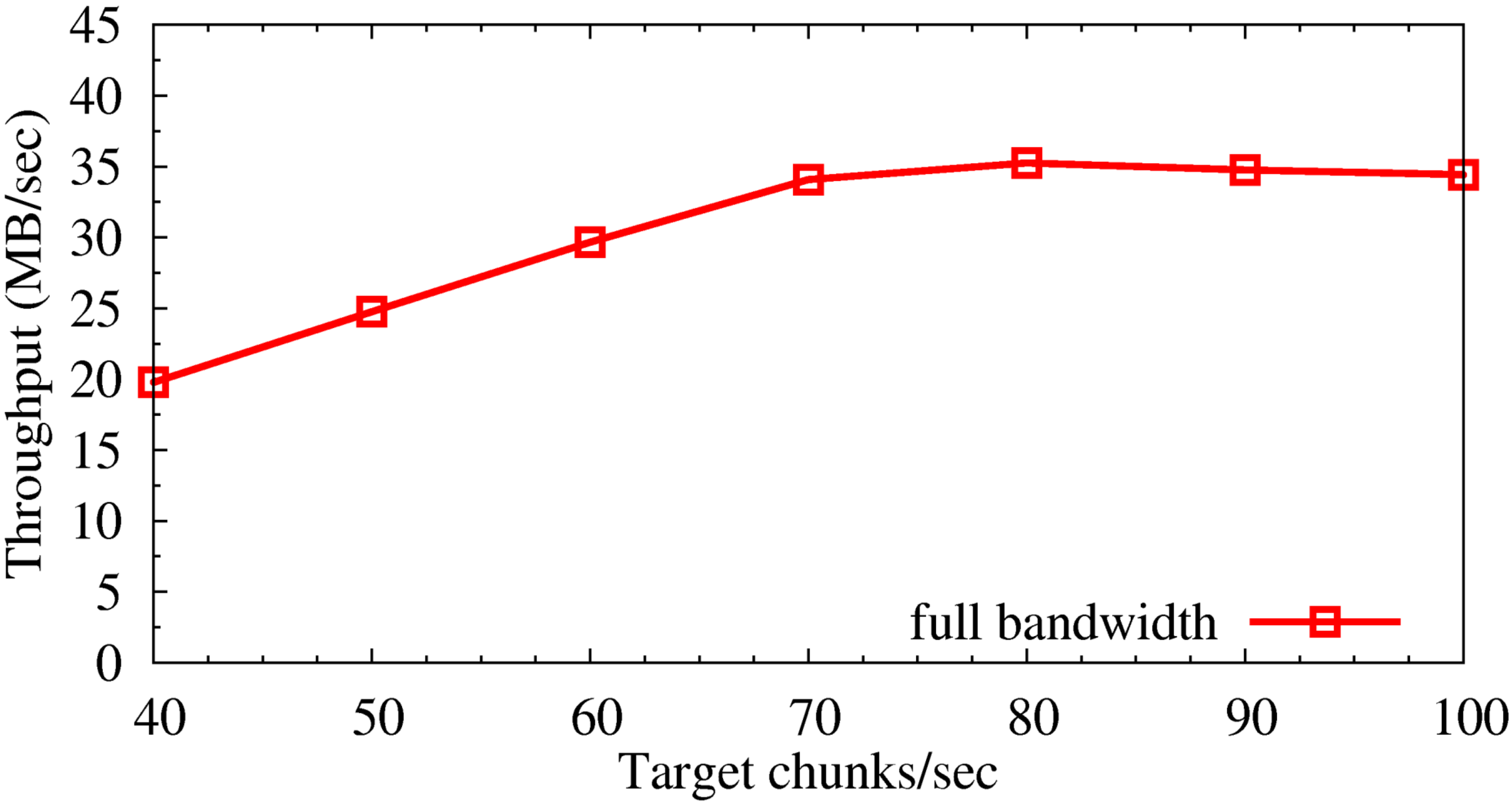
throughput with 0.5 MB chunks

Client Network Limiting

- Lab environment not realistic
  - Different devices and different network speeds
  - Not lab network speeds (e.g. 1 Gbps)

