

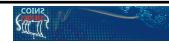


Network Monitoring: Algorithmic Designs and challenges

Yong Guan

Department of Electrical and Computer Engineering
Associate Director for Research, Information Assurance Center
Iowa State University

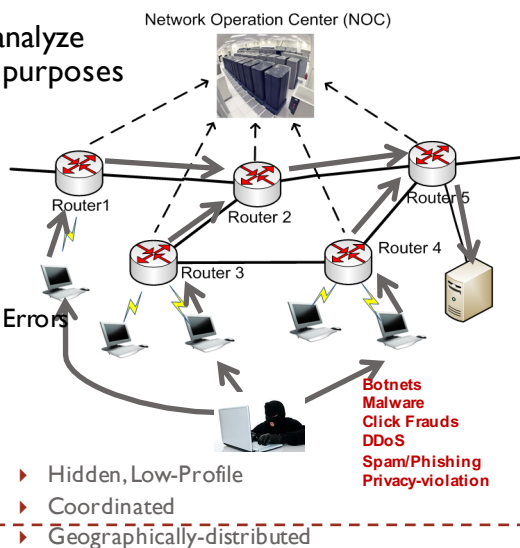
August 5, 2016

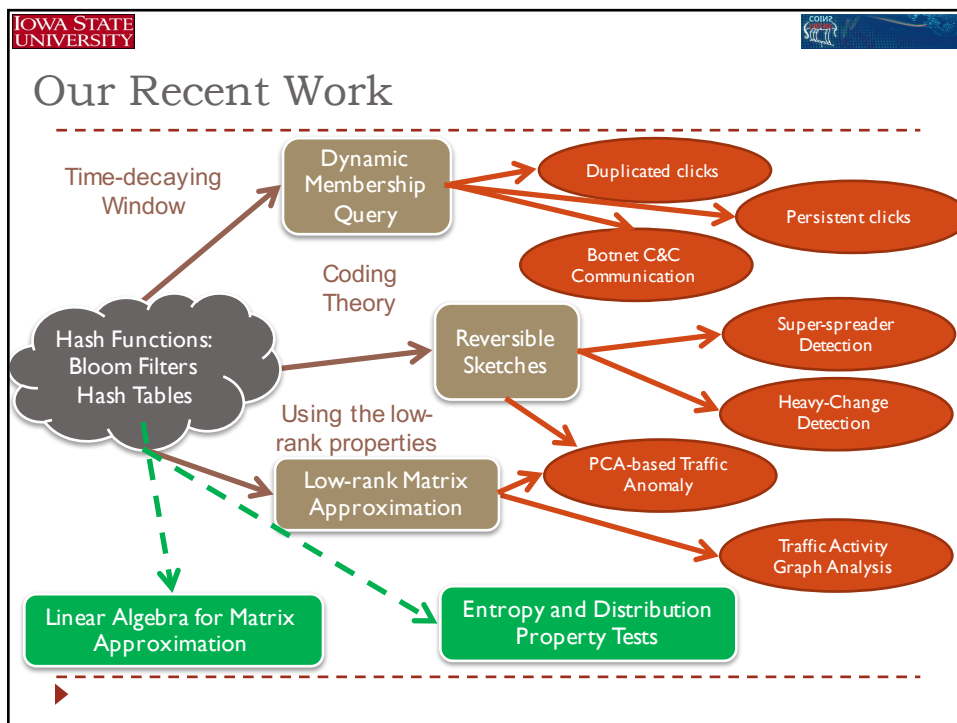
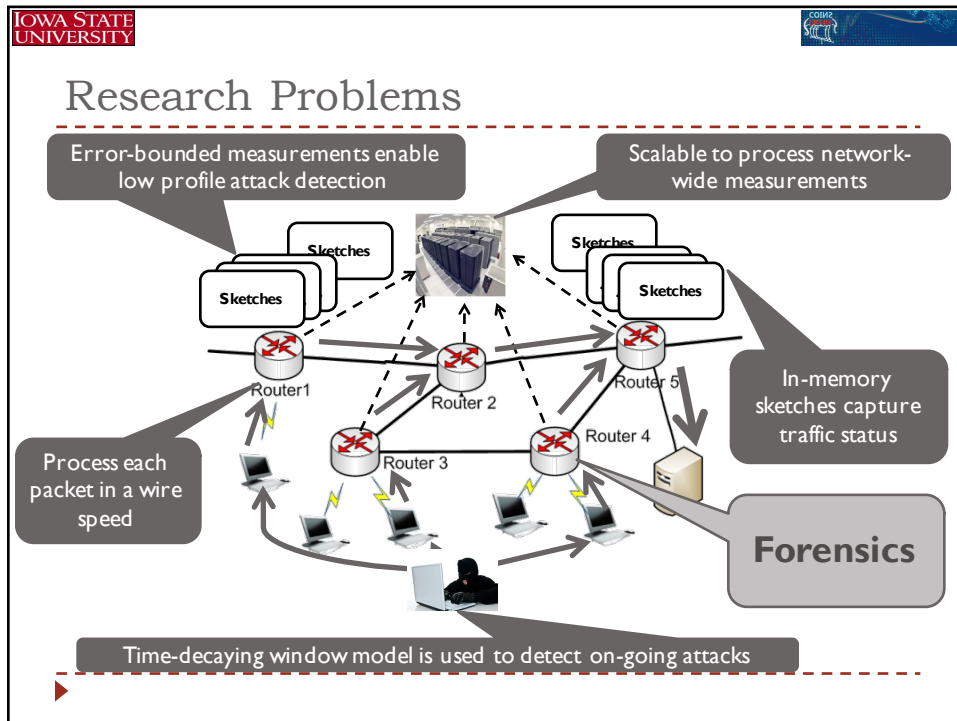


