University of North Carolina at Chapel Hill

Tuning RED for Web Traffic
Red considered harmful?

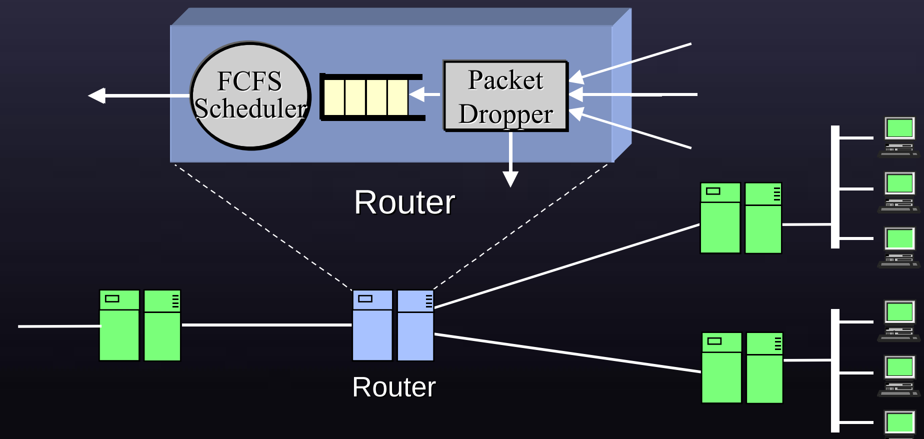
Kevin Jeffay
Don Smith
Mikkel Christiansen
David Ott

March 2000

http://www.cs.unc.edu/Reseach/dirt

Tuning RED for Web Traffic
Research context

- The IETF is strongly advocating deployment of *random early detection* (RED) active queue management in routers



Tuning RED for Web Traffic
Research context

- The IETF is strongly advocating deployment of *random early detection* (RED) active queue management in routers

  "All available empirical evidence shows that the deployment of active queue management mechanisms in the Internet would have substantial performance benefits. There are seemingly no disadvantages to using the RED algorithm, and numerous advantages. Consequently, we believe that RED active queue management algorithm should be widely deployed."

- Measurement studies have shown that 60-80% of traffic in the Internet is HTTP
- How is HTTP performance effected by RED and can RED be tuned to optimize it?

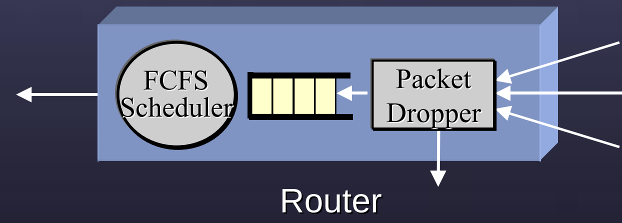
Tuning RED for Web Traffic
Overview

- We've conducted an empirical evaluation of the effect of RED on the performance of HTTP request/response transactions
- We conclude:
  - RED provides no advantage over FIFO for offered loads up to 90% of link capacity
  - Above 90% RED can be tuned to provide better performance, however,
    - » doing so is difficult & error prone
    - » "better" is subjective
    - » response times and link utilization are inversely proportional
- Do we really want RED?

Tuning RED for Web Traffic
Outline

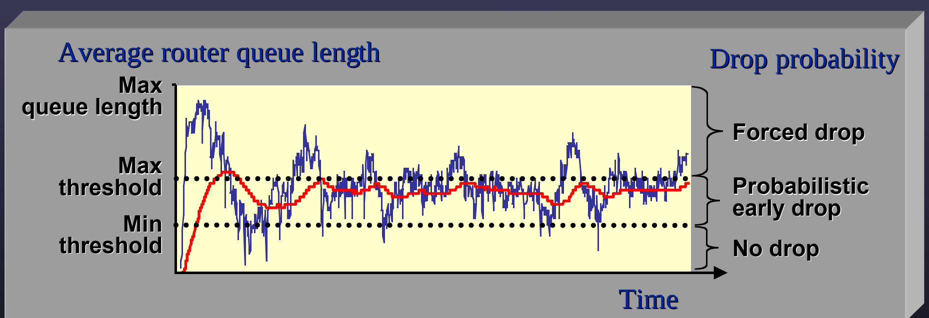
- RED active queue management
- What's known about tuning RED
- Experimental methodology
  - HTTP traffic model
  - Live simulation facility
  - Traffic generation method
- Experimental results
- Conclusions

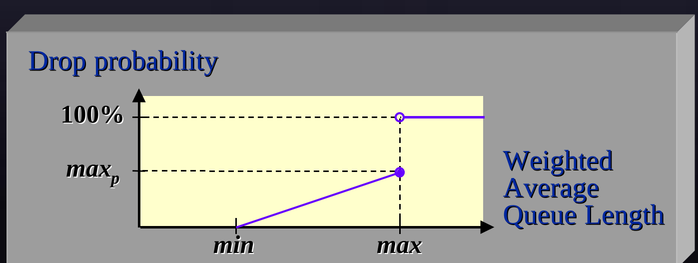
RED Active Queue Management
What's wrong with simple FIFO?



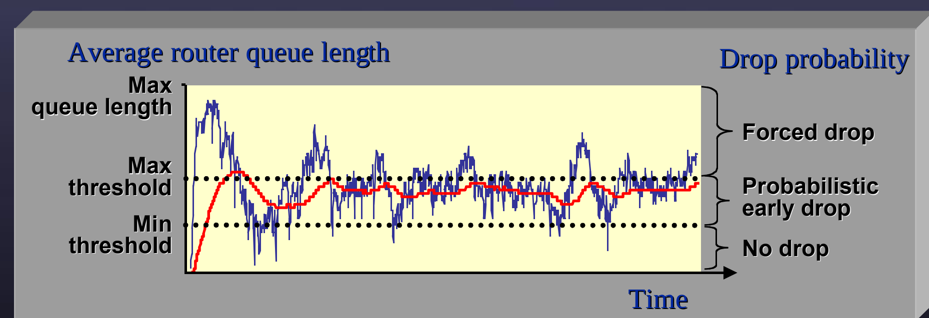
- FIFO can result in reduced link utilization & flows being "locked-out"
- Long queues lead to high latency for all flows
- RED leverages the fact that TCP flows respond to packet loss by reducing their transmission rate
  - By dropping packets "early," full queues are avoided

