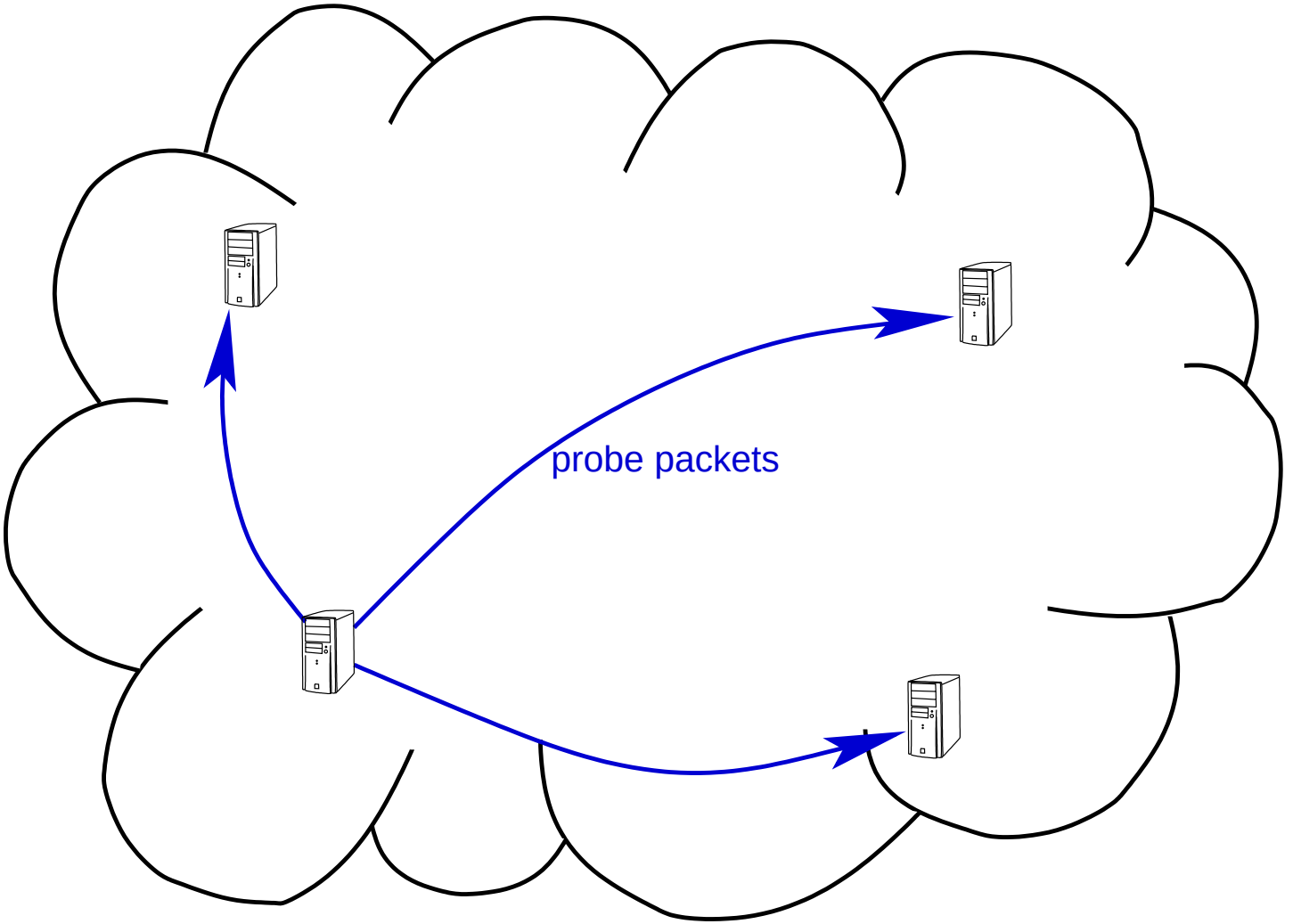
Privacy-Preserving Performance Measurements

Matthew Roughan
University of Adelaide
<matthew.roughan@adelaide.edu.au>

Yin Zhang
University of Texas
<yzhang@cs.utexas.edu>

Performance measurements

Active performance measurement by sending probe packets.



Why measure performance?

THE UNIVERSITY OF ADELAIDE AUSTRALIA

- Network quality assurance
  - to improve performance you must measure it
  - find problems quickly and repair
- Optimize for performance
  - want to test optimizations work
- Support of SLAs
  - customers often want high performance
  - need to prove it to them

What do we need?