Challenges in Security Monitoring and Forensic Analytics



► We need to measure and analyze network traffic for several purposes

- Usage monitoring:
 - Flash Crowds,
 - Large File Transfers
 - Term-of-service Abuse
- Maintenance:
 - Equipment Failures
 - Vendor Implementation Errors
 - Software Bugs
- Security:
 - Online Fraud Activities
 - Malware Spreading
 - DDoS Attacks





Identifying Significant Contributors from Aggregated Flows

- ▶ Identifying Significant Contributors from Aggregated Flows
- - - - -
- ▶ Sampling provides a poor detection rate.
 - ▶ Sampling rate is low if we need to save packets into the disk
 - ▶ Most malicious packets are missed.
 - ▶ Offline analysis is not enough.
- ▶ Aggregation: Online Solution
 - ▶ Each packet is aggregated into a small number of flows
 - ▶ Maintain a summary for each aggregated flow
 - ▶ Traffic volume
 - ▶ Number of distinct IP, Port, etc.
 - ▶ Entropy
 - ▶ Etc.
 - ▶ Detect attacks and anomalies from aggregated flows in a real time
- ▶ --- - - - -

Limitations of Aggregate Queries

► Advantages

- In-memory data structures
- Fast update for each packet
- Trigger an alarm in a real time

► Weaknesses

- **Difficult to identify the causes of the alarm**

► Existing Solutions

- Modular Hashing
- Combinatorial Group Testing
- Random Projection
- Chinese Remainder Theory