RED Active Queue Management
Algorithm description



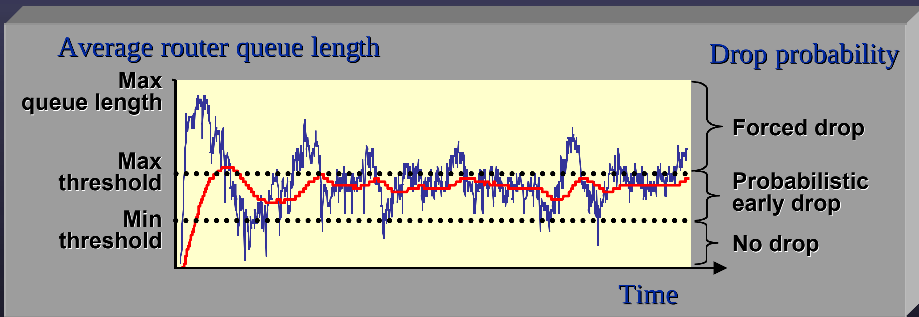


RED Active Queue Management
Algorithm description



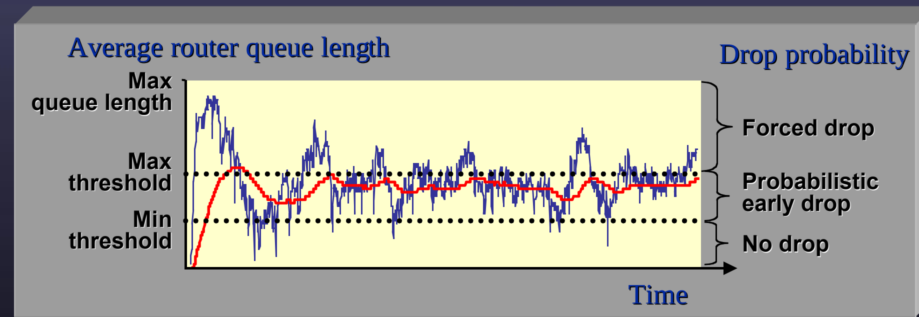
- Random drops avoid lock-out/synchronization effects
  - All flows see the same loss rate
- Early drops avoid full queues
  - Increases effective network utilization ("goodput")
  - Decreases end-to-end latency by decreasing queuing delay

Tuning RED
The RED parameter space



- RED is controlled by 5 parameters
  - *qlen* - The maximum length of the queue
  - w_q - Weighting factor for average queue length computation
  - min_{th} - Minimum queue length for triggering probabilistic drops
  - max_{th} - Queue length threshold for triggering forced drops
  - max_p - The maximum drop probability

Tuning RED
Rules of thumb

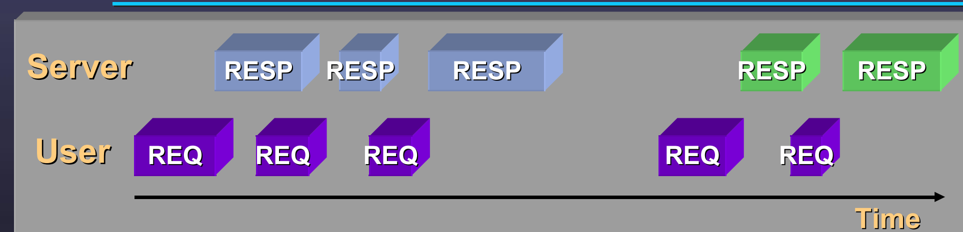


- *qlen* = 2-4 times the *delay-bandwidth* product
- $w_q = 1/2^n$, $n = 9$
- $min_{th} = 5$
- $max_{th} = 3 \times min_{th}$
- $max_p = 10\%$

Tuning RED for Web Traffic
Outline

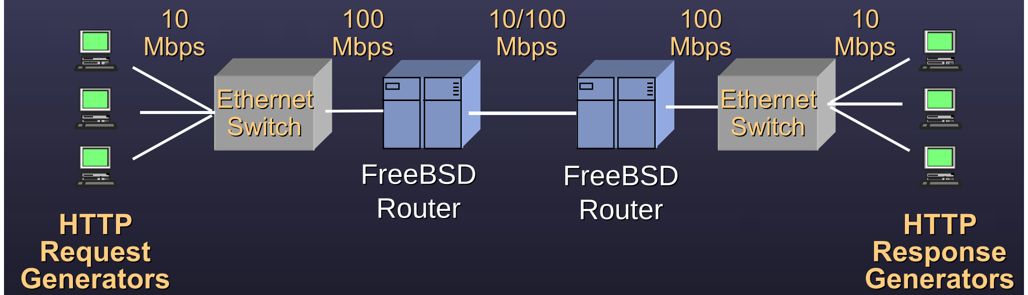
- RED active queue management
- What's known about tuning RED
- Experimental methodology
  - HTTP traffic model
  - Live simulation facility
  - Traffic generation method
- Experimental results
- Conclusions

Experimental Methodology
HTTP traffic generation



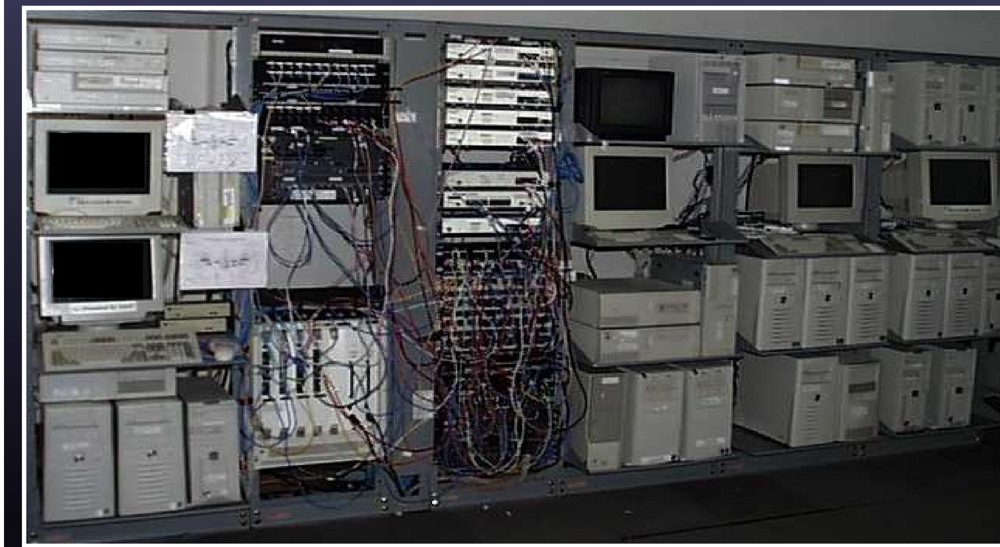
- We generate HTTP traffic using the Mah document model and his empirical distributions of parameters
- Primary random variables
  - Request sizes
  - Reply sizes
  - Number of embedded images/page
  - User inter-document-request think time
  - Consecutive documents per server