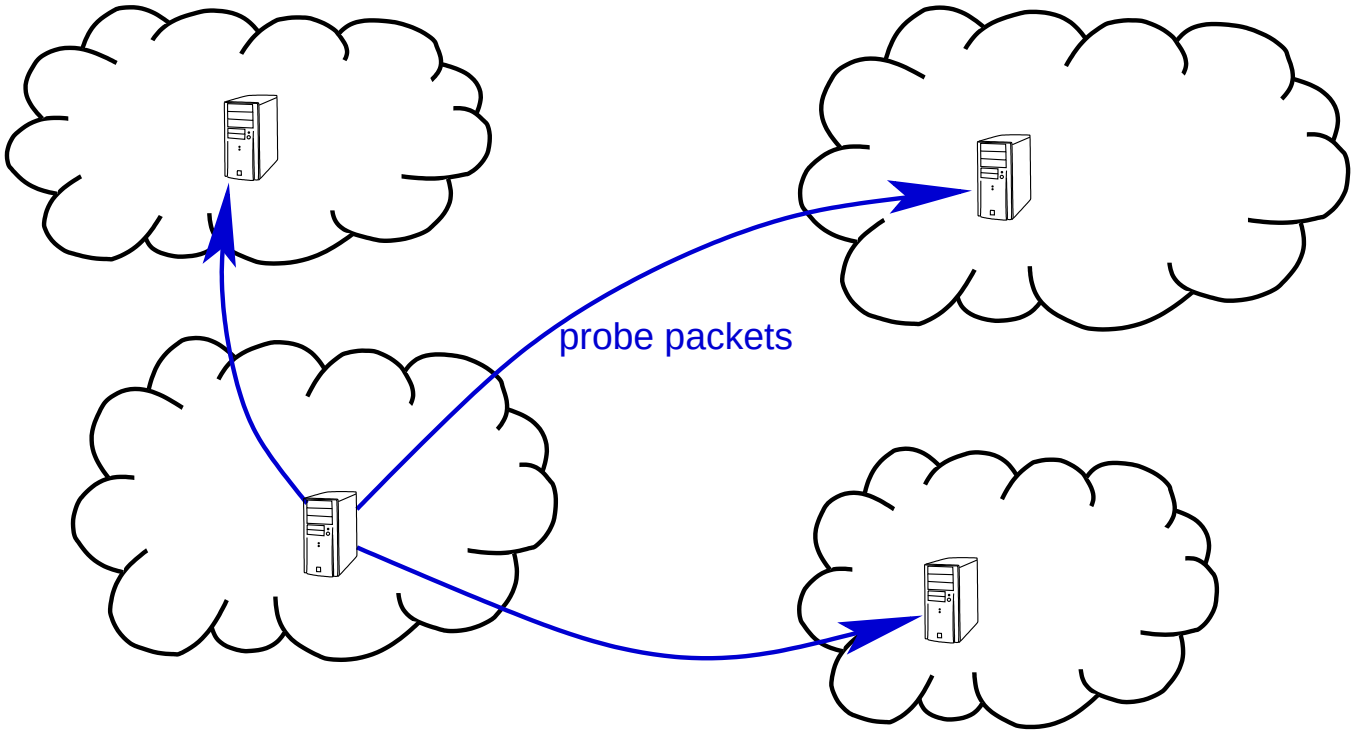
- We need inter-domain measurements
  - most problems happen at the edge
    - BGP routing is not transparent
      - hard to configure
      - hard to debug
    - peering links are a likely location for congestion
  - Intra-domain measurements are "easy"
- Measurements should be one-way
  - inter-domain routing is **intrinsically** asymmetric
  - we have reasonable control over outgoing traffic, but limited control of incoming traffic

Performance Measurements

THE UNIVERSITY OF ADELAIDE AUSTRALIA

ISPs measure one-way inter-provider performance

- **inter-provider:** many problems occur at the edges
- **one-way:** inter-ISP routing is asymmetric



So what is hard?