Aggregate Queries

► Heavy-Change Detection

- Identify flow(s) or host(s) that cause sudden changes in traffic volume

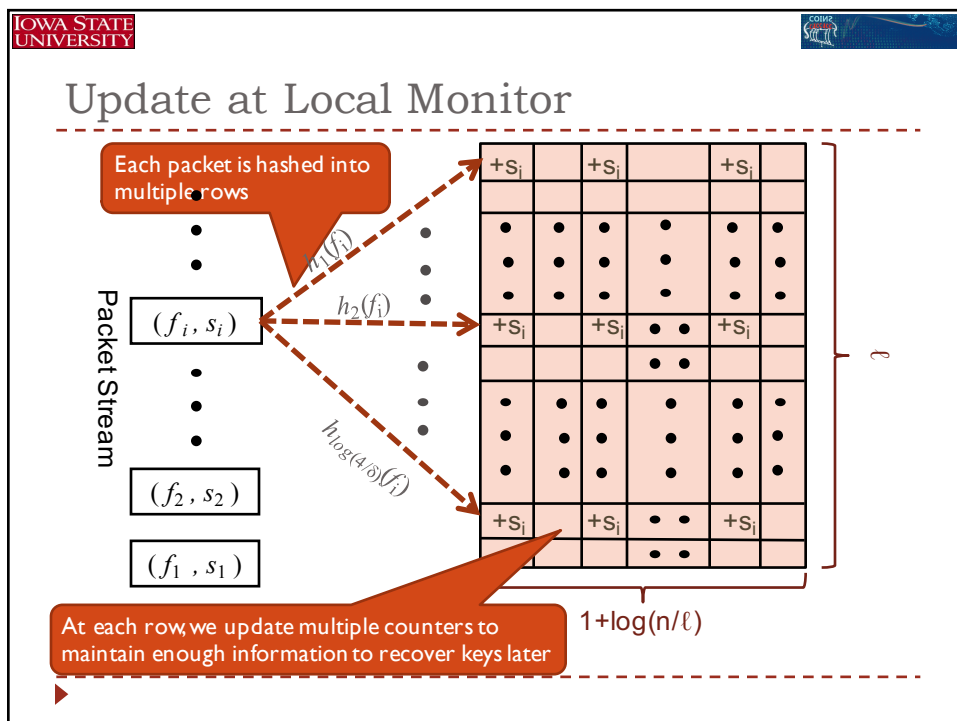
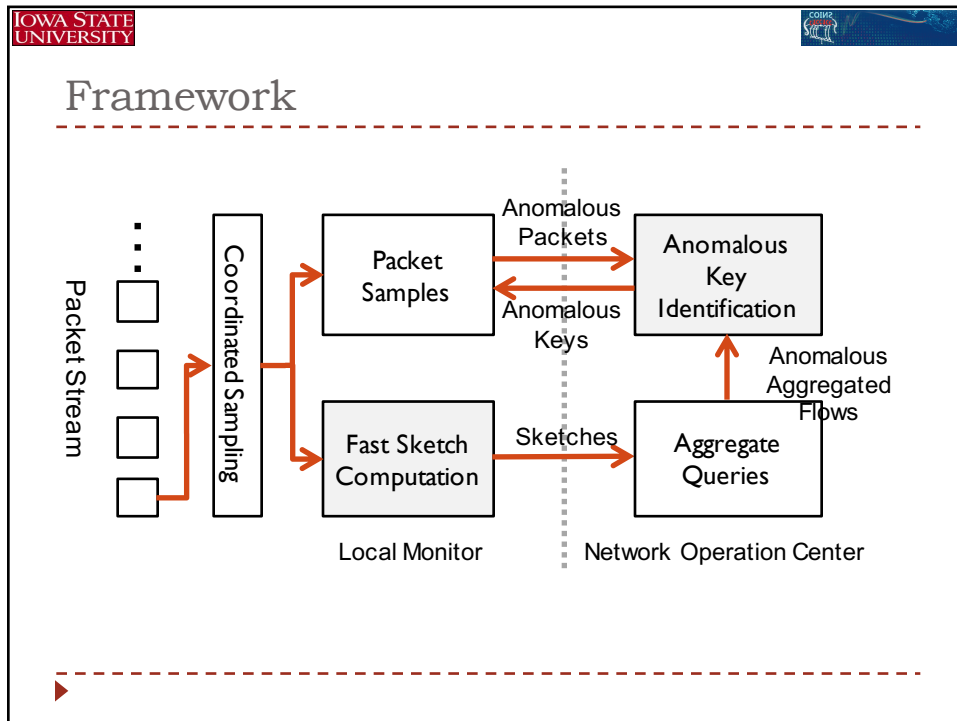
$$|v_f(t) - v_f(t-1)| \geq \theta \Delta V(t)$$

