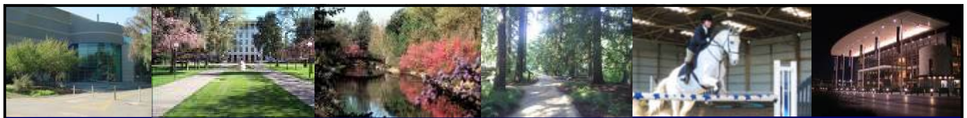


Impact of Sampling on Anomaly Detection

DIMACS/DyDan Workshop on Internet Tomography

Chen-Nee Chuah

Robust & Ubiquitous Networking (RUBINET) Lab

http://www.ece.ucdavis.edu/rubinet

Electrical & Computer Engineering

University of California, Davis

Outline

- Overview
- Impact of Sampling on Anomaly Detection
  - Volume Anomaly Detection
  - Portscan Detection
  - Entropy-based Traffic Profiling
- Towards Accurate Measurements for Anomaly Detection
  - Filtered Sampling
  - Programmable Measurement Approach

Network Monitoring Applications

- Traffic Engineering (TE)
  - Capacity planning, routing, load balancing, fault management
  - Tuning knob: routing configurations, link weights
- Ensuring service level agreements (SLA)
- Security: Detect and keep out unwanted traffic
  - Anomaly/intrusion detection
  - Tuning knobs: IDS rules, firewall configurations

Anomaly Detection

Anomaly detection heavily depends on

- Accurate traffic measurements/observations:
  - What to measure?
  - How to measure? (Limited resources: CPU, memory)
  - Where to measure?
- Robust detection algorithm
  - What is normal/abnormal?
  - Target specific
    - E.g., portscan detection, signature based worm detection
  - Generalized traffic profiling
    - E.g., Entropy based profiling

Detecting Anomalies in IP-Backbone

Why?

- ISPs interested in detecting and stopping anomalous traffic early
  - Additional service to stub networks
  - Protecting scarce resources in wireless access links
- Ability to observe more diverse traffic mix
  - Global view of traffic better capture scanning patterns
- Inherent monitoring capability
  - Sampled traffic used for traffic engineering
    - Cisco's Netflows, Juniper's Traffic Sampling

Sampling

- Sampling typically used in high-speed networks
  - Reduce monitoring/measurement overhead (CPU, memory)
- Sampling distorts traffic statistics
  - Miss packets from the same flow, miss flows all together, …
  - Affect estimates of mean rate, flow size distributions

Coping with Sampled Data

Prior work related to TE

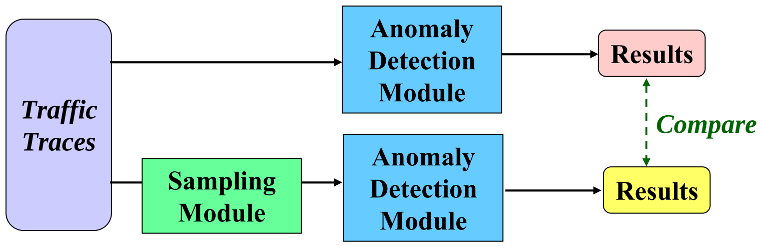
- Inferring accurate flow statistics (flow size or total # of flows) from sampled data [Duffield03, Hohn03]
  - SYN flag in TCP header
- Tracking heavy hitters [Estan02]
- Maintain accurate ranking of flows [Barakat05]
  - TCP/RTP sequence#

Impact of Sampling on Anomaly Detection

- Question we ask:
  Does sampled traffic contain sufficient information for effective anomaly detection?
- Approach: Empirical experiments to gauge impact of sampling on anomaly detection algorithms

[JSAC06] J. Mai A. Sridharan, C-N. Chuah, T. Ye, and H. Zang, "Impact of Packet Sampling on Portscan Anomaly Detection," *IEEE JSAC - Special Issue on Sampling the Internet*, vol. 24, no. 12, pp. 2285-2298, December 2006.
[IMC06] J. Mai, C-N. Chuah, A. Sridharan, T. Ye, and H. Zang, "Is Sampled Data Sufficient for Anomaly Detection?" *ACM/USENIX Internet Measurement Conference*, October 2006 .