Tuning RED for Web Traffic
Experimental Methodology

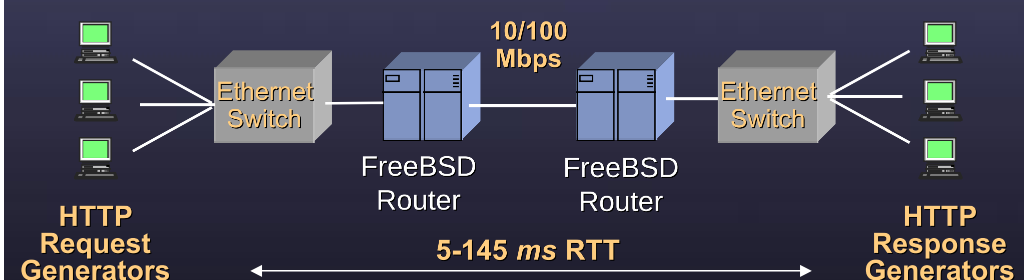


- Evaluate RED through "live simulation"
  - Simulate a large collection of users browsing the web from a number of locations distributed across the USA

Tuning RED for Web Traffic
Experimental Methodology



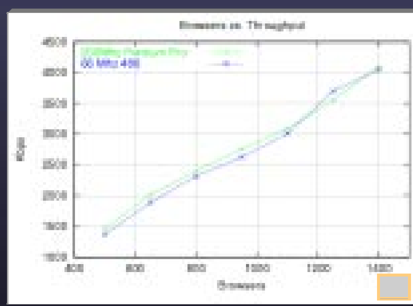
Tuning RED for Web Traffic
Experimental Methodology



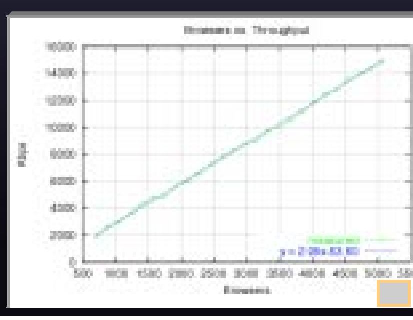
- Evaluate RED through "live simulation"
  - Run RED only on the path from servers to browsers
- Simulation parameters
  - Number of simulated users/browsers
  - RTT between a browser/server pair

Experimental Methodology
100 Mbps calibration experiments

- How many browsing users can a request generator emulate?
  - Need to ensure end-systems are not the bottleneck

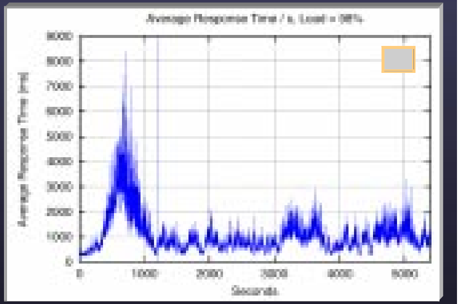


- Is offered load on an un-constrained (100 Mbps) network linear in the number of users?
  - We'll study RED & FIFO at 50, 70, 80, 90, 98, and 110% link utilizations

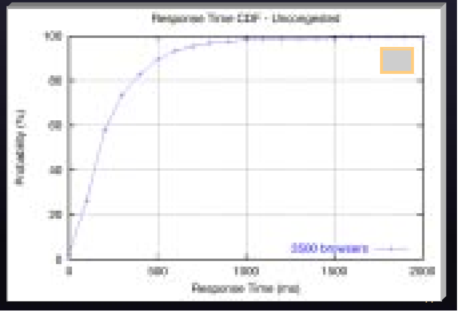


Experimental Methodology
100 Mbps calibration experiments

- Each experiment runs for 90 minutes
  - The results from the initial 20 minutes are discarded

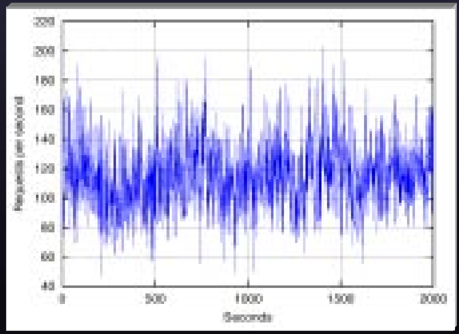


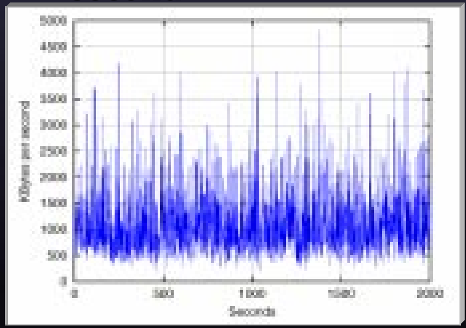
- Sample result:
  - Response time distribution for 3,500 users
  - 90% of requests complete in 500 *ms* or less



Experimental Methodology
100 Mbps calibration experiments

- Generated traffic is suitably bursty
- For 3,500 simulated users (11 Mbps)
  - Requests per second
  - Bytes requested per second