- where

$$f = \{SrcIP, SrcPort, DstIP, DstPort, Protocol\}$$

$$\Delta V(t) = \sum_f |v_f(t) - v_f(t-1)|$$





Update at Local Monitor

- ▶ The quotient function $x=q()$ is the division function

$$q(f) = \lfloor f/\ell \rfloor$$

- ▶ We construct a set of universal hash functions $h'_j()$

$$h'_j(x) = ((\alpha_j x + \beta_j) \bmod P) \bmod \ell$$

- ▶ The hash functions $h_j()$ maps a flow into a row

$$h_j(f) = (f \bmod \ell) \oplus h'_j(\lfloor f/\ell \rfloor)$$

- ▶ Update the first counter in each row $h_j(f)$

$$C_{h_j(f_i),0} = C_{h_j(f_i),0} + s_i.$$

- ▶ Update the b -th counter if the b -th bit in its quotient $q(f)$ is 1

$$C_{h_j(f_i),b} = C_{h_j(f_i),b} + s_i$$

▶

Query at Network Operation Center

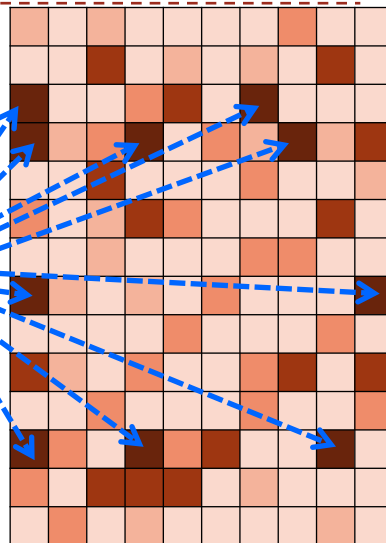
- ▶ Merges all sketches
- ▶ Determine an aggregated flow is anomalous or not

$$|\Delta C_{a,b}^{all}(t)| \geq \frac{\theta}{2} \Delta \hat{V}(t)$$

- ▶ Our Goal
 - ▶ Recover a set of flow keys corresponding the traffic anomalies

?

SrcIP,
SrcPort,
DstIP,
DstPort,
Protocol

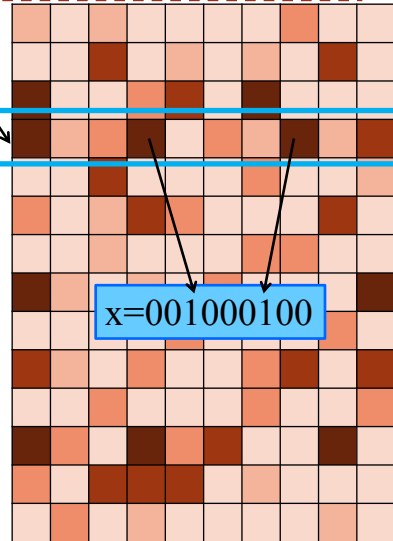


▶

Query at Network Operation Center

- ▶ We go through each row one by one
- ▶ If the first aggregated flow is malicious (its change is larger than the threshold):
 - ▶ We conclude that there is a heavy-change flow in this row
 - ▶ We set b-th bit in its quotient to 1 if the b-th counter is larger than the threshold
 - ▶ We recover its key by

$$\varphi_j(x, y) = x\ell + y \oplus h'_j(x)$$



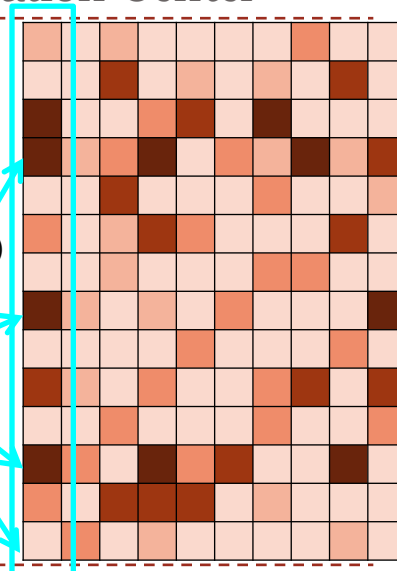
Query at Network Operation Center

- ▶ If there is only one malicious flow in this row, it is easy to verify that

$$\begin{aligned} \varphi(x, y) &= x\ell + y \oplus h'(x) \\ &= \lfloor f / \ell \rfloor g + (f \bmod \ell) \oplus h'(\lfloor f / \ell \rfloor) \oplus h'(\lfloor f / \ell \rfloor) \\ &= \lfloor f / \ell \rfloor g + (f \bmod \ell) \\ &= f \end{aligned}$$

$$f = \varphi(x, y)$$

- ▶ Remove false positives by checking the first counter in each row for $f = \varphi(x, y)$



