Protecting Network Quality of Service Against Denial of Service Attacks

Douglas S. Reeves
Peter Wurman
NC State University

S. Felix Wu
U.C. Davis

Dan Stevenson
Xiaoyong Wu
MCNC

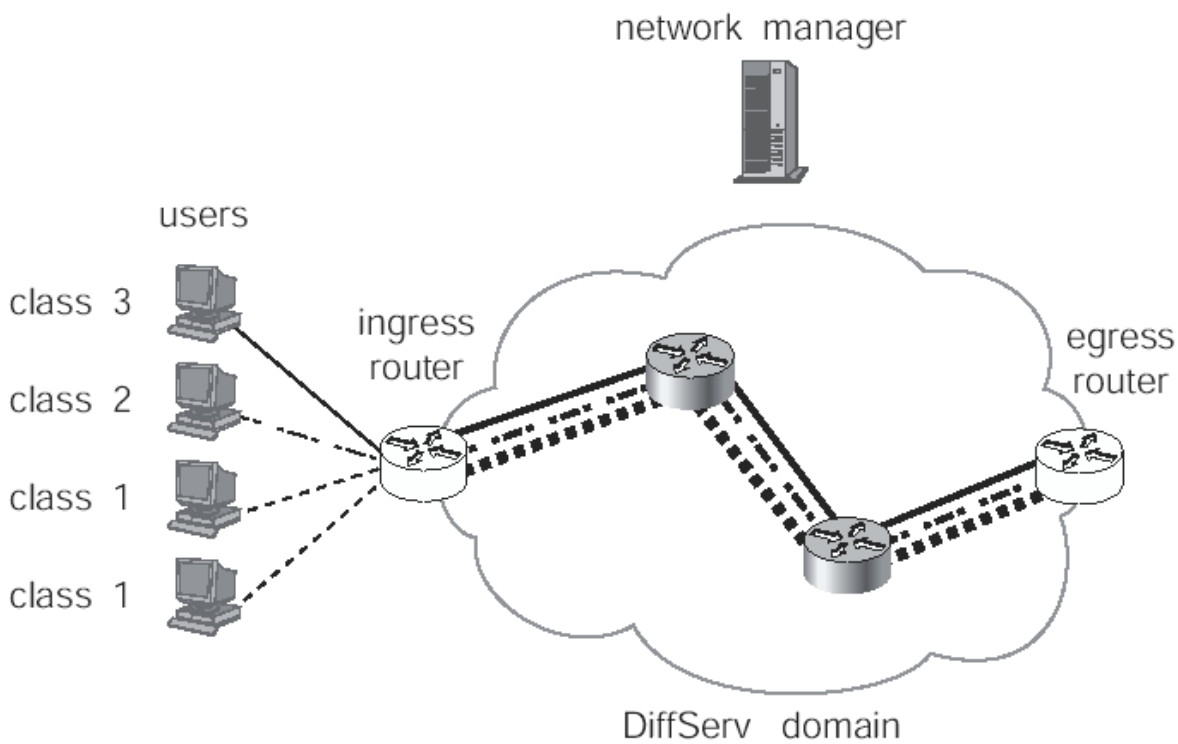
DARPA FTN PI Meeting
January 17, 2002

Scope of the Project

Function → / Protocol ↓	Control Flow		Data Flow		Network Management
	Protect	Detect Attacks	otect	Detect Attacks	
IntServ	RSVP authenti-cation				Pricing for bandwidth allocation Trust-based allocation
DiffServ					
Reliable Delivery					
IPSec					IPSec Policy generation, correctness checking
IP				**Applicat watermarking** **Tr correlation**	

DiffServ Pricing

- DiffServ provides different classes of service to aggregated traffic



DiffServ Pricing (cont'd)

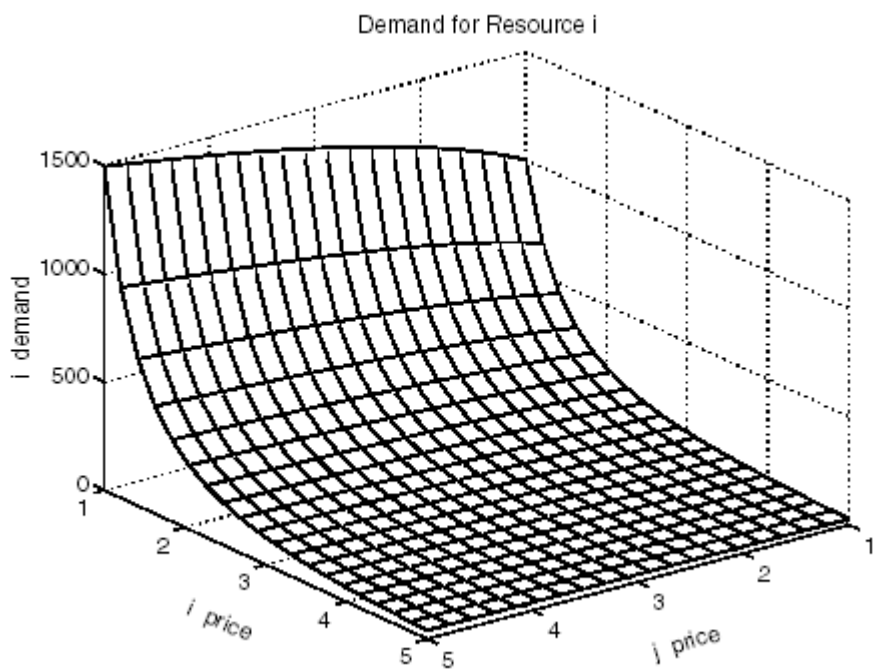
- Two decisions
  - traffic class bandwidth provisioning: long-term, large-scale capacity assignment problem
  - user traffic bandwidth allocation: short-term, small-scale assignment problem
- Pricing is a mechanism to achieve efficient resource utilization, encourage truthfulness, and discourage attacks
  - reactive or predictive pricing?