Experimental Methodology
Experimental plan

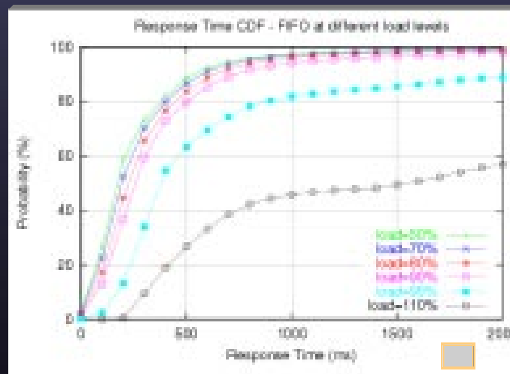
- First determine “best” HTTP request/response response time distribution under FIFO queuing
  - Need to determine optimal FIFO queue length
- Next, determine best RED parameter settings as a function of offered load
- Compare all against performance on the unconstrained (100 Mbps) network

Tuning RED for Web Traffic
Outline

- RED active queue management
- What’s known about tuning RED
- Experimental methodology
  - HTTP traffic model
  - Live simulation facility
  - Traffic generation method
- Experimental results
- Conclusions

Experimental Results
FIFO queue length determination

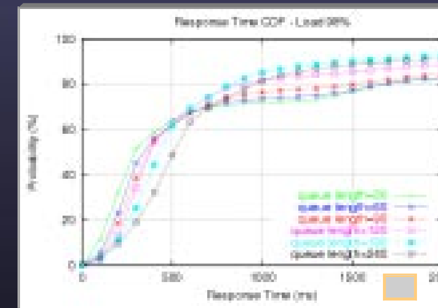
- Queue length not a significant factor below 90% of link capacity
- Above 90% of capacity, response time degrades quickly
- (We'll consider offered loads of 80, 90, 98, and 110% of link capacity)

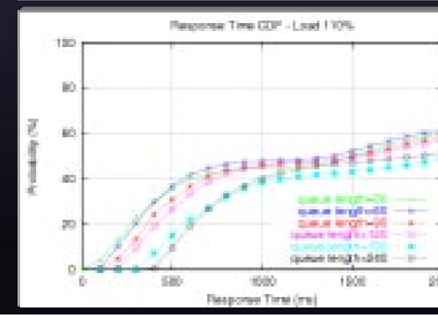


q_{len} = 120 packets

Experimental Results
FIFO queue length determination

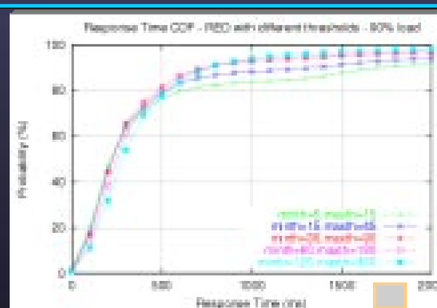
- Queue lengths from 30-240 packets were considered
- We declare a q_{len} = 120 to be the "winner"
  - 120 = 1.25 x *bandwidth* x *delay*
- Larger queues provide slightly higher link utilization and lower drop rates
- Trade-off between optimizing for shorter responses *v.* longer responses

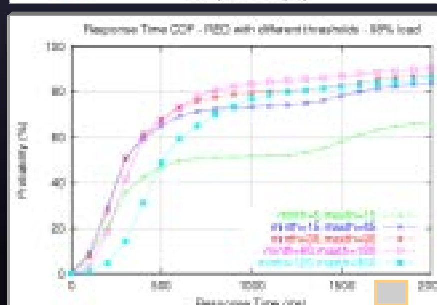




Experimental Results
RED parameter determination

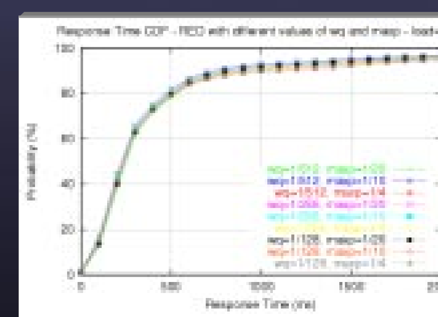
- Ignore the effects of queue length
  - Set *qlen* to infinity (480)
- Vary min_{th} from 5-120
  - Assume the rule-of-thumb $max_{th} = 3 \times min_{th}$
- Best performance results from thresholds in the range (30, 90) - (60, 180)
- min_{th} = 5 gives poor performance
- Same trade-off exists between optimizing for shorter *v.* longer responses





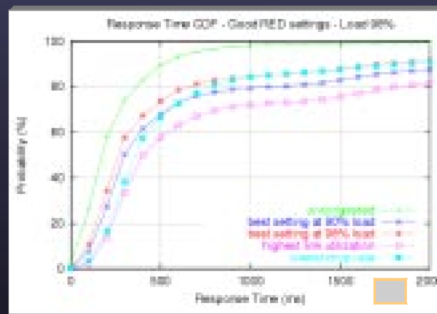
RED Parameter Determination
Changing w_q and max_p

- Combine testing of w_q and max_p
  - The two were determined to be closely related
- Recommended:
  - w_q = 1/512, max_p = 1/10
- Results:
  - Impact of changing w_q from 1/128 to 1/512 was minimal (1/1024 was quite poor)
  - Settings max_p = 1/4 increased response times
  - No significant difference in performance between max_p = 1/10 or 1/20



RED Parameter Determination
"Good" RED setting

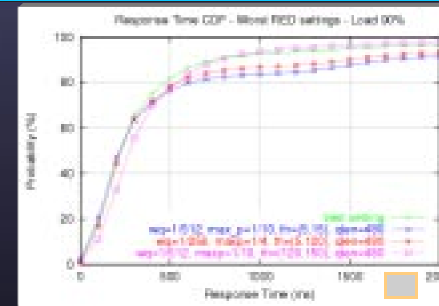
- Tuning for better link utilization has a negative effect on response times
- Tuning for lowest drop rate also has a negative effect on response times