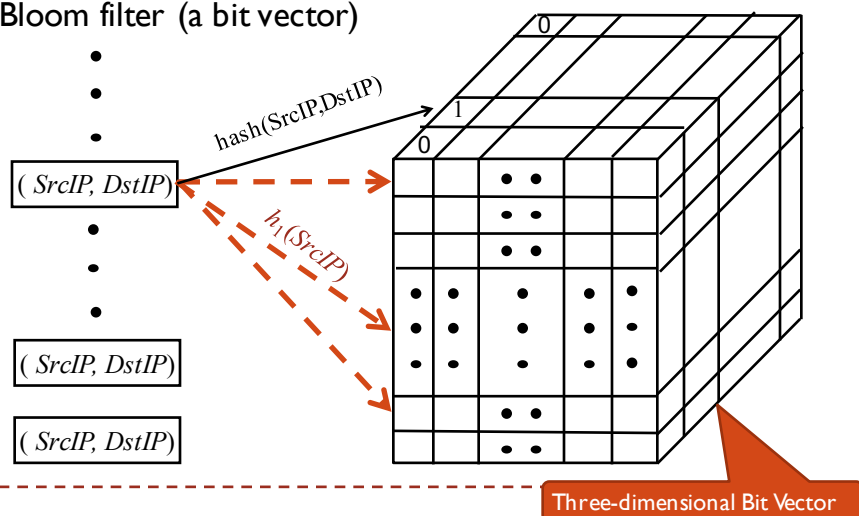
Comparison with Existing Work

	Space	Update Time	Query Time
Modular Hashing	$O(n^{1/\log \log n} \log \log n)$	$O(\log n / \log \log n)$	$O(kn^{1/\log \log n} \log \log n)$
Combinatorial Group Testing	$O(k \log n)$	$O(\log n)$	$O(k \log n)$
Random Projection	$O(k)$	$O(1)$	$O(n)$
Chinese Remainder Theory	$O(n^{0.5})$	$O(1)$	$O(k^2)$
Our Fast Sketch	$O(k \log n/k)$	$O(\log n/k)$	$O(k \log n/k)$



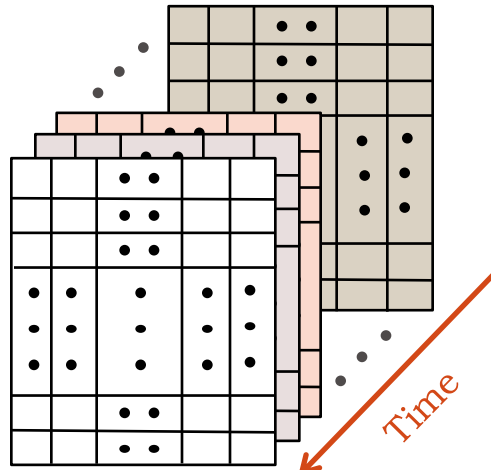
Application 1: Super-spreader

- Each counter is replaced by Bloom filter (a bit vector)



Application 2: PCA-based Anomaly Detection

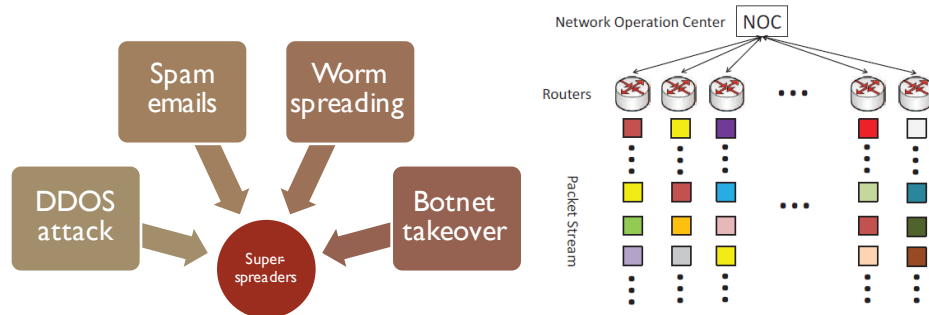
- ▶ We maintain a series of fast sketches.
- ▶ Run Principle Component Analysis on traffic volumes in a time window
- ▶ Identify anomalous aggregated flows
- ▶ Recover the roots of the detected traffic anomalies



**Identifying High-Cardinality
Hosts from Network-wide
Traffic Measurements**

Problem Definition

- High-cardinality hosts (e.g., super-spreaders) are the signs of several known security problems.