- no particular company controls all the Internet
  - the Internet is (by its nature) distributed
  - we need measurements between these companies
- Companies don't like to share
  - companies don't want to reveal data
    - afraid of misuse of data
    - afraid it will reveal business secrets
    - afraid it will reveal incompetence
  - sometimes they are not allowed to
    - e.g. privacy legislation [1]

Related problems

THE UNIVERSITY OF ADELAIDE AUSTRALIA

- How much traffic is there on the Internet?
  - the argument is made [2] that lack of such data contributed to the tech-wreck
  - regulators need such information
    - e.g. anti-trust cases
- Detecting distributed attacks
  - DDoS (Distributed Denial of Service), Worms/viruses,
  - e.g. Worms are easy to detect once they are well under way, but if you want to detect it early, the more data points you have the better.
- but if companies won't share data, how can we collect Internet wide measurements?

Similar problems elsewhere

THE UNIVERSITY OF ADELAIDE AUSTRALIA

- The Center for Disease Control and Prevention (CDC) who have to detect new health threats
  - need data from
    - hospitals
    - insurance companies, airlines, ...
    - NGOs (e.g. charities)
    - other government bodies
  - data is
    - proprietary (e.g. insurance risks)
    - protected by privacy legislation
  - data-mining community has developed solutions
    - secure-distributed computing [3, 4, 5]
    - privacy-preserving data-mining [6, 7]

Trusted third party

- simple answer: a trusted third party
  - independent party (e.g. with no vested interest)
  - trusted by all other parties
  - collects data, and shares aggregated results
- problems:
  - hard to find such parties
    - need to be trusted by all parties in the measurements
  - often requires special legislation
    - lacks flexibility

A better way

THE UNIVERSITY OF ADELAIDE AUSTRALIA

There are some generic techniques that can help us out

- Secure Distributed Summation (SDS)
- Secure distributed dot product
- Oblivious transfer

Secure Distributed Summation

Problem: N parties each have one value v_i and they want to compute the sum

$$V = \sum_{i=1}^{N} v_i$$

but they don't want any other party to learn their value.

SDS algorithm [6]

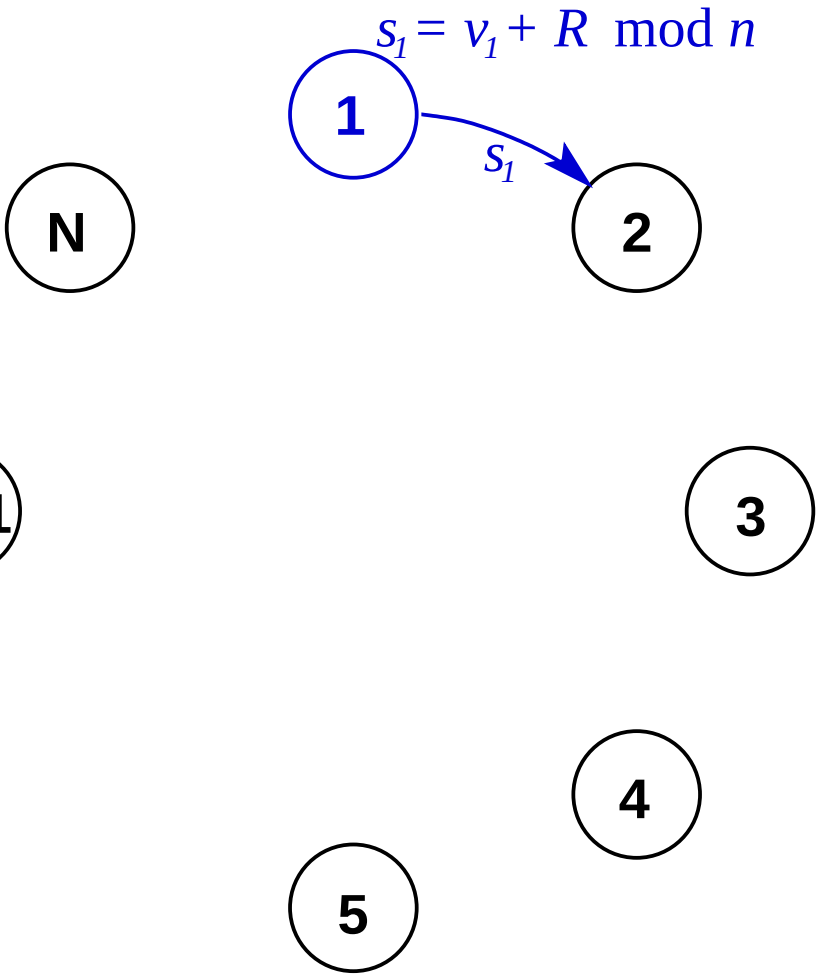
Assume the value $V \in [0, n-1]$ (for large n)

party 1: randomly generate $R \sim U(0, n-1)$
party 1: compute $s_1 = v_1 + R \mod n$
party 1: pass s_1 to party 2
for i=2 to N
  party i: compute $s_i = s_{i-1} + v_i \mod n$
  party i: pass s_i to party $i+1$
endfor
party 1: compute $v_N = s_N - R \mod n$