Optimization

- Assume traffic class demands are predictable (although variable)
- Profit = sum over all times and traffic classes of (revenue - cost)
  - cost is independent of time, revenue is dependent (changing user demands)
- Optimal solution:
  marginal change in revenue =
  marginal change in cost

Model

- Aggregate user demand is known for each time of day
- Each user has minimum acceptable class of service
- Users are willing to accept higher grade of service -- cross-price elasticity



Provisioning

- Set of nonlinear equations for optimal traffic class provisioning

$$
\begin{aligned}
p_{i,t} \cdot x_{i,t} &= \left(\frac{x_{i,t}}{\beta_{i,t} \cdot \prod_{j \in Q, j \neq i} p_{j,t}^{\alpha_{ij,t}}} \right)^{\frac{1}{\alpha_{ii,t}}} \cdot x_{i,t} \\
&= x_{i,t}^{1+\frac{1}{\alpha_{ii,t}}} \cdot \beta_{i,t}^{\frac{-1}{\alpha_{ii,t}}} \cdot \left(\prod_{j \in Q, j \neq i} p_{j,t}^{\alpha_{ij,t}} \right)^{\frac{-1}{\alpha_{ii,t}}} \quad (4)
\end{aligned}
$$

- Solve by gradient method, running time not a big issue

Allocation

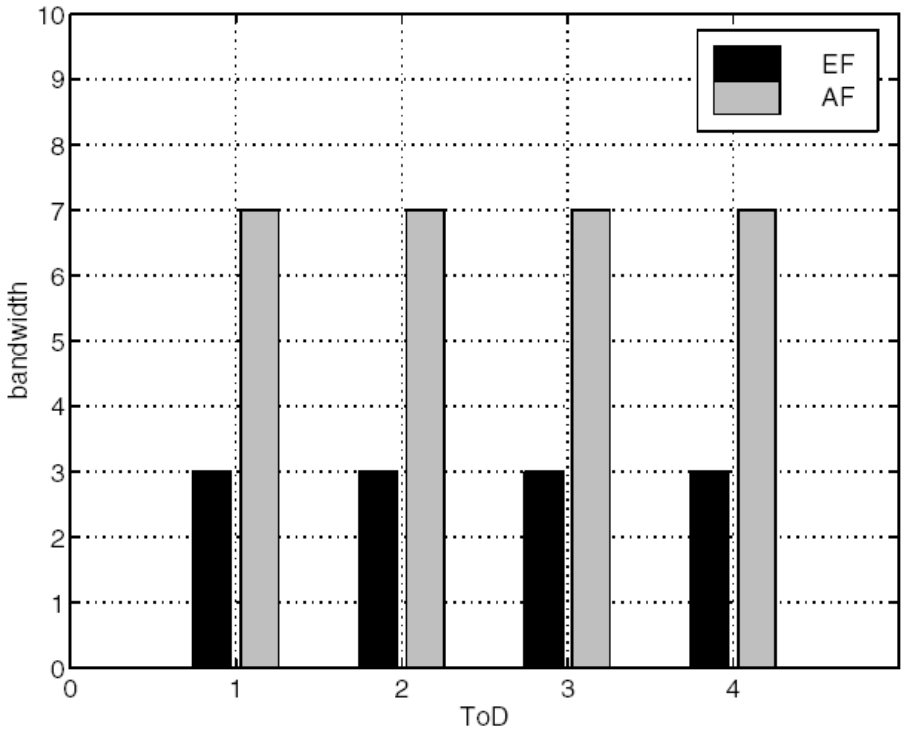
- After provisioning (long time scale), must allocate to users (short time scale)
- Users will "promote" to higher grade of service if insufficient demand for higher class
- Steps
  - #1: determine optimal aggregate allocation (demand)
  - #2: set prices to achieve this predicted demand
  - also non-linear, requires iterative solution techniques
- Price schedule is published (e.g., time-of-day pricing)
- Blocking of user requests occurs if the actual demand exceeds predicted demand