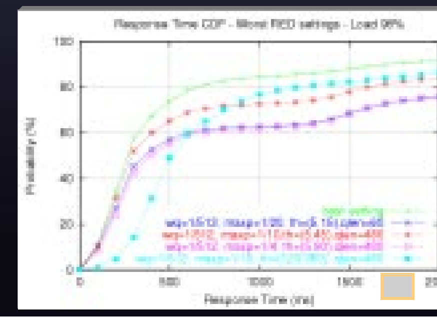


- Settings for the best overall response times at 98% load, differ from our general guidelines for optimal response time setting
  - *thresholds* = (5, 90), w_q = 1/128, max_p = 1/20, q_{len} = 480

RED Parameter Determination
Bad RED setting

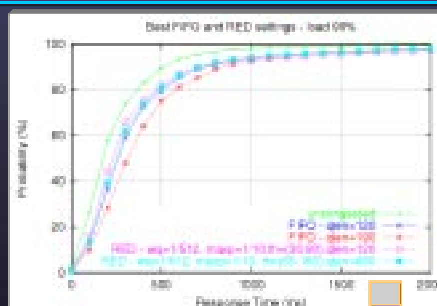
- Worst RED settings can significantly decrease response time performance
- An example is the default setting in the RED distribution for FreeBSD
  - *thresholds* = (5, 15)
    w_q = 1/512
    max_p = 1/20
    q_{len} = 60

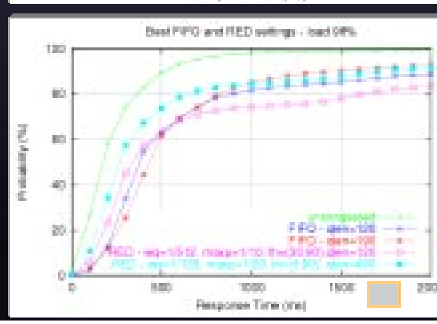




FIFO *v.* RED
Comparison

- FIFO and RED have equal response time performance at 90% load and below
- At 98% load RED can outperform FIFO
- At 110% load RED and FIFO have equal performance





Tuning RED for Web Traffic
Summary and Conclusions

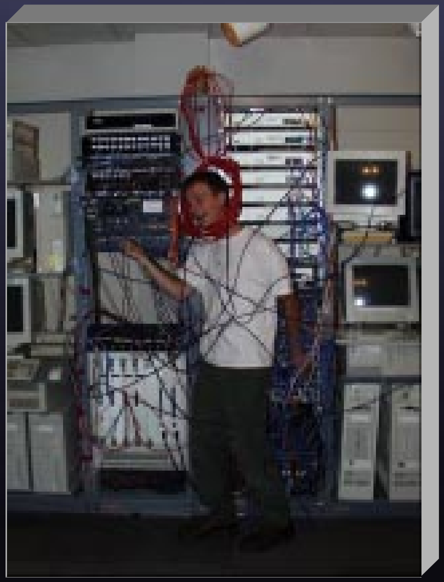
- RED provides no advantage over FIFO for offered loads up to 90% of link capacity
  - The Braden *et al*. performance claim doesn't hold for HTTP response times
- Above 90% RED can be tuned to provide better performance, however, ...
  - Doing so is difficult & error prone
    - » Braden *et al*. "no harm" claim doesn't hold for HTTP response times
  - "Better" is subjective
    - » Response times and link utilization are inversely proportional
- Widespread deployment of RED at present may cause more harm than good

Tuning RED for Web Traffic
Next steps

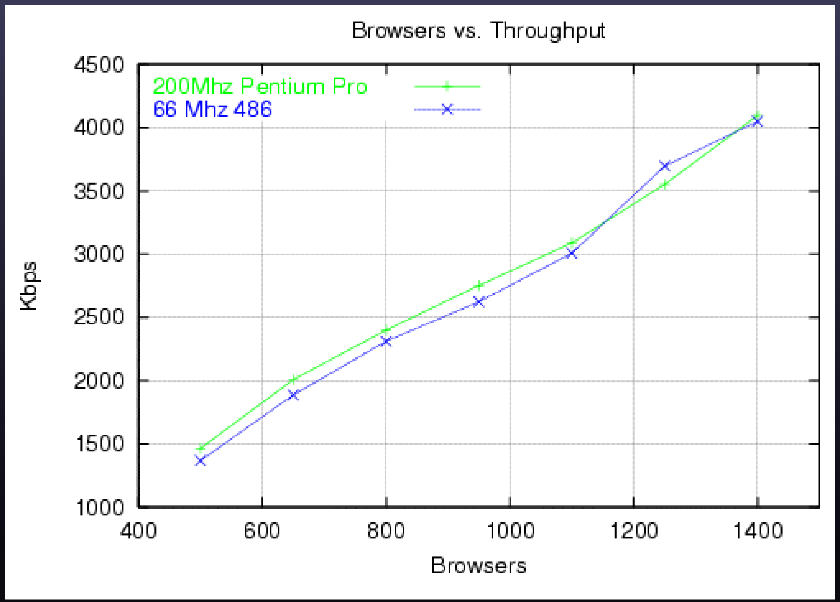
- Redo experiments using a realistic mix of HTTP 1.0/1.1 traffic
  - … with updated parameter distributions
- Redo experiments using a realistic mix of HTTP and other TCP (and UDP) traffic
- Examine the impact of packet-drop RED *v.* ECN RED

Tuned RED for Web Traffic
Research on RED considered harmful!

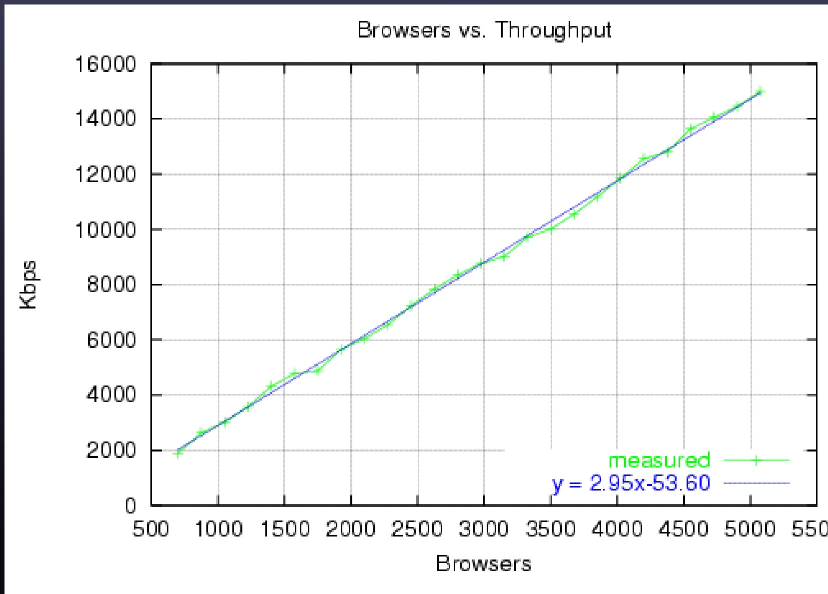
- Live simulation gone awry...