Finally, party 1 has to share the result with the others.

s_i will be uniformly randomly distributed over $[0, n]$ and so we learn nothing about any other parties' values.

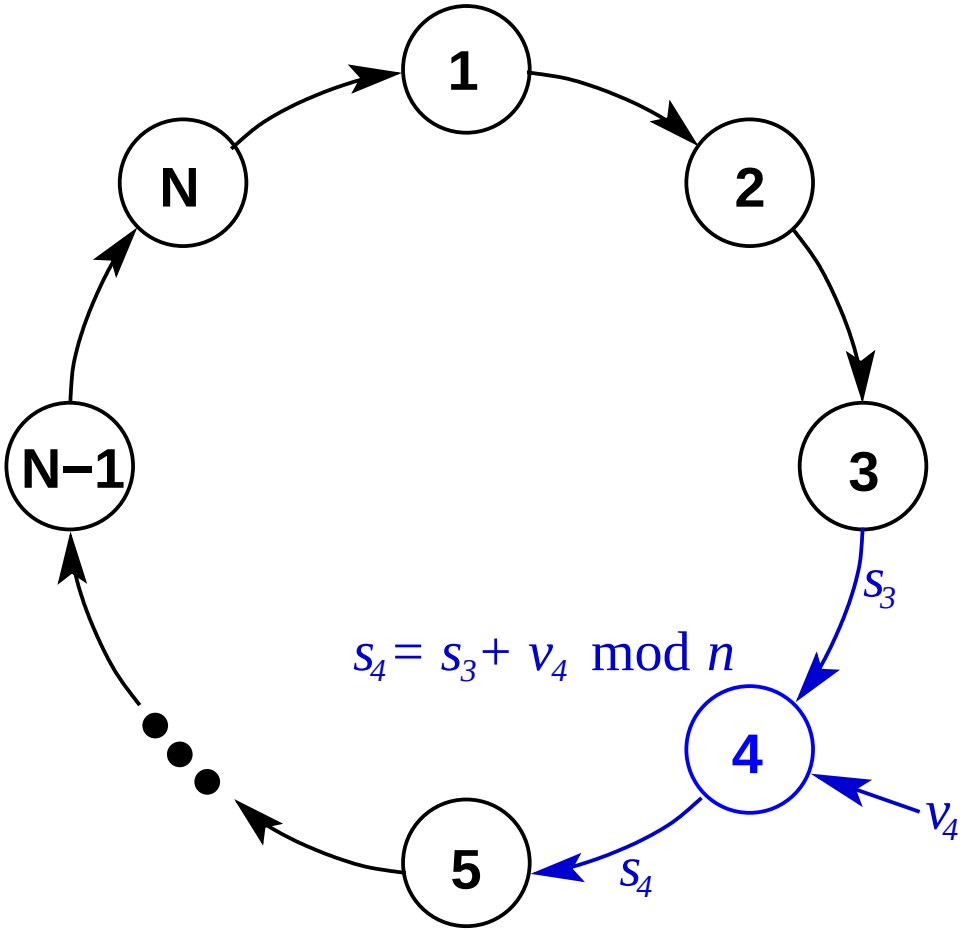
SDS algorithm



party 1: randomly generate $R \sim U(0, n-1)$
party 1: compute $s_1 = v_1 + R \mod n$
party 1: pass s_1 to party 2
for i=2 to N
  party i: compute $s_i = s_{i-1} + v_i \mod n$
  party i: pass s_i to party $i+1$
endfor
party 1: compute $v_N = s_N - R \mod n$