Example of Class Promotion

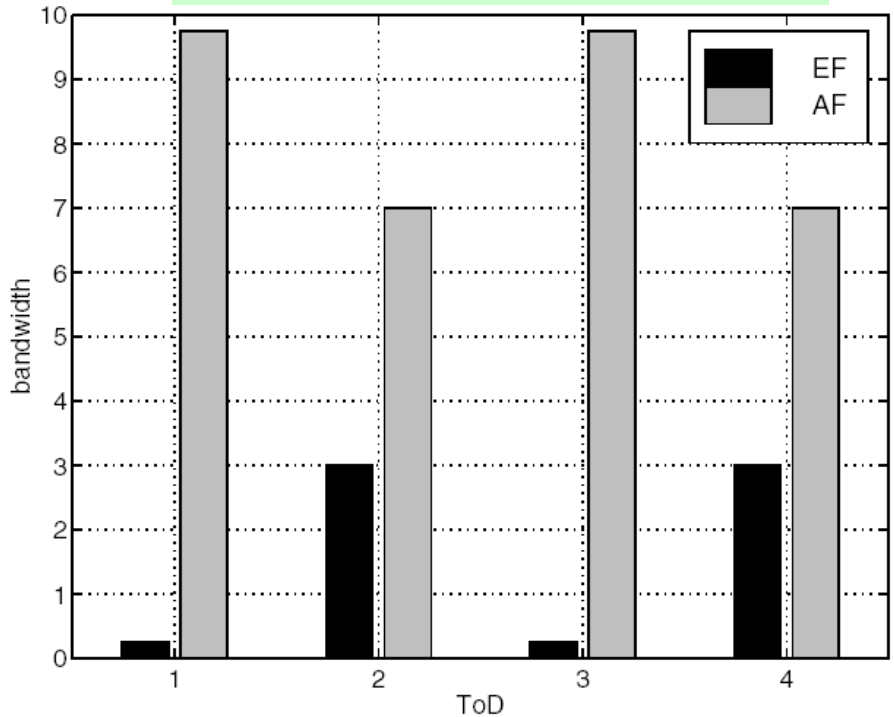
Demands by time of day

User Type	Wealth				Elasticity	
	$w_{i,1}$	$w_{i,2}$	$w_{i,3}$	$w_{i,4}$	α_{ii}	$\alpha_{\mathrm{AF,EF}}$
EF	500	5000	3000	5000	1.75	0
AF	500	10	3000	10	1.5	0.2

Without class promotion



With class promotion



Optimal DiffServ Capacity Planning

- DiffServ provides service guarantees to aggregations of traffic, based on static provisioning of resources
- Careful analysis of expected demands, provisioning of adequate resources, and assignment of those resources is essential
- What's *new* about this problem?
  - multiple classes of traffic (EF, AF, BE, …)
- What's *hard* about this problem?
  - multi-class performance constraint → non-linear
  - routing and discrete link capacity → integer programming

Model and Assumptions

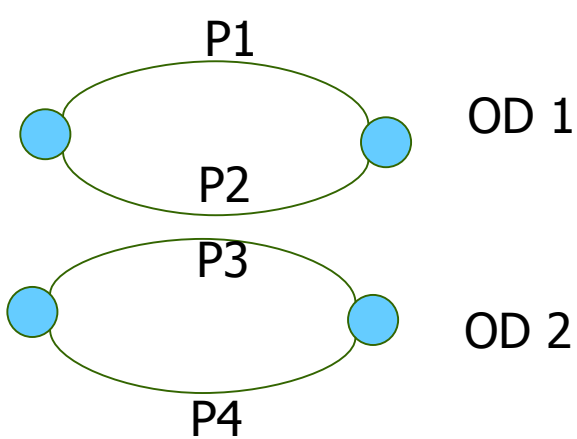
- Given
  - network topology
  - Best Effort user demands (per link)
  - EF user demands from network entry points (origin) to exit points (destination)
- Goal
  - minimize the total network cost
- Solve
  1. route the traffic
  2. dimension (size) the network links
- Constraints
  - Satisfactory performance for both EF and BE traffic

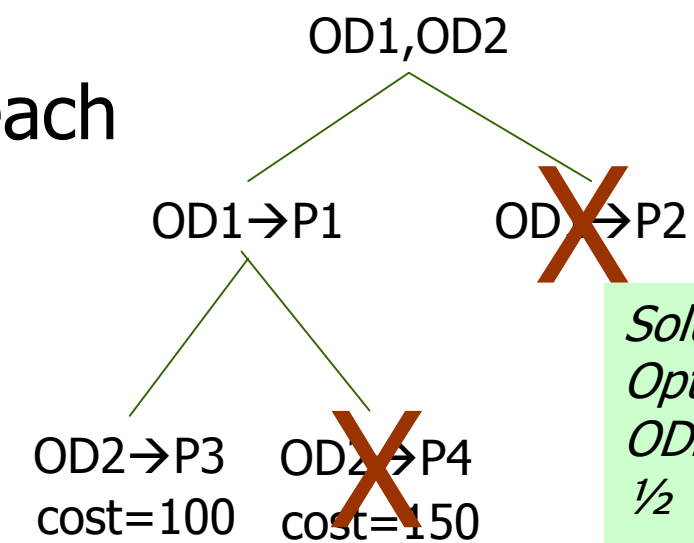
Model and Assumptions (cont'd)

- EF traffic
  - no multiplexing gain for EF traffic
  - ignore limits on maximum EF class utilization (for now)
- Best Effort traffic
  - average packet delay must be constrained➔ convex constraint
  - strict priority queue scheduler is used in every router
  - M/G/1 system: Poisson arrival for both EF and BE, arbitrary packet length distribution
- Link capacities are discrete
- Path-based formulation
  - "candidate" paths for each O-D (origin-destination) pair are given

Proposed Solution Technique

- Initially, assume link capacities are *continuous*
- <u>Branch-and-bound</u>
  - exact (optimal) solution
- Example: 2 OD pairs, each with 2 candidate paths