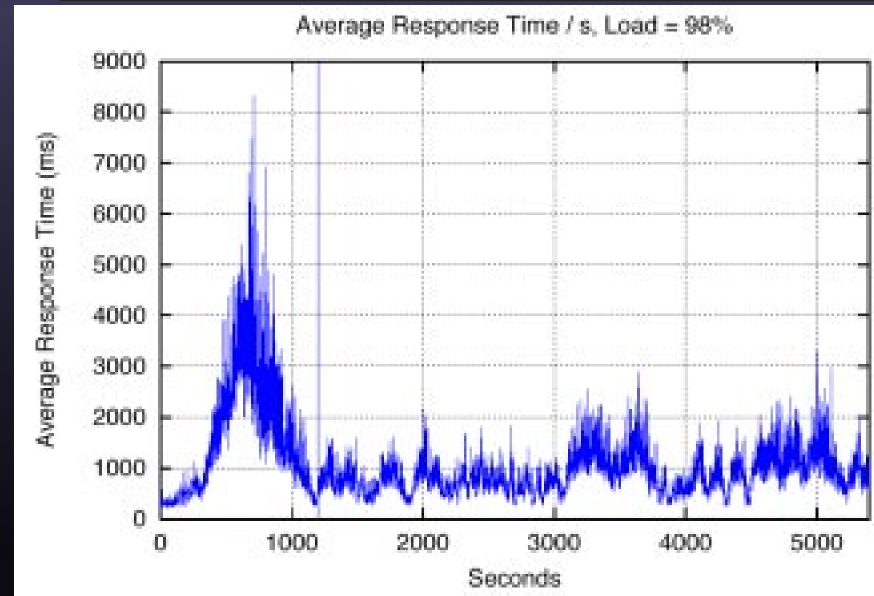
Experimental Methodology
100 Mbps calibration experiments



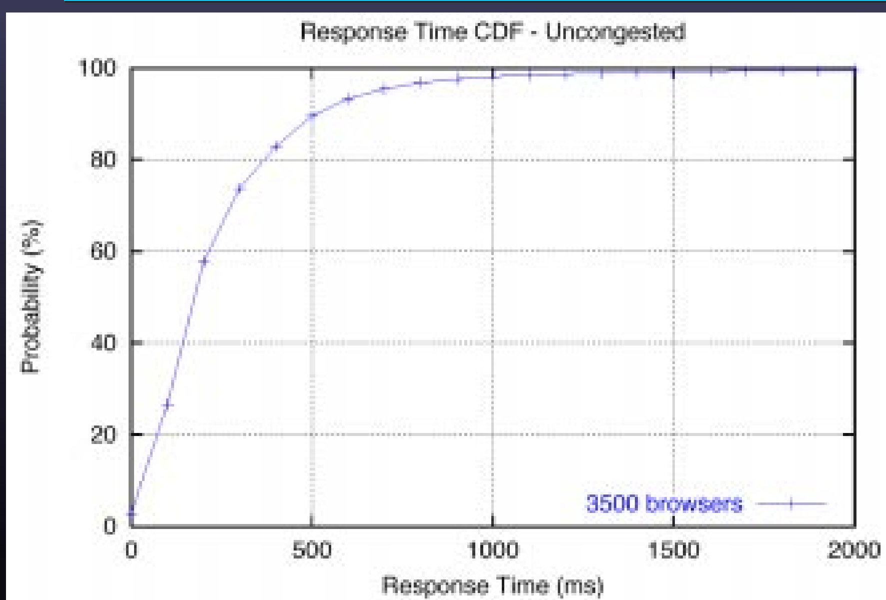
Experimental Methodology
100 Mbps calibration experiments



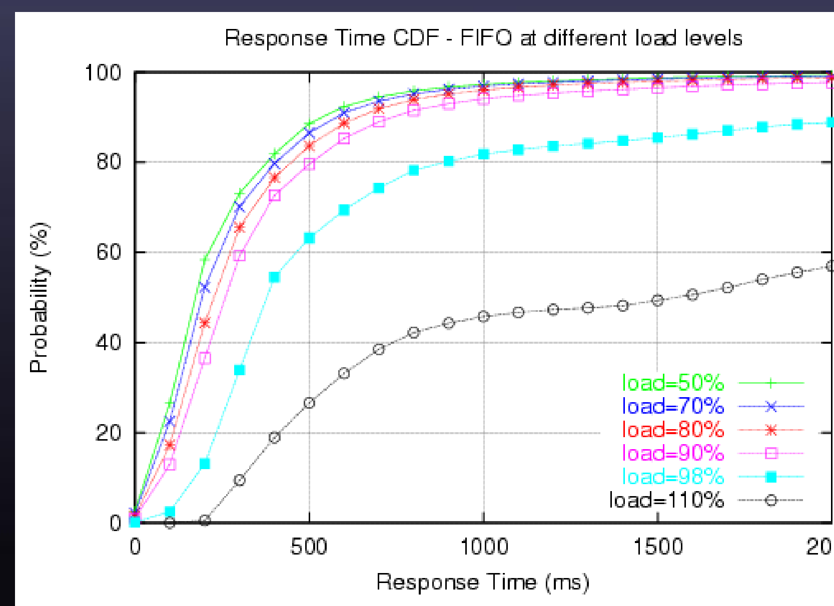
Experimental Methodology
100 Mbps calibration experiments