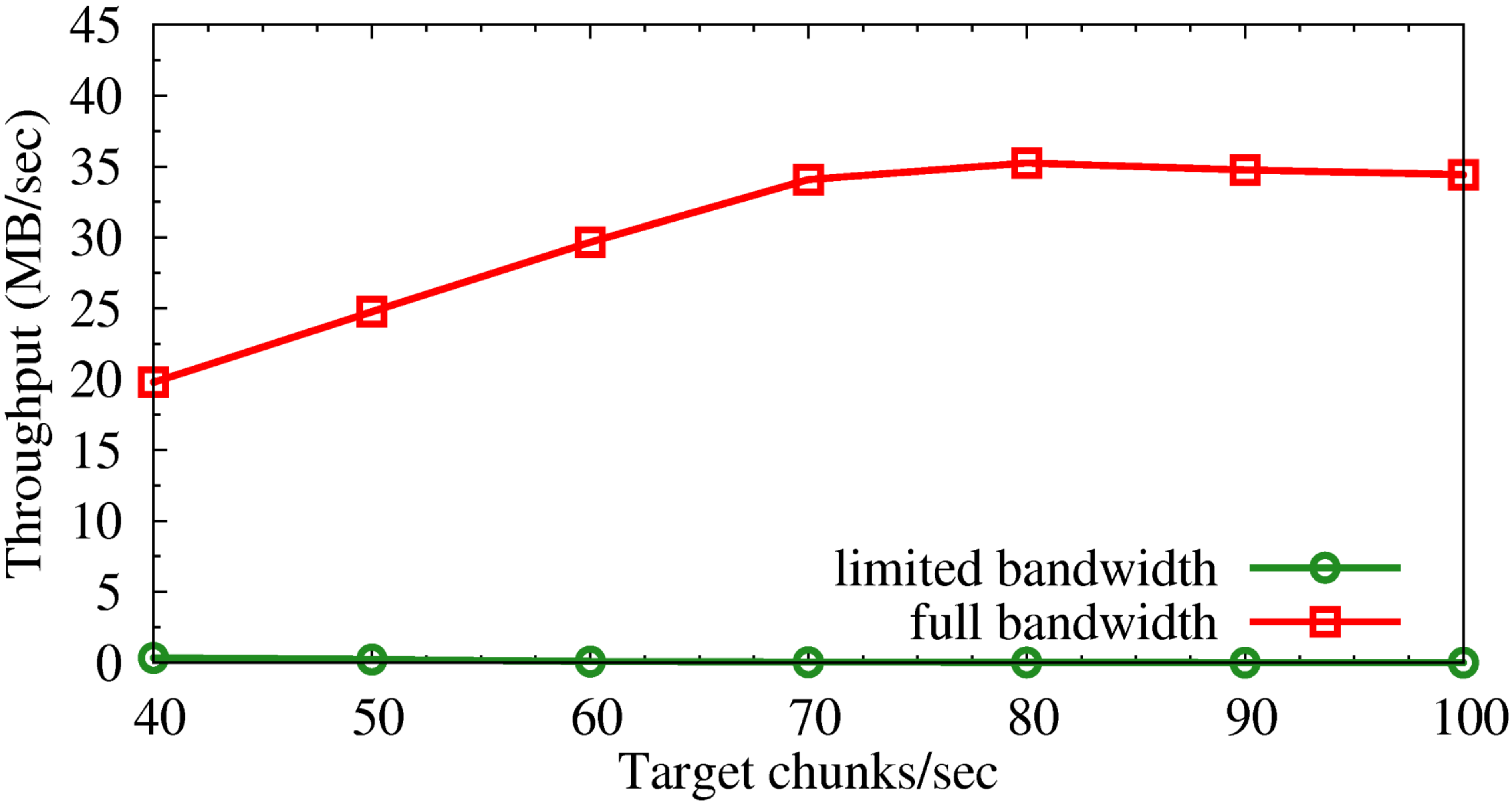
- Preliminary tests: poor disk throughput
- Simple experiment: Service videos one at a time
  - Expected to improve disk throughput