Solution of Optimal routing: OD2: ½ → P3, ½ → P4, cost=110

Current best solution OD1→P1,OD2→P3, cost=100

Heuristic Solution Techniques

- Single-adjustment local search
  - find an initial solution
  - find the single path adjustment that maximize the total cost reduction
  - repeat until no such single path adjustment can be found
- Multiple-adjustment local search
  - first iteration: single-adjustment local search
  - $(k)^{th}$ iteration: find path adjustment for k O-D pairs that maximize the total cost reduction
  - Repeat until no improvement in (k)th iteration

Solution Technique (cont'd)

- Original problem (discrete link capacity)
- Lagrangean relaxation
  - Place Lagrangean multipliers on constraint, bring them into the objective function
- The dual problem decomposes to two sub-problems
  - one is actually the problem with continuous link capacity!
  - the other one is trivial

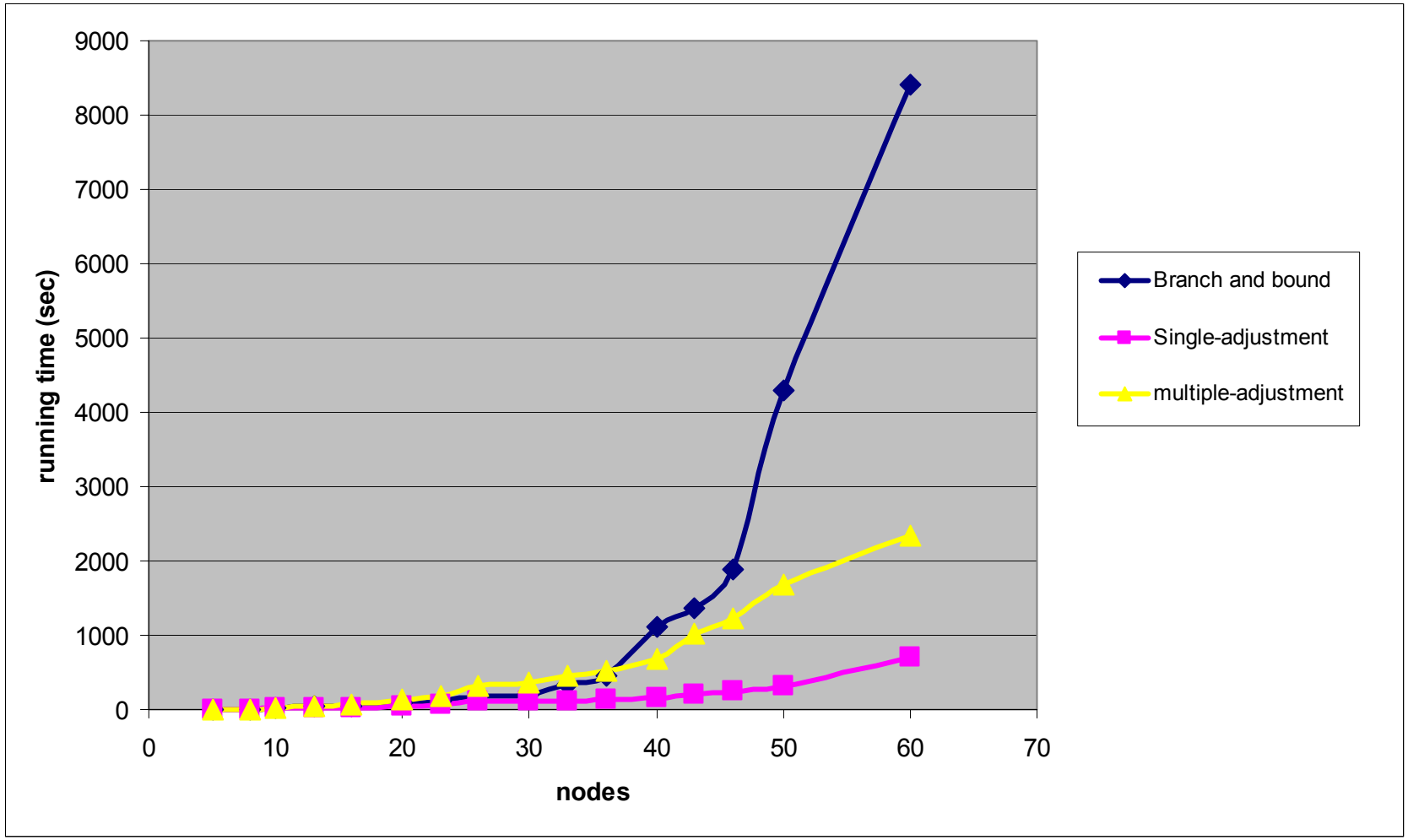
Computational Results

- Generate random network topology using Waxman's method
  - Network size: 5 nodes…60 nodes
  - O-D (traffic) pairs: 10 … 1500 pairs
- Generate candidate paths using Yen's K-shortest path method
  - 10 candidate paths for each OD pair
- Optimal routing is performed using Bertseka's Gradient Projection method

Preliminary Results

- Results of optimal routing show very few OD pairs will split across multiple paths
  - → branch and bound is not so expensive
- Single-adjustment heuristic achieves optimal solution 95% of the time
- Multiple-adjustment heuristic achieves optimal solution 100% of the time

Results (Execution Time)



Status and Future Work

- First results for multi-class capacity planning
  - optimal results possible for "reasonable-sized" networks
- Additional characterization of solutions and algorithms
- Improved accuracy: more traffic classes, more accurate performance models, …
- Integration of pricing and capacity planning
- Apply to capacity planning for failure recovery