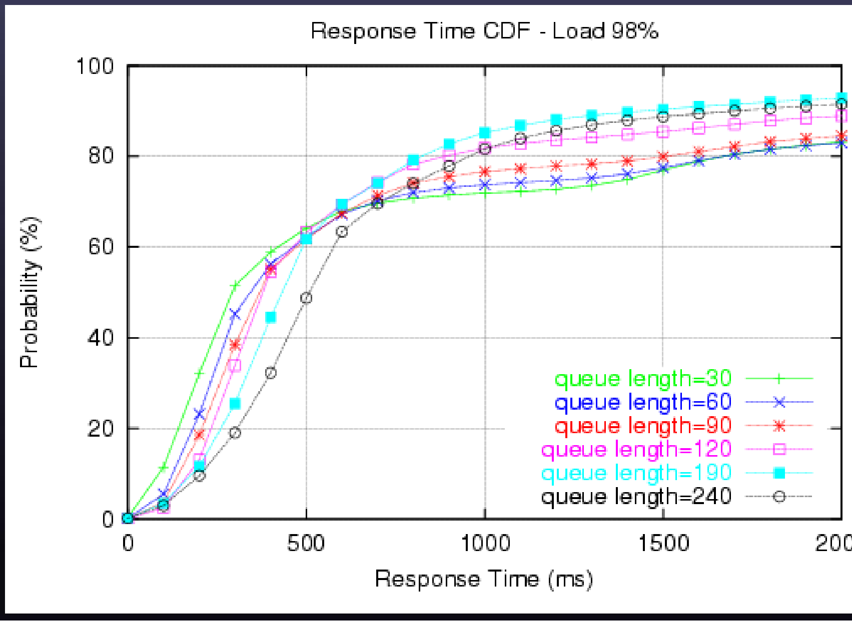
Experimental Methodology
100 Mbps calibration experiments



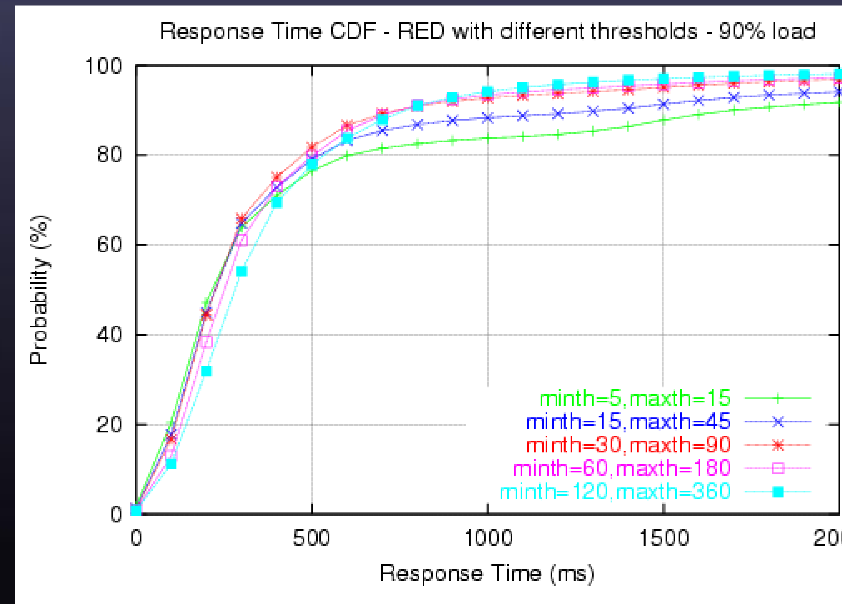
Experimental Results
FIFO queue length determination



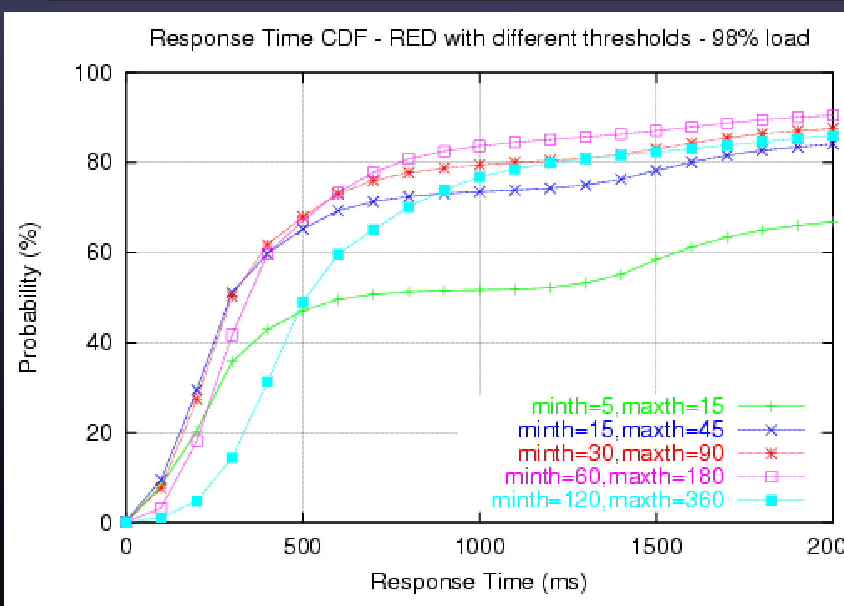
Experimental Results
FIFO queue length determination



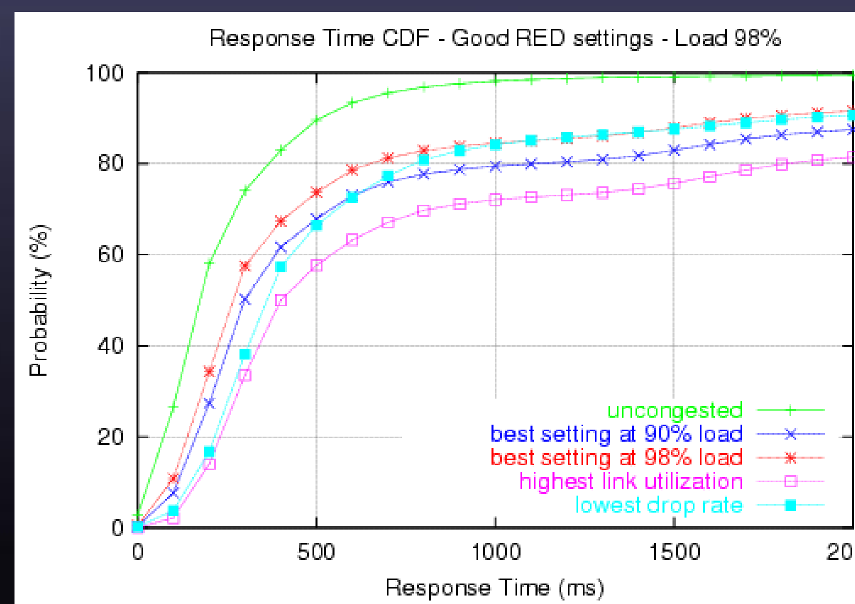
Experimental Results
RED parameter determination