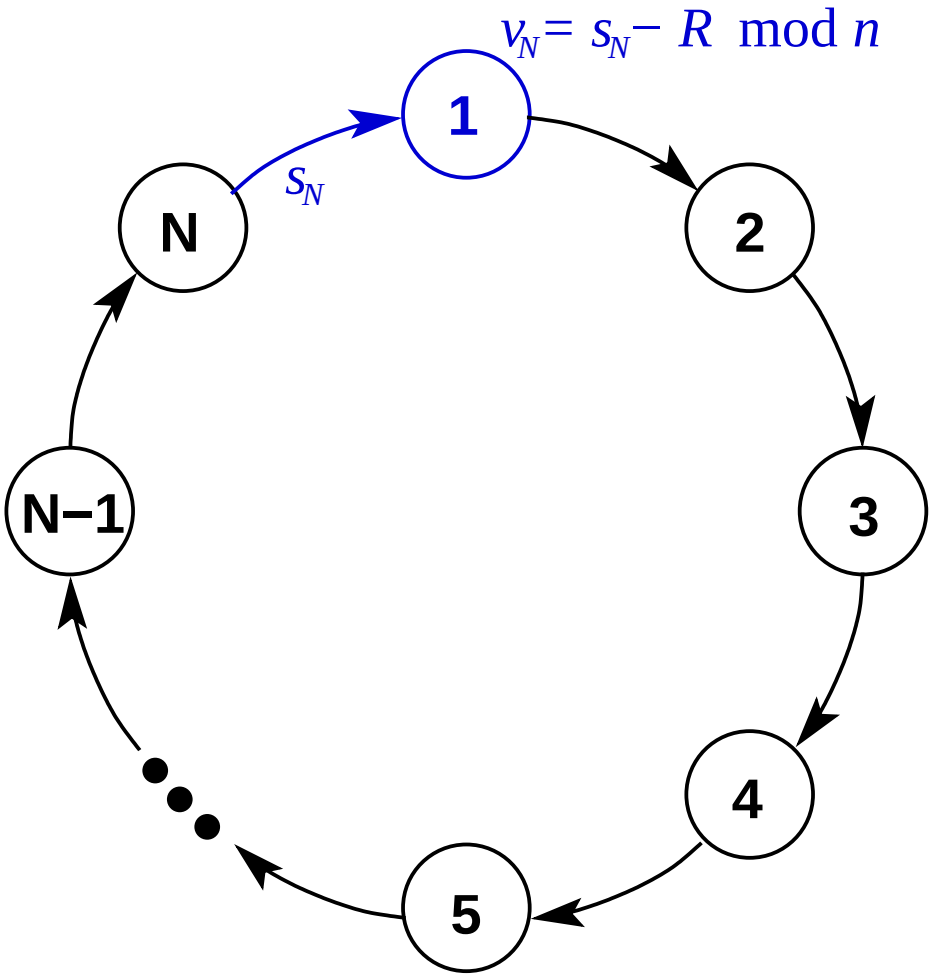
SDS algorithm

party 1: randomly generate $R \sim U(0, n-1)$
party 1: compute $s_1 = v_1 + R \mod n$
party 1: pass s_1 to party 2
for i=2 to N
  party i: compute $s_i = s_{i-1} + v_i \mod n$
  party i: pass s_i to party $i+1$
endfor
party 1: compute $v_N = s_N - R \mod n$



SDS algorithm

party 1: randomly generate $R \sim U(0, n-1)$
party 1: compute $s_1 = v_1 + R \mod n$
party 1: pass s_1 to party 2
for i=2 to N
  party i: compute $s_i = s_{i-1} + v_i \mod n$
  party i: pass s_i to party $i+1$
endfor
party 1: compute $v_N = s_N - R \mod n$