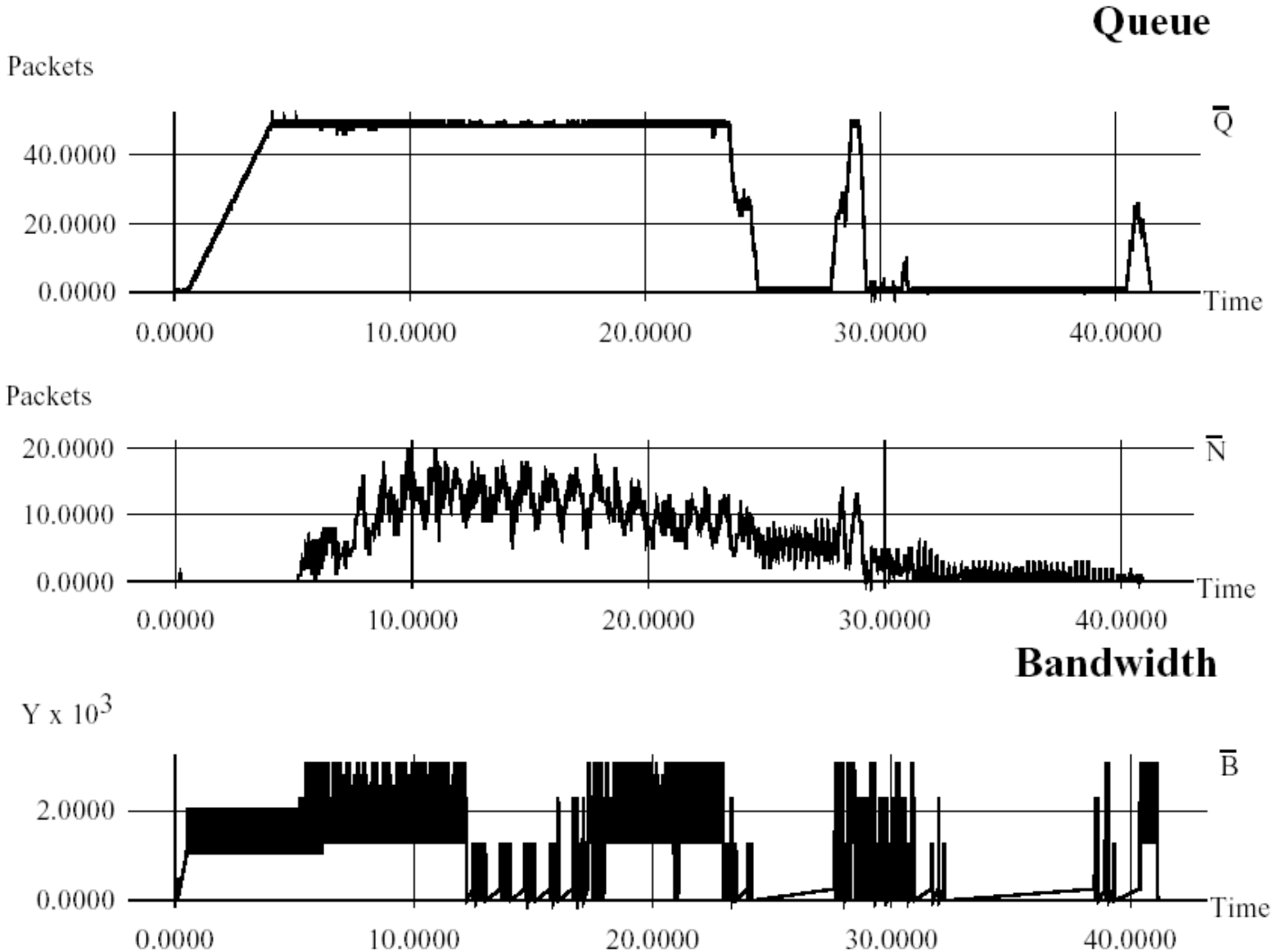
Pricing to Reduce DoS Attacks Against Reliable Multicast

- Properties of Reliable Multicast
  - send the same data to a group of receivers
  - data must be reliably received
  - at approximately the same time
- Applications: on-line trading, gaming / simulation, news distribution, database updates, …
- Achieving reliability: ACKs vs. NACKs