Client Network Limiting



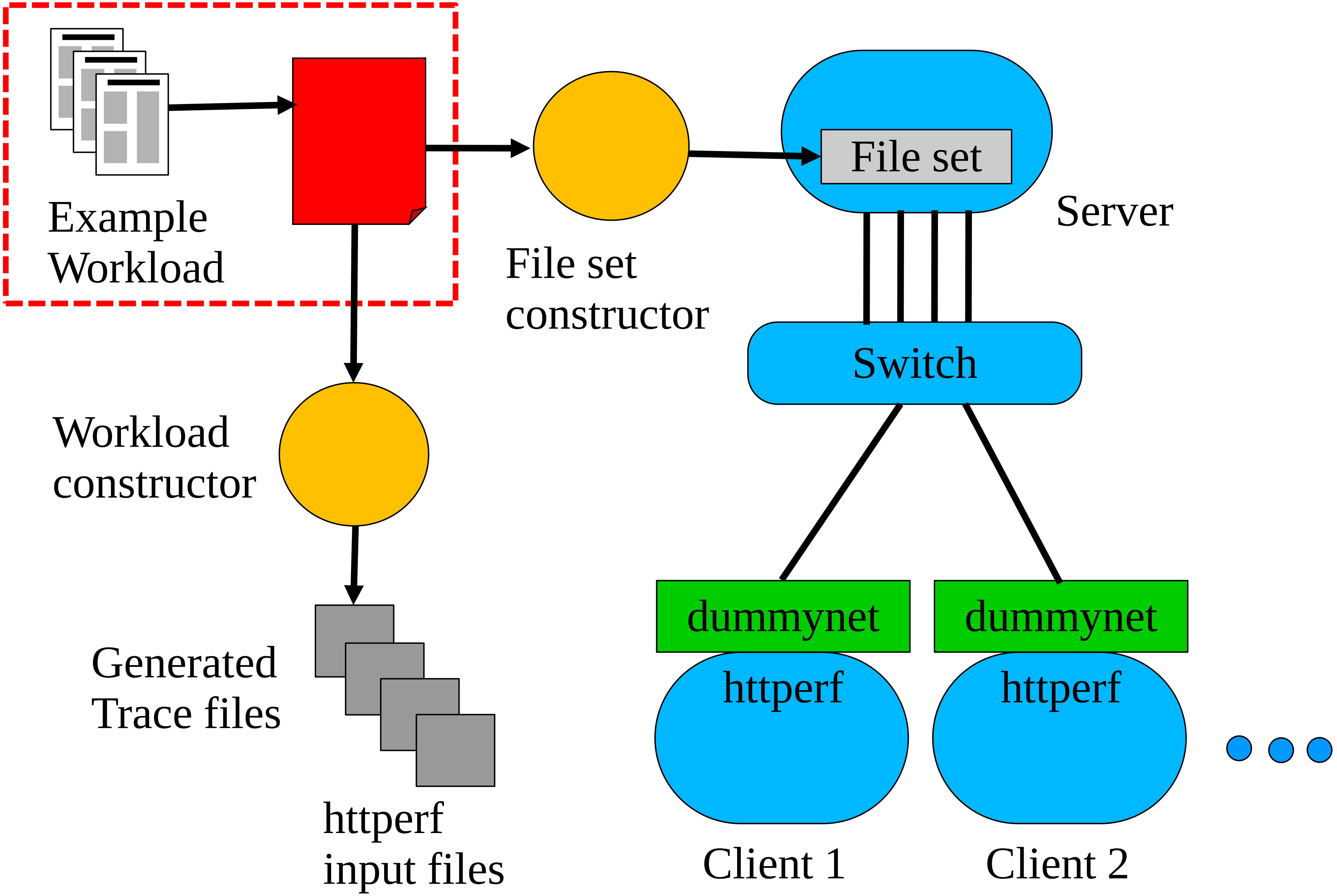
Single-connection throughput with 0.5 MB chunks