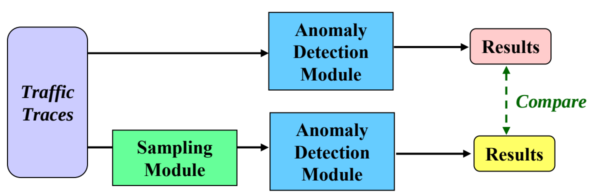
Experiment Methodology



- Backbone traffic traces

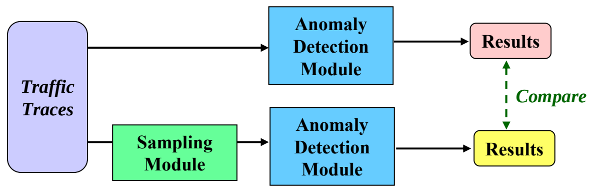
Trace	Average Rate	Anomaly	Duration
BB-East	207 Mbps	DoS	17 hours
BB-West	55 Mbps	Portscan	1 hour
Wireless	7 Mbps	Portscan	3 hours

Anomalous Traffic and Detection Algorithm



Type of Anomalies	Detection Algorithms
Volume anomaly: *DoS attacks, flash crowds*	1. Wavelet-based abrupt change detection [Barford02]
Port scanning: *Worm/virus propagation*	2. Threshold random walk (TRW) [Jung04]
	3. Time Access Pattern Scheme (TAPS) [Sridharan06]

Sampling Methods (1)



Random packet sampling: packets sampled with probability p < 1.

- Simple and efficient, widely deployed (NetFlow)
- Hard to infer flow statistics

Random flow sampling: flows sampled with a probability p < 1.

- Prohibitive resource requirement
- Accurate estimation on flow statistics [Hohn03]

Sampling Methods (2)

Non-uniform flow sampling: focus on catching heavy-hitters

- Smart sampling [Duffield02] – flow records selected with a probability

$$p(x_i) = p_z(x_i) = \begin{cases} \frac{x_i}{z} & \text{if } x_i < z \\ 1 & \text{if } x_i \geq z \end{cases}$$

- Sample-and-hold (S&H) [Estan02]
  - Packet is sampled and flow entry created with probability $h_s=1-(1-h)^s$, as if each byte of a packet sampled with a small probability h.
  - All the following packets in the flow will be sampled once the a packet in the flow gets sampled.

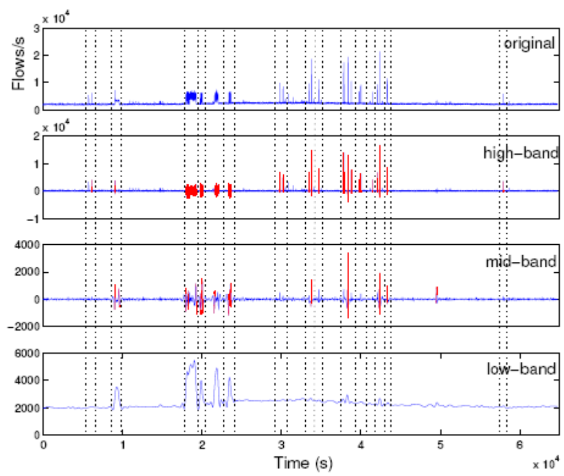
Comparing Various Sampling Schemes

- How to compare: normalizing CPU load, or memory consumption
- Our choice – the percentage of flows sampled
  - Input to the anomaly detection based on flows
  - Number of flows translates to memory consumption
- Example of sampling parameter settings:

% flows	random packet		random flow		smart sampling	
	r	% pkts	p	% pkts	z	% pkts
34.4%	0.1	10.0%	0.344	34.4%	11	84.5%
6.91%	0.01	1.00%	0.691	6.96%	75	62.7%

Case Study #1: Volume Anomaly Detection

- Discrete Wavelet Transform (DWT)* based Change Detection
  - Decomposition
  - Re-synthesis into 3 band:
    - High: 1 second,
    - Mid: 1 minute,
    - Low: 15 minutes.
- Detection
  - Sliding window
  - Deviation score
- Original trace
  - 21 potential anomalies



Detection Result from Sampled Traces

- Apply DWT* to Sampled Data

Sampling interval	10	100	1000
Percentage of flows (%)	36.7	8.03	1.47
Random packet sampling	19	6	1
Random flow sampling	21	18	13
Smart sampling	18	1	1
Sample-and-hold	18	2	1

*[Barford02] P. Barford, J. Kline, D. Plonka, and A. Ron. A Signal Analysis of Network Traffic Anomalies. In Proc. ACM SIGCOMM IMW'02, Nov. 2002.