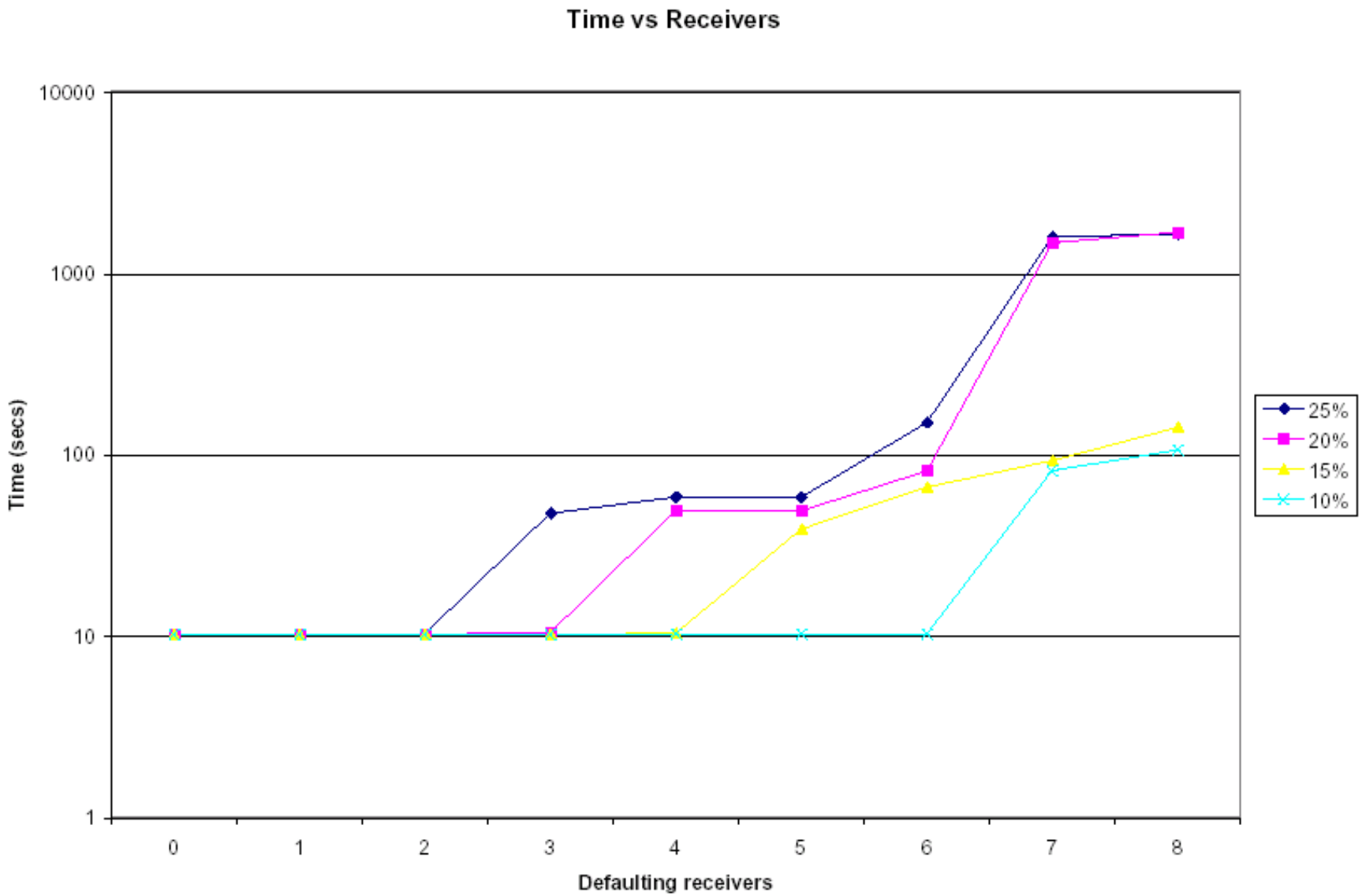
Vulnerability

- Transmission speed is limited by the receive rate of the slowest receiver (yikes!)
- An experiment: receiver impact on PGM
  - simple, scalable NACK-based reliable-multicast method
  - guarantees reliable but not in-order delivery
  - uses "designated local repairers" to recover lost packets
  - simulated using ns-2
- Test network: one sender, 8 receivers, 6 branching nodes (routers)

Exp. #1: Single "Slow" Receiver



Exp. #2: Multiple Slow Receivers



Proposed Solution

- Generally: restrict set of receivers (admission control)
  - "To join this multicast group, you must be able to receive at X Kbps or greater"
- Problem: what should X be?
  - sender-determined or receiver-determined?
- Proposal: Generalized Vickrey Auction (GVA)

Model and Assumptions

- Discrete set of possible transmit/receive rates
- Each receiver has a valuation for each possible transmit rate, and reports this when joining
- GVA: receiver r "pays" for the marginal negative effect that its participation has on the reported value of the other receivers
  - two receivers can pay different amounts for the same receive rate!
- Optimal strategy for receiver (i.e., maximum payoff): tell the truth about valuation
  - strategy is explicit

Example

Receive Speed →	Fast	Medium	Slow
User A value	12	7	3
User B value	0	9	4
User C value	0	0	4
Total	**12**	**16**	**11**

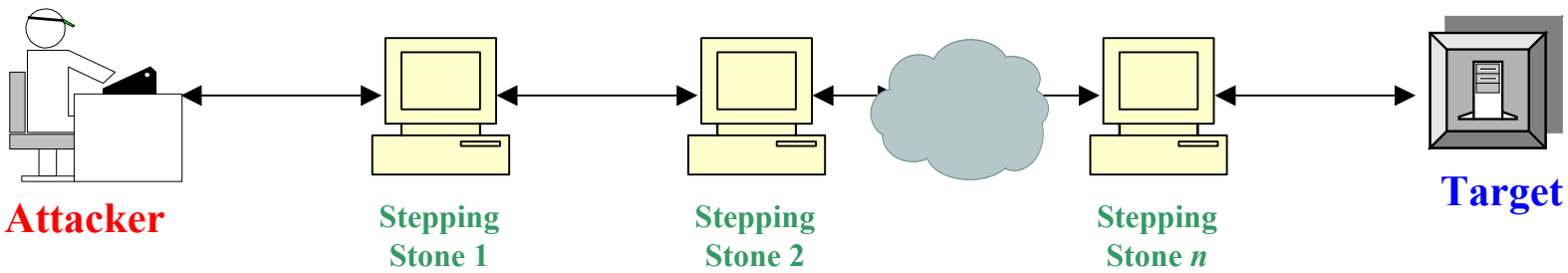
- Optimum send speed is medium
- User A pays 0
- User B pays 5
- User C pays 0