Client Network Limiting



Single-connection throughput with 0.5 MB chunks

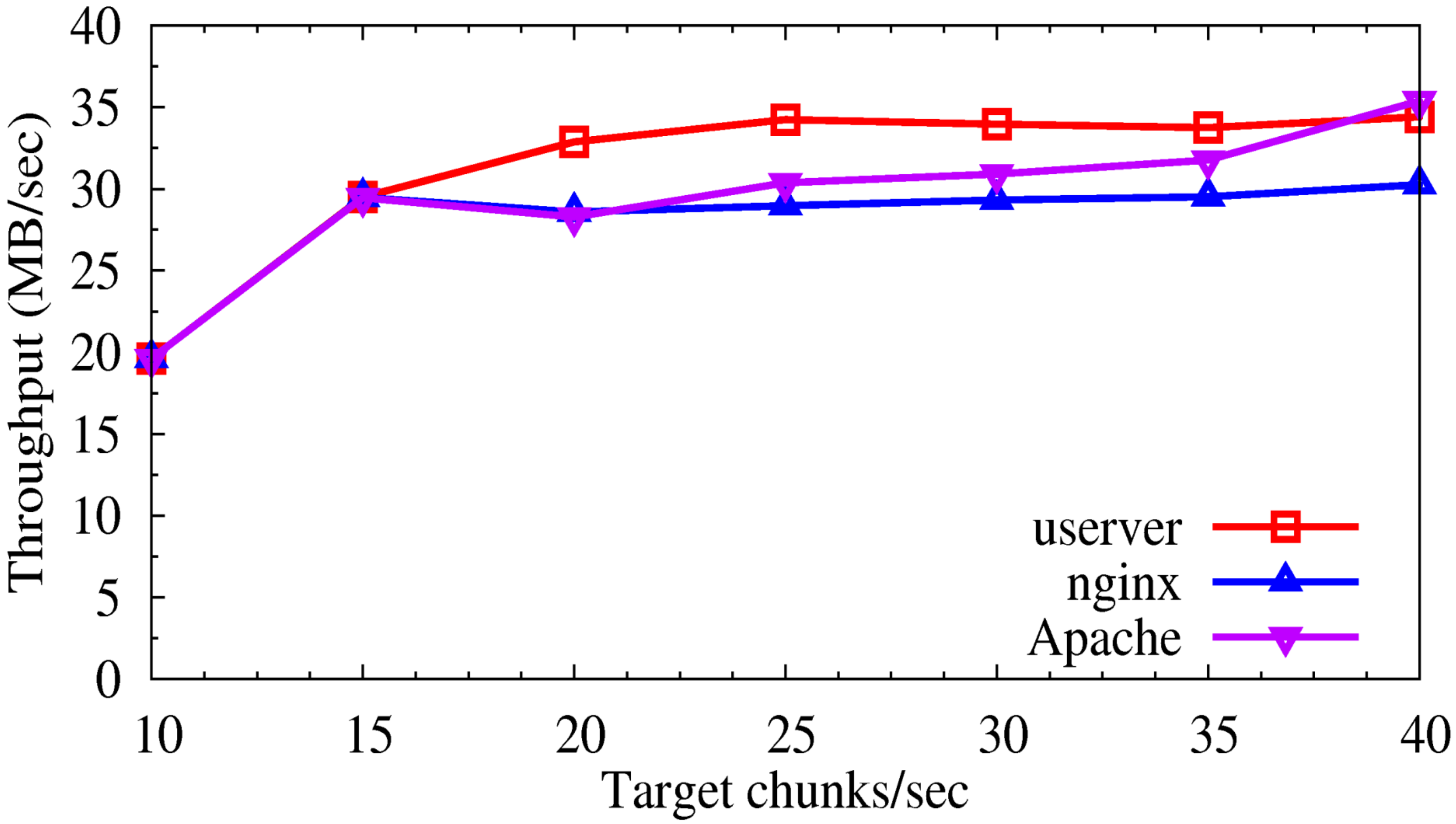
Example Workload



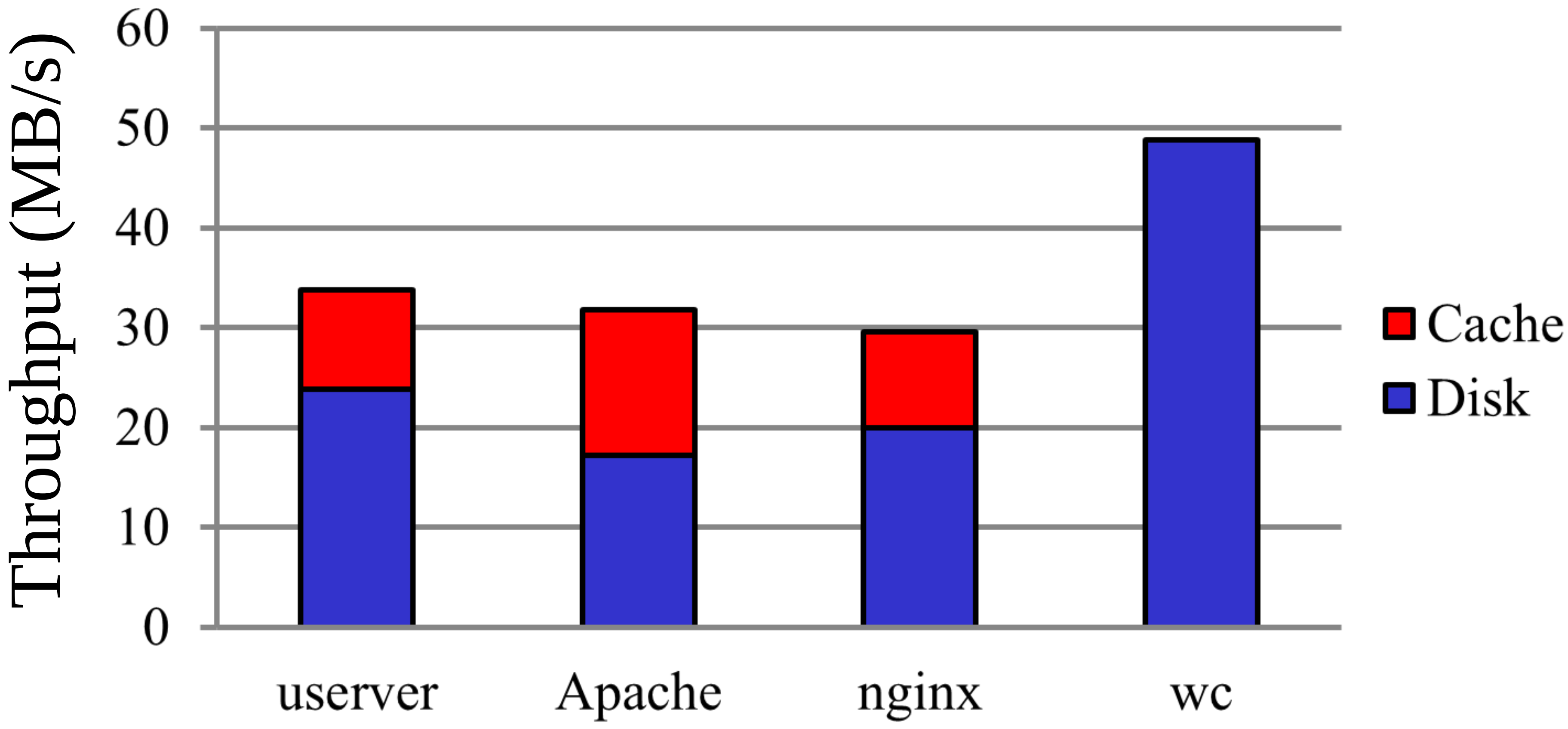
Example Workload

- Video session characteristics [Finamore et al, IMC ’11]
    - Video popularity and duration like YouTube
    - Viewing length distribution like YouTube
- Network Characteristics
    - Bandwidth 10 Mbps, 3.5 Mbps, and 1 Mbps [Akamai]
    - One-way delay 50 ms [N.A. coast-to-coast]
- Server File Set Characteristics
    - Chunks size 0.5 & 2 MB [10 & 40 second chunks]