Applications

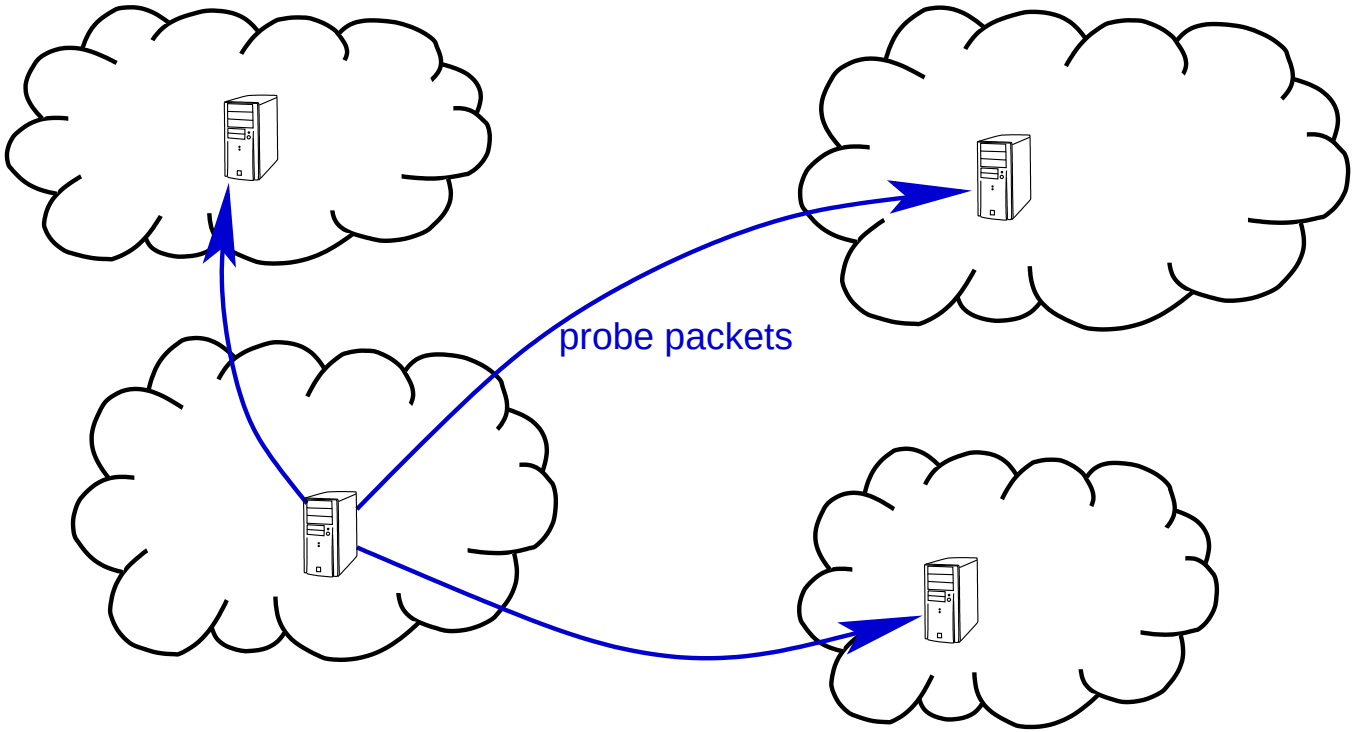
- calculating the total traffic on the Internet
  - v_i is total per ISP
- more sophisticated traffic measurements
  - detection of large-scale security threats
    - e.g., worms, viruses, large-scale DDoS
- sketches can be used this way
- intra-domain performance measurements
  - e.g. v_i is packet loss percent at each ISP
  - use sum to compute (weighted) average
  - provide an overall Internet Health metric

Inter-domain Measurements

THE UNIVERSITY OF ADELAIDE AUSTRALIA

ISPs measure one-way inter-provider performance

- **inter-provider:** many problems occur at the edges
- **one-way:** inter-ISP routing is asymmetric



Internet perf. measurement

THE UNIVERSITY OF ADELAIDE AUSTRALIA

Experiment and notation:

- send K_{ij} probe packets from ISP $i \longrightarrow j$
- sender i notes transmit times $t_{ij}^{(k)}$
- receiver j notes receive times $r_{ij}^{(k)}$
- delay $d_{ij}^{(k)} = r_{ij}^{(k)} - t_{ij}^{(k)}$
- averages:

$$\bar{D}_{ij} = \frac{1}{K_{ij}} \sum_{k=1}^{K_{ij}} r_{ij}^{(k)} - t_{ij}^{(k)}$$

$$\bar{R}_{ij} = \frac{1}{K_{ij}} \sum_{k=1}^{K_{ij}} r_{ij}^{(k)}, \qquad \bar{T}_{ij} = \frac{1}{K_{ij}} \sum_{k=1}^{K_{ij}} t_{ij}^{(k)}$$

Internet perf. measurement

- but ISPs don't want others to be able to make comparisons?
  - obviously this limits the type of measurements we can make: consider averages across providers, e.g.

$$\bar{D}_i^{\text{out}} = \frac{1}{N-1} \sum_{\substack{j=1 \\ j \neq i}}^{N} \bar{D}_{ij}$$