Impact of Sampling

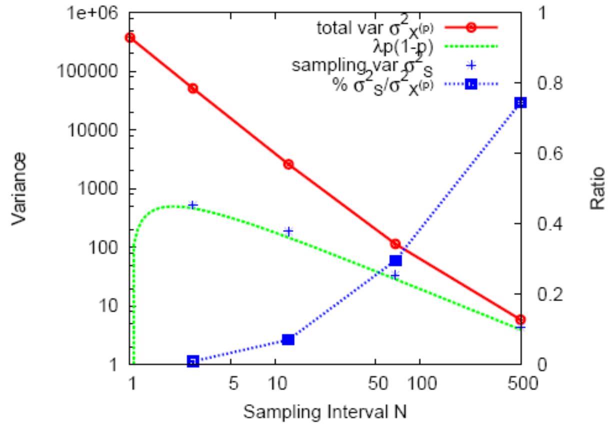
- Sampling distorts variance of time series => signals become noisier, especially at high frequency band
- False Negatives caused by the increase of sampling variance:
  - Let flow arrivals be stationary i.i.d. point process $\{X_t\}$ with variance $\sigma_X{}^2$ and average arrival rate λ
  - With random flow sampling, total variance of sampled process becomes

$$\sigma^2_{X(p)} = p^2\sigma^2_X + \lambda p(1-p) = \sigma^2_{pX} + \sigma^2_S$$

- **No False Positives**

Sampling variance

Sampling Variance



- Variance of the sampled (random flow sampling) time series:

$$\sigma^2_{X(p)} = p^2\sigma^2_X + \lambda p(1-p) = \sigma^2_{pX} + \sigma^2_S$$

Case Study #2: Port Scan Detection

- Port scan typically precedes worm/virus propagation
  - Vertical scan: scan for vulnerable ports on a targeted machine
  - Horizontal scan: scan for vulnerable hosts on a targeted port
- Consider two target-specific detection schemes:
  - TRWSYN

    [Jung04] J. Jung, V. Paxson, A. W. Berger, and H. Balakrishnan, "Fast Portscan Detection Using Sequential Hypothesis Testing," *IEEE Symposium on Security and Privacy*, May 2004.
  - TAPS

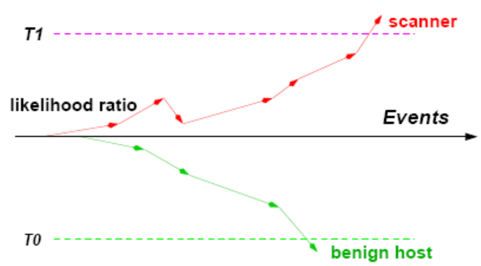
    [Sridharan06] A. Sridharan, T. Ye, and S. Bhattacharyya, "Connection Port Scan Detection on the Backbone," *Malware Workshop*, April 2006.

TRWSYN

- Rationale: scanners makes a lot more failed connection attempts than a benign host
- We need an **ORACLE**
  - which tells upon seeing a SYN packet if the connection will succeed, be rejected or go unanswered ...
- A flow of single SYN-packet is a failed connection
- The connection state drives the random walk.

TRWSYN (Cont'd)

- Sequential Hypothesis Testing
  - Hypotheses: H_0 – a benign host; H_1 – a scanner
  - Sequence of events: Y_i
  - Likelihood ratio $\Lambda(Y) = \prod_{i=1}^{n} \frac{\Pr[Y_i|H_1]}{\Pr[Y_i|H_0]}$
  - Random walk:



TAPS

TRWSYN

- Caveat: single observation point on uni-directional backbone link

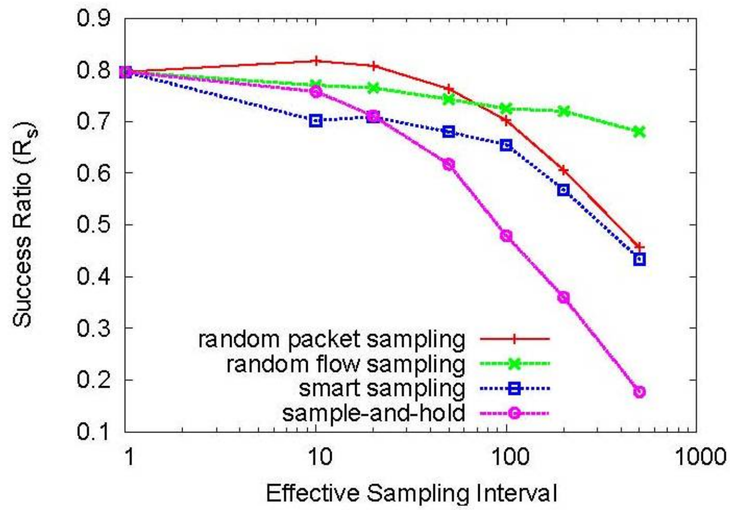
TAPS

- Rationale: scanners tends to access a large number of distinct destination addresses (or port numbers)
- Time-bin driven random walk
  - In each time bin, compute ratio (distinct dest.IP or port #); if exceed threshold k, mark Y_i to 1
  - Update likelihood ratio as TRWSYN
- Designed to lower the false positives

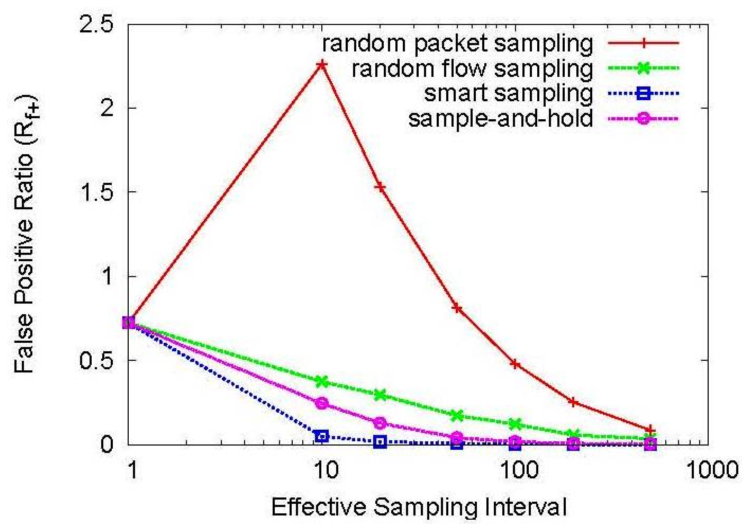
Performance Metrics

- Success Ratio $R_s = \frac{\#(\text{True Scanners Detected})}{\#(\text{True Scanners})}$
- False Positive Ratio $R_{f+} = \frac{\#(\text{False Scanner Detected})}{\#(\text{True Scanners})}$
- R_s => effectiveness, R_{f+} => errors,
- Challenge: how to generate the "True Scanners" set?
  - Use list of scanners manually generated [Sridharan06]
  - We care about relative performance of the portscan detection algorithms with sampled vs. original data
    - Less interested in absolute performance

TRWSYN Detection Results: Success Ratio



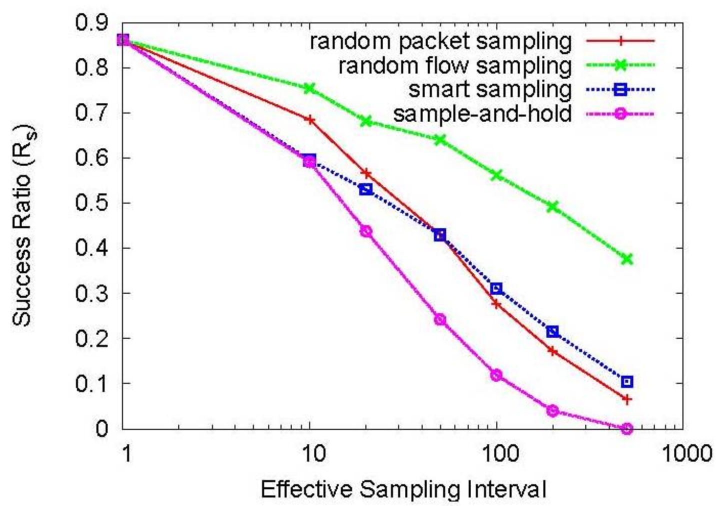
TRWSYN Detection Results: False Positives