Challenges / Opportunities

- Distributed, dynamic implementation
- Monitoring / policing / billing
- Application to protection of distributed services

Timing-Based Tracing Across Hosts

- Traceback allows tracking down the source of malicious data, despite IP spoofing
- Most traceback does not address the use of "stepping stones" by attackers



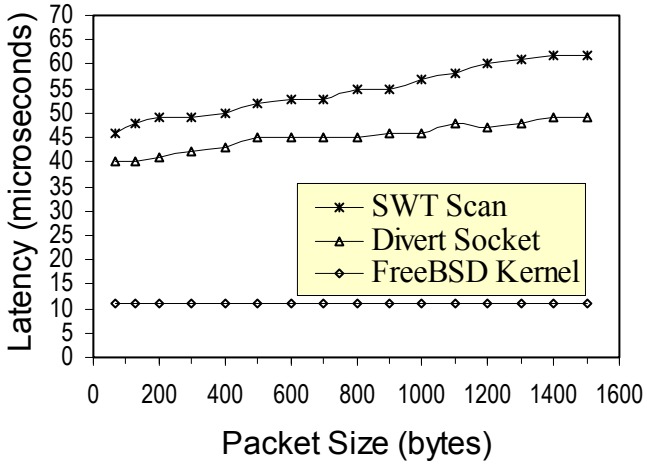
- Active (i.e., modify packets), and passive (no modifications) tracing are both possible
- Our two approaches
  1. application-level tracing (active)
  2. timing-based tracing (passive or active)

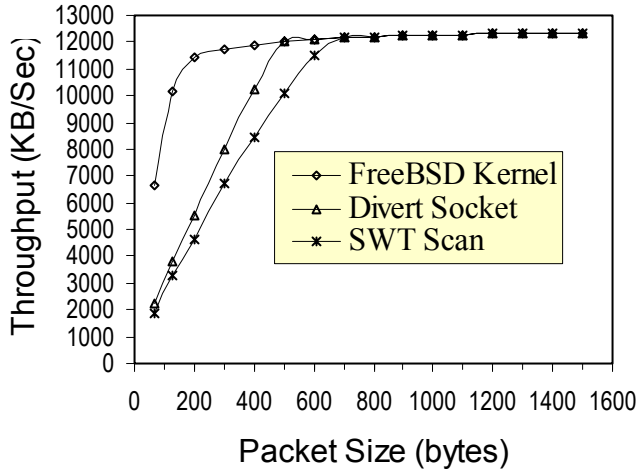
1st Solution: Application-Level

- Idea: track the payload contents across the network
- Challenges
  1. contents not unique
  2. payload examination overhead at routers
  3. payload modification / transformation by intermediate hosts (e.g., encryption)
- Watermarking (fixes #1, uniqueness of contents)
  - target host modifies payload contents on feedback path to attacker; active approach
  4. watermark should be hard to detect without prior knowledge

Application-Level Tracing (cont'd)

- Routers are notified when and what watermarks to watch for (addresses #2, router overhead)





- Steganography helps with #4 (concealment of watermarks)

- No solution for #3 (encryption)

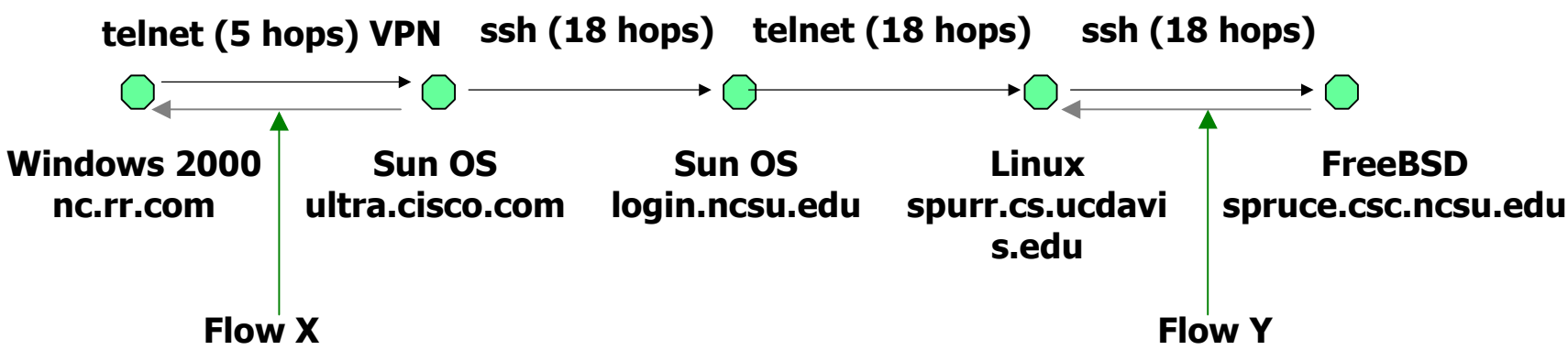
2nd Solution: Timing-Based

- Idea: correlate the packet timing characteristics of flows entering / exiting a stepping stone
- Challenges
  1. timing characteristics not unique across flows
  2. timing collection / processing overhead at routers
  3. timing modification across stepping stones, and across routers
  4. packet dropping / retransmissions, refragmentation, …
  5. timing watermark detectability