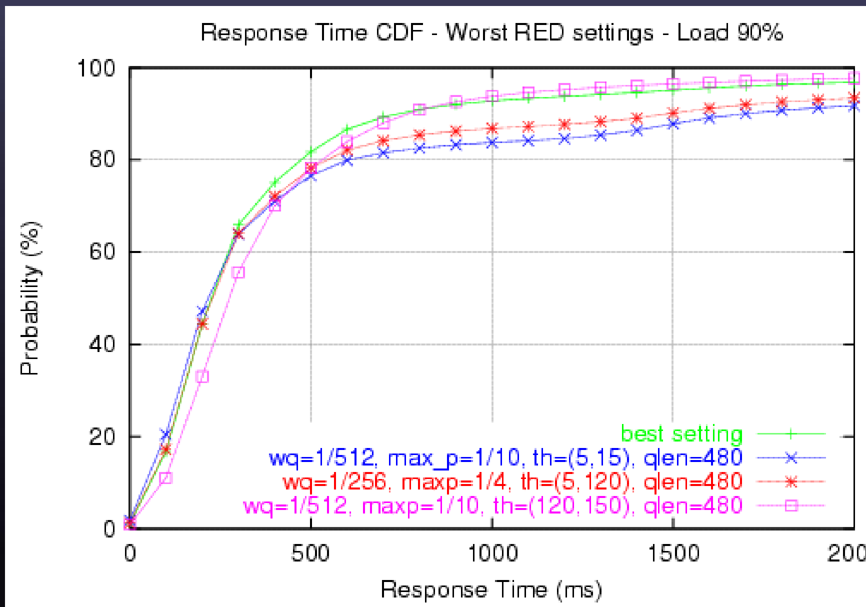
Experimental Results
RED parameter determination



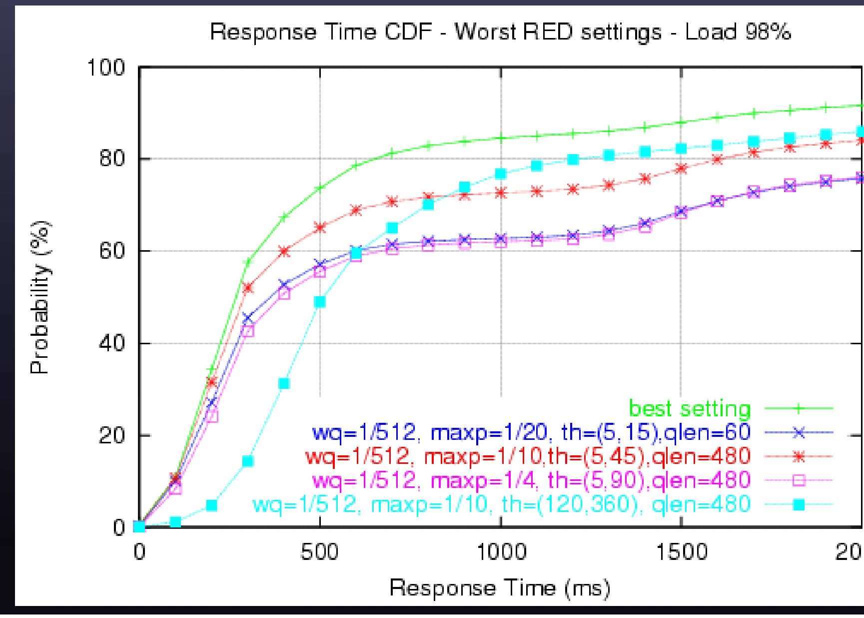
RED Parameter Determination
"Good" RED setting



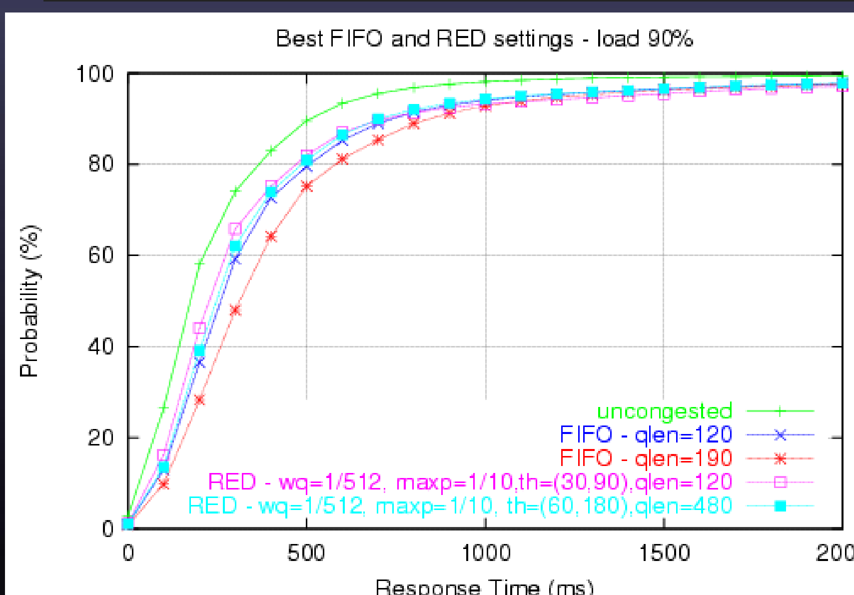
RED Parameter Determination
Bad RED setting



RED Parameter Determination
Bad RED setting



FIFO *v.* RED
Comparison



FIFO *v.* RED
Comparison