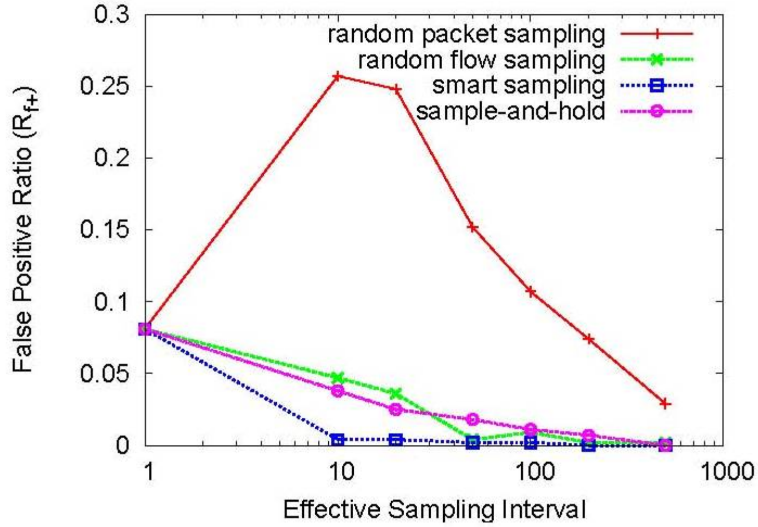
Impact of Sampling

- Flow count reduction – false negatives
- Flow shortening – false positives shoot-up in random packet sampling
  - A multi-packet TCP flow shrunk to a single SYN-packet flow
  - The result: scanners and benign hosts are statistically indistinguishable.

TAPS Detection Results: Success Ratio



TAPS Detection Results: False Positives



Implications of Our Results

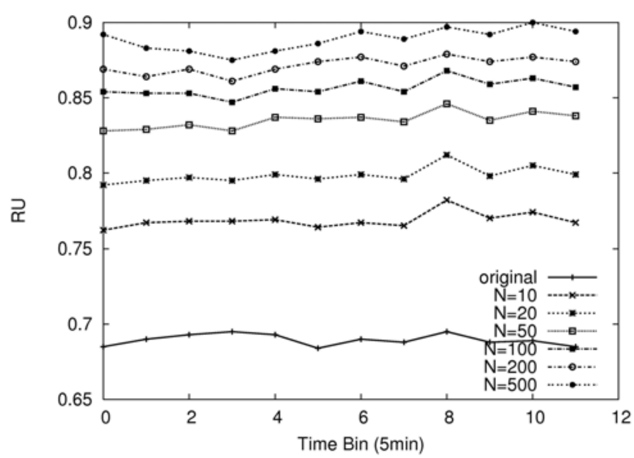
- Random packet sampling is oblivious to any underlying traffic features, and causes information loss and distortion which degrade the performance of anomaly detection algorithms.
- Random flow sampling is generally robust to both volume anomaly and portscan detections.
- Smart sampling and sample-and-hold target heavy-hitters, thus not quite suitable for anomaly detections.

Entropy-Based Traffic Profiling

- We also study non-target-specific detection scheme, e.g, entropy-based traffic profiling*
  - Construct entropy time series along four dimensions {SrcIP, DstIP, SrcPort, DstPort)
  - Extract 'Significant Clusters (SCs)' until the rest looks random (uniform)
  - Categorize SCs into behavior classes (BCs) based on similarity or dissimilarity of communication patterns
- *Sampled traffic tends to be more uniform*
  => increase in entropy & lower # of SCs

*[Xu05] K. Xu, Z. Zhang, and S. Bhattacharrya, "Profiling Internet Backbone Traffic: Behavior Models and Applications," *ACM SIGCOMM*, Aug 2005.

Results with Random Packet Sampling



- Relative Uncertainty (RU) increases, closer to 1
  - Distribution becomes closer to uniform instead of cluster-like

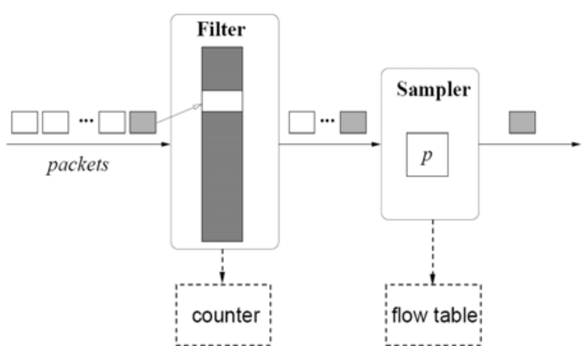
Closing Remarks

Towards accurate measurement for anomaly detection …

- Two on-going directions
  - 'Universal box' that works for both TE & anomaly detection
  - 'Programmable' measurement modules that can be customized depending application requirements