Timing Correlation (cont'd)

- Use inter-packet delays (IPDs) as the metric for tracing
  - establish a similarity threshold to detect correlation
  - currently using specific sequences of IPDs
  - later will use statistics of IPDs
- Examine windows of packets only
  - make window size as small as possible (addresses #2, router overhead)

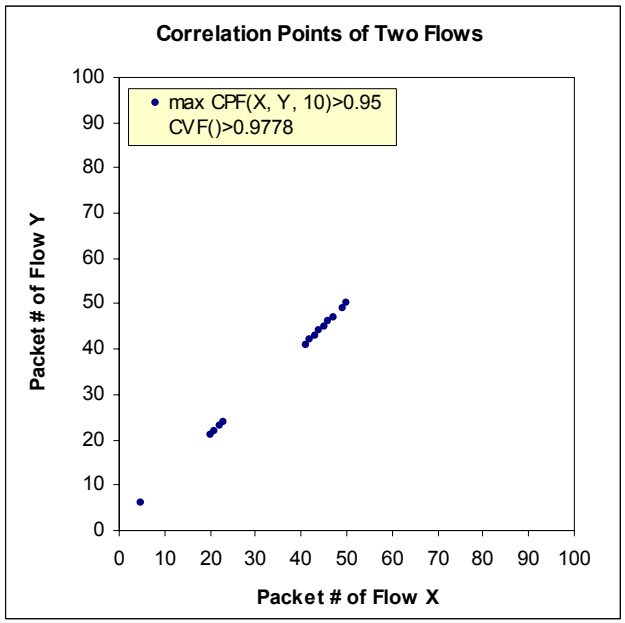
Timing Measurement Experiment

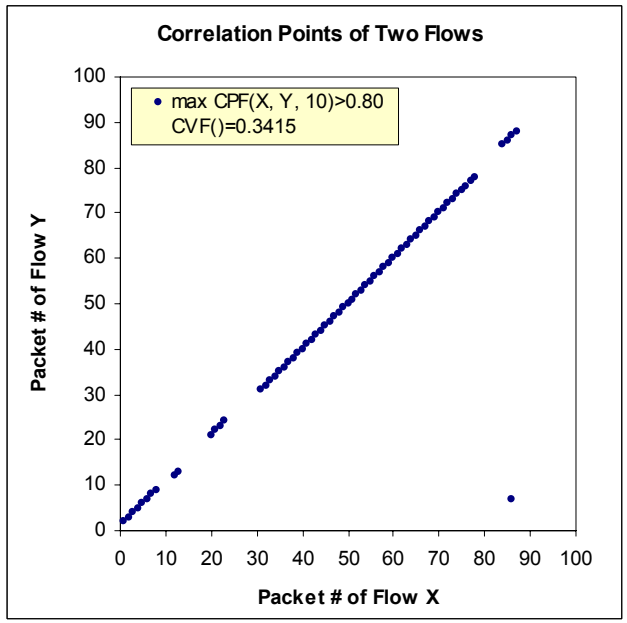


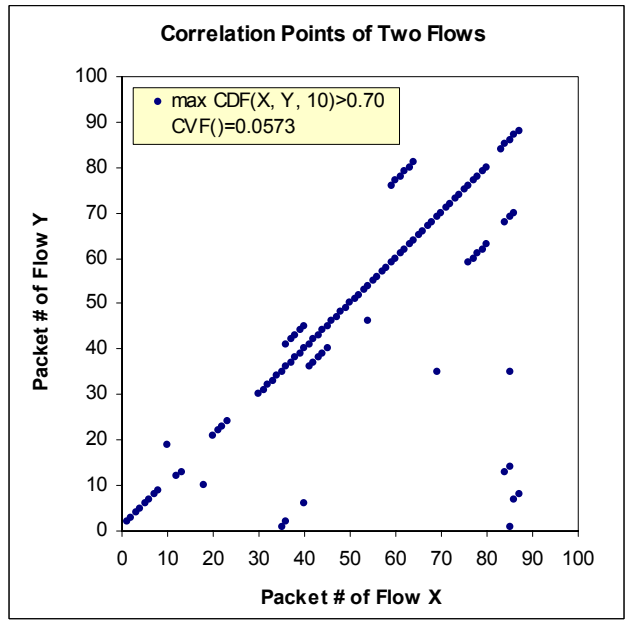
- 4 Different OS Platforms
- 59 Hops and 3 Stepping Stones
- NAT, IPSec VPN, telnet and ssh

Experiment (cont'd)

- Correlation Window Size 10, with varying thresholds
- Correction for retransmissions (#4)







- Result: minimal timing modification (#3)

Uniqueness of Timing (#1)

- Brute-force correlation check
  - Measured IPDs of a generated flow (SSH back traffic)
  - Compared with combined SSH back traffic out of two 3-million packet trans-Pacific trace files

Correlation Window Size	5	10	15	20
"False Positive" Rate	1.6×10^{-4}	6.4×10^{-6}	0	0

Summary

- Capacity planning in DiffServ networks
  - based on pricing
- Protection of reliable multicast against malicious receivers
- Application-level watermarking and timing-based correlation of IP traffic for tracing across stepping stones