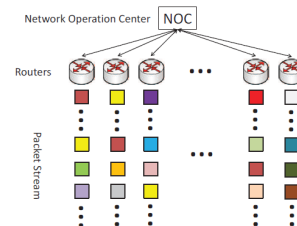


8/25/16

19

Challenges

- Network-wide traffic view
- Duplicate removing
- Mergeable measurements
- Super-spreader identification
- Space & time limitation



8/25/16

20

Problem Formulation

- ▶ We have k routers. For i_{th} router $i = 1, \dots, k$, there is a packet stream

$$(s_{i1}, d_{i1}), (s_{i2}, d_{i2}), \dots, (s_{it}, d_{it}), \dots$$

where s_{it}, d_{it} are the source and destination of the packet.

- ▶ For a source x , the set of distinct destinations of x in i_{th} stream at time t is \mathcal{D}_{it}^x .
- ▶ The destination cardinality of x in all k streams is

$$D_t^x = \left| \bigcup_{i=1}^k \mathcal{D}_{it}^x \right|$$

Problem Formulation (cont.)

- ▶ Let \mathcal{A}_{it} denote the set of distinct packets at i_{th} router at time t with window τ ,

- ▶ Let $\mathcal{A}_{it} = \{(s_{ij}, d_{ij}) \mid j \in [t - \tau, t]\}$. Let F_t denote the number of distinct destinations in k streams:

$$F_t = \left| \bigcup_{i=1}^k \mathcal{A}_{it} \right|.$$


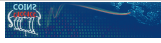
We design an (ϵ, δ) -approximation algorithm which can report any host x such that

$$D_t^x > \theta + \epsilon F_t$$

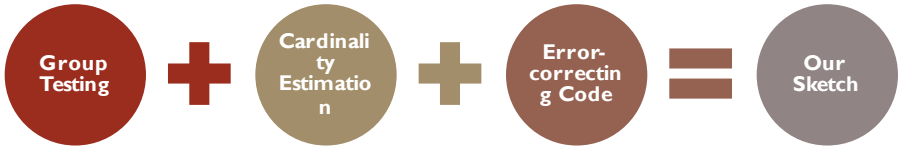
as a high-cardinality host with a probability at least $1 - \delta$, and will report a host x' such that

$$D_t^{x'} < \theta - \epsilon F_t$$

as a high-cardinality host with a probability at most δ .


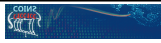



Our Idea - Sketch Design



- Sketches
 - Give (ϵ, δ) -approximations on cardinalities of super-spreaders in each data stream with using space and time.
 - Mergeable: merging two sketches equals to merging two data streams.
 - Reversible: recover the identity of the super-spreaders from the sketch.

8/25/16
23

Group Testing

- ▶ False coin problem
 - ▶ N coins with 1 false (weighing less than a real coin)
 - ▶ Using a balance scale, we can separate the coins in half, compare the two halves and choose the half with less weight.
 - ▶ Repeat the above step until there is one coin remaining which is the false one.
 - ▶ Need $O(\log N)$ weighing.

8/25/16
24

Group Testing in Our Sketch

- ▶ **Randomly map all the hosts into multiple groups**
 - ▶ Each group = 1 super-spreader + n small-cardinality hosts
- ▶ **For each group**
 - ▶ Each of its hosts is randomly mapped to multiple subgroups according to the host's ID.
 - ▶ Maintain cardinality for each subgroup.
 - ▶ The subgroups with cardinality larger than the threshold can identify the super-spreader's ID.

8/25/16

25

Sketch Design

- ▶ **Cardinality Estimation**
 - ▶ For each subgroup, we use an existing optimal cardinality estimation algorithm to maintain the cardinality of the subgroup.
 - ▶ Supports merging of multiple data structures by MAX operation.
 - ▶ Brings in error/noise: subgroups not having a super-spreader may be considered to have one.
- ▶ **Error-correcting Code**
 - ▶ Use error-correcting code, e.g. Hamming code, to encode each host's ID: $q \rightarrow w(q)$.
 - ▶ Encoded IDs are used to map hosts into subgroups.
 - ▶ Decoding $w(q)$ helps us to remove the cardinality estimation errors and get the correct q .