Throughput with 2 MB chunks



Web Server Throughput

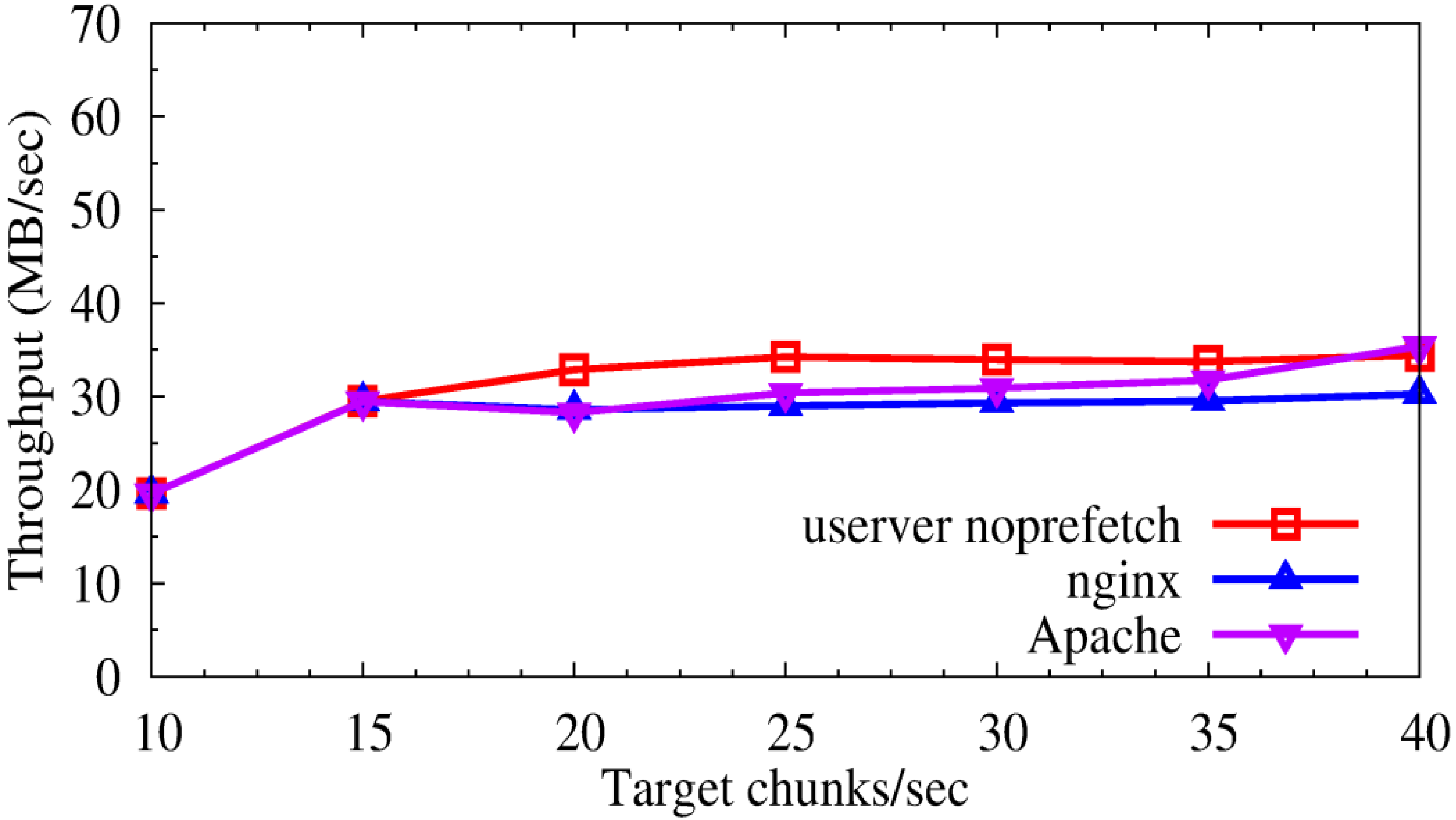


Throughput at 35 chunks/sec with 2 MB chunks

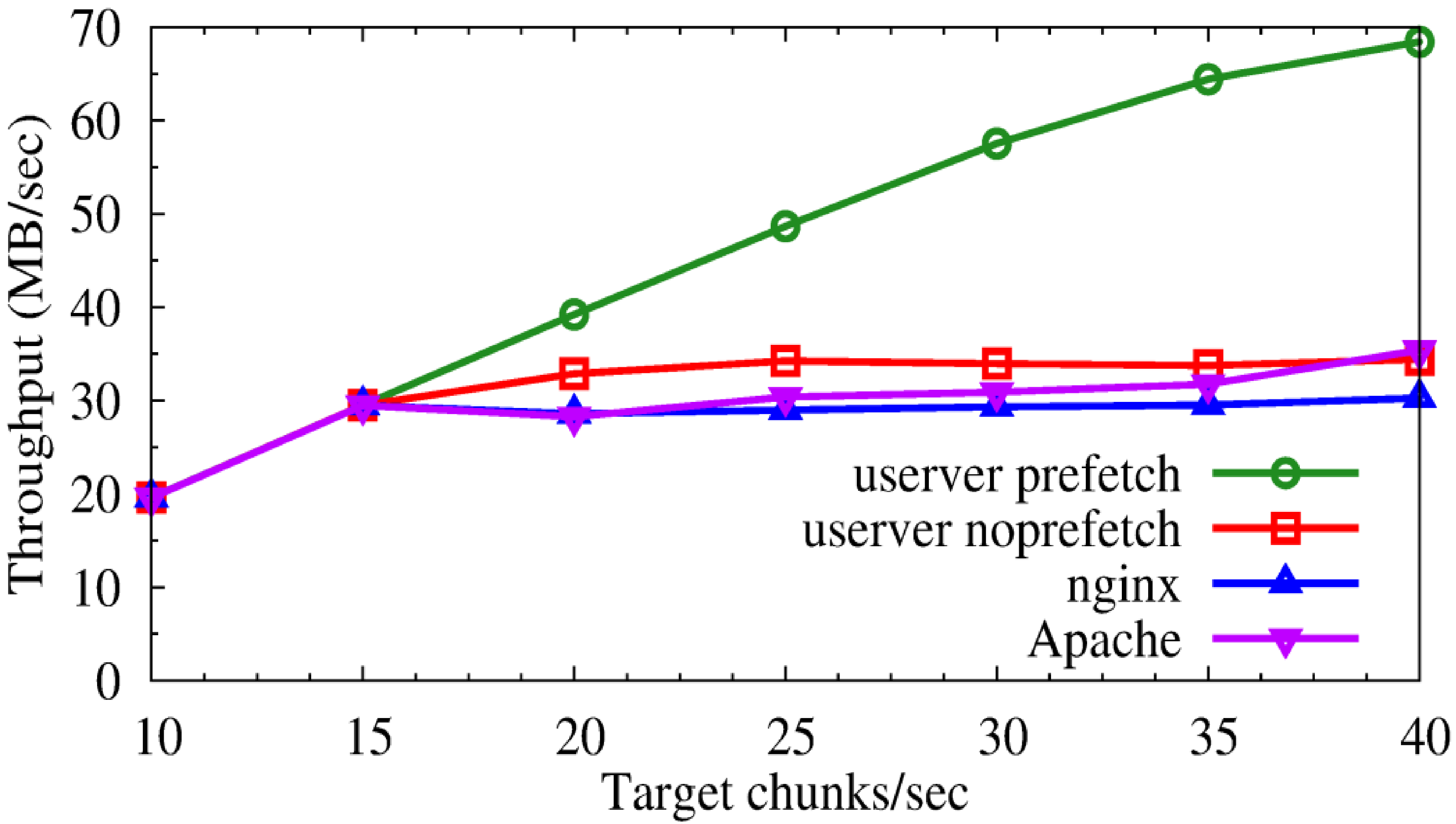
Sequential Prefetching with userver

- Problem:
  - userver uses multiple threads to service requests
  - FreeBSD interleaves concurrent read requests (fairness)
- Ideas:
  - Sequentialize disk access (file/chunk at a time)
  - Agressive application prefetching (entire chunk)