Approach#1: Catching both elephants & mice

- *Preview:* **Fast Filtered Sampling**
  - Goal: Catch both elephants & mice
  - Constraint: Low measurement cost



N counters of m bits

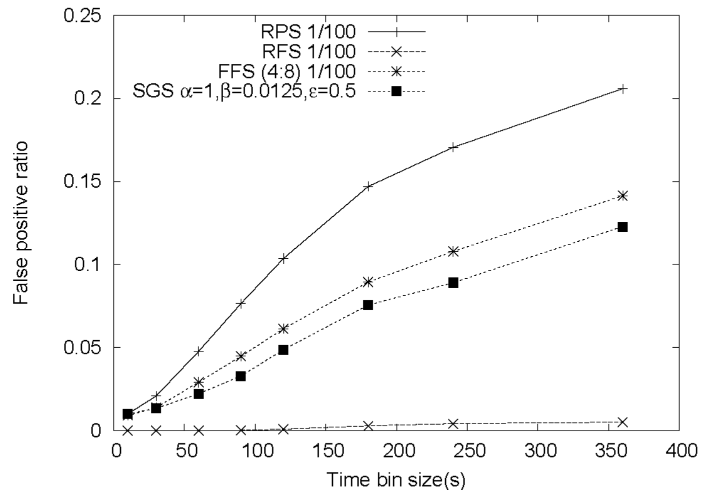
If counter value ≤ s, pass packet to sampler, else discard.

If counter value ≥ l, it is reset to zero.

Pr {packet sampled from flow size i}

$$= \begin{cases} p & \text{if } 1 \le i < s \\ ps/i & \text{if } s \le i < l \\ ps/l & \text{otherwise} \end{cases}$$

Reducing False Positives for TAPS

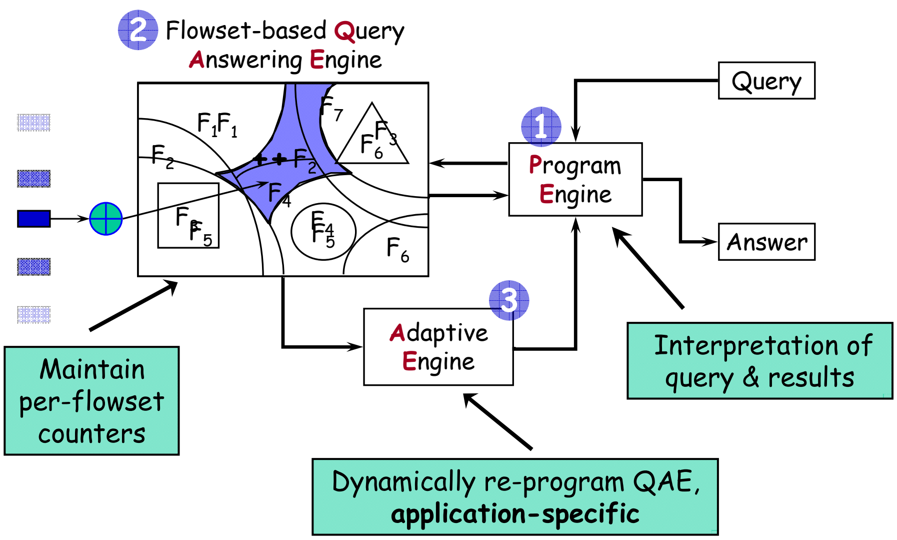


Approach #2: Programmable Measurement

- New abstraction for measurements: **Flowset***
  - Arbitrary set of flows or traffic subpopulation
- Flexibly defined by user
  - E.g. "bogon traffic", "traffic going to ISP X"
- Can be dynamically redefined
  - To match application requirement (TE vs. anomaly detection) or traffic condition
- Significant implication to scalability
  - Per-flowset counters vs. per-flow counter
- **Caveat:** You know what to 'query'

*[Yuan07] L. Yuan, C-N. Chuah, and P. Mohapatra, "ProgME: Towards Programmable Network MEasurement" *ACM SIGCOMM*, Aug 2007

ProgME Architecture



Questions & Comments?

- E-mail: chuah@ucdavis.edu