8/25/16

26

Data Structure & Algorithm

3. Each subgroup where (s,d) is mapped to will update its cardinality using the destination d .

(s, d)

$h_j(s)$

$c_{s,b}(d)$

L layers(groups).

Counters used in cardinality estimation for each subgroup.

1. Each packet is independently hashed into multiple groups according to the source s . Hash functions are based on the quotient and remainder of s divided by L .

n subgroups for group testing + 1 subgroup for FP removing

2. In each group, (s,d) is mapped into multiple subgroups according to the 1-bit of quotient q of s divided by L . Error-correcting code is used to encode q to $w(q)$ before mapping.

27

Algorithm – Recover Super-spreaders from Sketch

Create a 2D binary matrix from $C[*,*,*]$; test each subgroup $C[a,b,*]$ in each layer/group to see if its cardinality is larger than the threshold. If yes, set $B[a,b]=1$, else set $B[a,b]=0$.

$B[*,*]$

Layers (groups)

1	0	0	0	1	0	0	0	1	0
0	1	1	0	0	0	1	1	0	0
0	0	1	0	0	0	0	0	0	1
0	0	0	1	0	1	0	0	0	0
1	0	0	1	0	0	0	0	0	0
0	1	0	0	1	1	0	1	0	0
1	1	1	0	1	1	1	1	0	0
0	0	0	1	1	1	0	0	0	0
1	0	0	0	0	1	0	1	0	0
0	0	0	1	1	0	0	0	0	0

Subgroups

$W(y) = 000001010$

8th layer

try each of the hash functions on decoded y

+

$a = 1000$. Layer number is also used to recover the super-spreader's ID.

+

$y = 0010$. y is the quotient of the super-spreader in this group with high probability.

super-spreader candidate x

decoding

28

Algorithm –False Positives Filtering

On each candidate x , try each hash function to see if the group it is mapped to has a super-spreader.
If half of the groups do have, then x is reported as a super-spreader.

Candidate
 x

$h_j(x)$

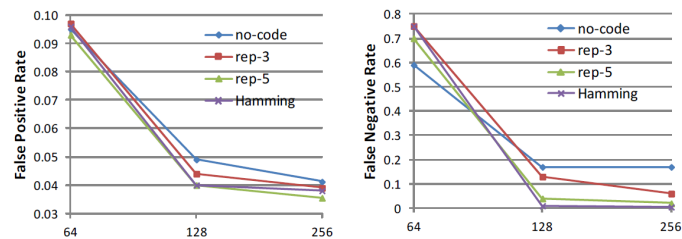
$B_t[*,*]$

1	0	0	0	1	0	0	0	1	0
0	1	1	0	0	0	1	1	0	0
0	0	1	0	0	0	0	0	0	1
0	0	0	1	0	1	0	0	0	0
1	0	0	1	0	0	0	0	0	0
0	1	0	0	1	1	0	1	0	0
1	1	1	0	1	1	1	1	0	0
0	0	0	0	1	1	0	0	0	0
1	0	0	0	0	0	1	0	1	0
0	0	0	1	1	0	0	0	0	0

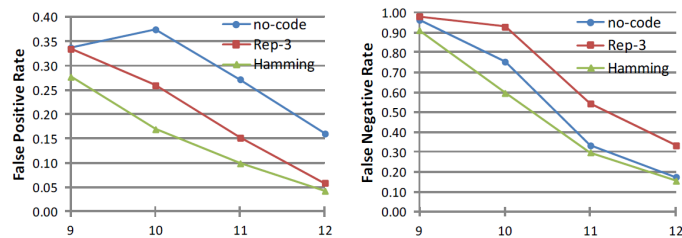
8/25/16

29

Performance



(a) Compare Coding Methods



(b) WittyWorm Realtrace

8/25/16

30

Small Group Discussions

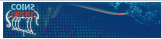

- ▶ Balancing accountability and PRIVACY
- ▶ Working with big, sometimes incomplete data
- ▶ Mobile and wearable platform forensics
- ▶ Proving the relevance of evidence and human/platform that generated them.



"Liberty means responsibility. That is why most men dread it."

— George Bernard Shaw





Thanks Q&A

Yong Guan
guan@iastate.edu
Iowa State University