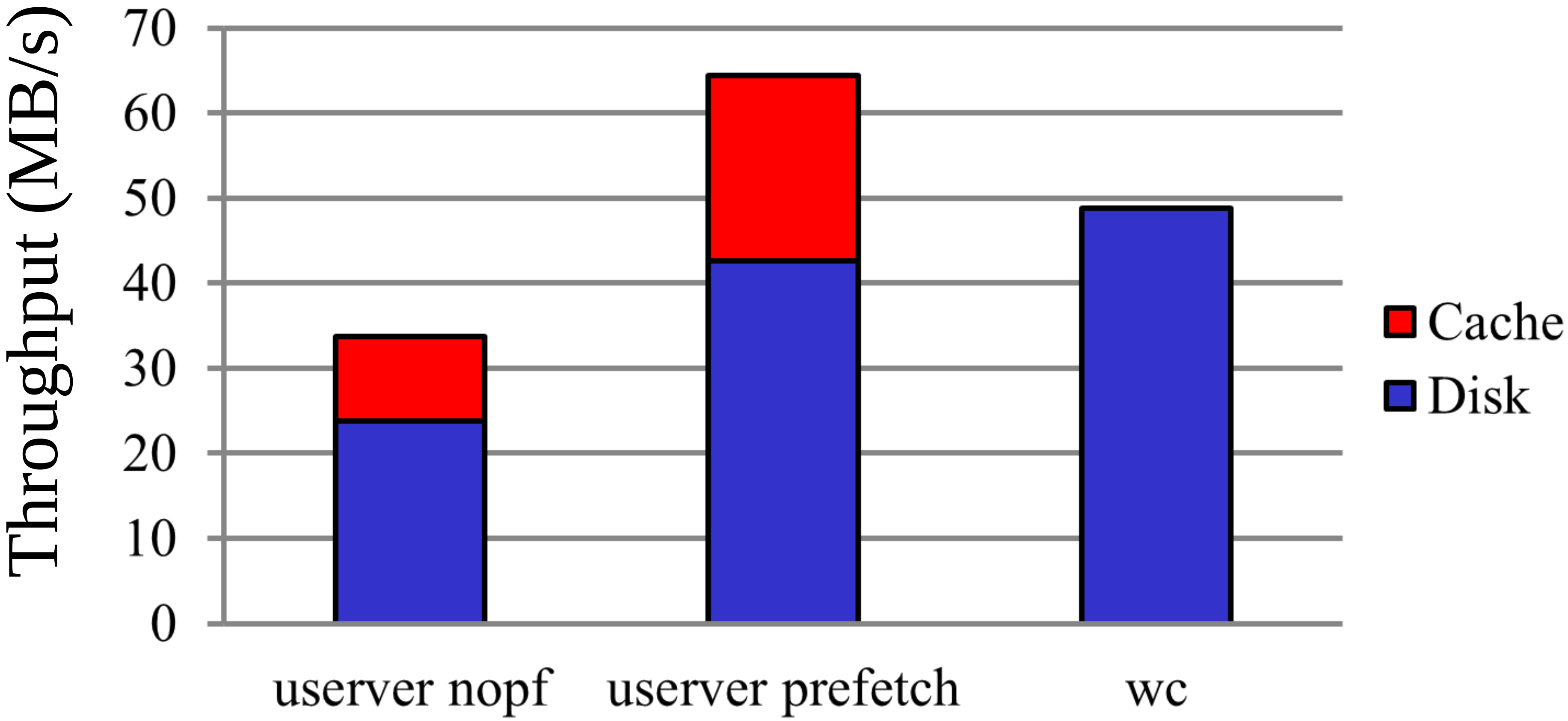
Throughput with 2 MB chunks



Throughput with 2 MB chunks



Improved Disk and Total Throughput



Throughput at 35 chunks/sec with 2 MB chunks

Summary and Conclusions

- Workload Methodology
  - Flexible, representative, practical, useful
- Demonstrate:
  - Client pacing affects results
  - Must emulate client network speeds
- Web servers can be improved
- Study HTTP ecosystem

`cs.uwaterloo.ca/~brecht/papers/systor-2012`

Future Work

- To chunk or not to chunk [Our work, NOSSDAV ’12]
- Sensitivity analysis
- More server improvements
- Library to use with Apache and nginx

`cs.uwaterloo.ca/~brecht/papers/systor-2012`

END