  - limits what data can be shared:
    - ISPs can't share individual measurements $r_{ij}^{(k)}$ or $t_{ij}^{(k)}$

SDS to the rescue

$$\bar{D}_i^{\text{out}} = \frac{1}{N-1} \sum_{\substack{j=1 \\ j \neq i}}^{N} \frac{1}{K_{ij}} \sum_{k=1}^{K_{ij}} \left[r_{ij}^{(k)} - t_{ij}^{(k)} \right]$$

$$= \frac{1}{N-1} \left[\sum_{\substack{j=1 \\ j \neq i}}^{N} \bar{R}_{ij} - \sum_{\substack{j=1 \\ j \neq i}}^{N} \bar{T}_{ij} \right]$$

- $\sum_{\substack{j=1 \\ j \neq i}}^{N} \bar{T}_{ij}$ is already known by i
- $\sum_{\substack{j=1 \\ j \neq i}}^{N} \bar{R}_{ij}$ calculate using SDS and give i the result

Honest but curious model

- any party could corrupt the total V by inputing incorrect data v_i
- calculation has implicit assumption of honesty
  - let us extend this
- "Honest but curious" security model
  - **honest:** honestly follow protocol
  - **curious:** may perform more operations to try and learn more information (than they were supposed to learn)
- doesn't prevent colluding coalitions
- conditions can be weakened (e.g. honest majority)

Conclusion

- we can perform performance measurements, and preserve privacy
  - in this solution, **no-one** obtains any individual performance measurements!!!
  - only aggregated performance measures are created
- A little more care is needed
  - what about lost packets?
  - see the paper for the solution!

THE UNIVERSITY OF ADELAIDE AUSTRALIA

Bonus slides

But wait...

What happens when packet are lost?

- we can't compute $\bar{D}_i^{\text{out}}$ without censoring the transmit times for the lost packets
- we can't tell other ISPs when packet are lost
  - this would reveal a great deal about performance
- we can't include straight sequence numbers in packets
  - these would allow statistical inference

Secure Dot Product (SDP) [11]

THE UNIVERSITY OF ADELAIDE AUSTRALIA

- Alice has a vector $\mathbf{a}$, and Bob has a vector $\mathbf{b}$.
- They want to compute

$$\mathbf{a}.\mathbf{b} = \sum a_i b_i$$

  without revealing any a_i or b_i to each other
- can't just return $\mathbf{a} \cdot \mathbf{b}$ because some choices of $\mathbf{a}$ would reveal parts of $\mathbf{b}$.
- so split the solution

$$V_a + V_b = \mathbf{a} \cdot \mathbf{b}$$

  and return V_a to Alice and V_b to Bob.

Solution

- add a randomly chosen packet ID to each packet:
  - ID chosen randomly from $\{1, 2, \ldots, L\}$ where $L \geq K_{ij}, \forall i, j$
  - create Identity vectors (at receivers)

$$I_{ij}^{(k)} = \begin{cases} 1, & \text{if the packet with ID } k \text{ from } i \text{ to } j \text{ is received,} \\ 0, & \text{otherwise.} \end{cases}$$

- now the calculation is

$$\bar{D}_i^{\text{out}} = \frac{1}{M_i} \sum_{\substack{j=1 \\ j \neq i}}^{N} \left[\sum_{k=1}^{L} I_{ij}^{(k)} r_{ij}^{(k)} - \sum_{k=1}^{L} I_{ij}^{(k)} t_{ij}^{(k)} \right].$$

Solution

- $I_{ij}^{(k)} r_{ij}^{(k)}$ is known to each receiver j, and so the sum (over k) is easily performed, and we can compute the sum over j using a SDS as before
- the sum $\sum_{k=1}^{L} I_{ij}^{(k)} t_{ij}^{(k)}$ is a dot product, and so we use SDS to get two parts of this $s_{ij}^{(t)}$ and $s_{ij}^{(r)}$.
  - $s_{ij}^{(t)}$ goes to the transmitter, and so we can perform a standard sum over j on these
  - $s_{ij}^{(r)}$ goes to the receivers, so we sum using a SDS
- M_i, the total number of received packets (transmitted from i) can be computed using a SDS
- transmitter gets all the info. to compute $\overline{D}_i^{\text{out}}$

Preventing Collusion in SDS

- Assume party j and $j+2$ collude
  - They know at least s_j and s_{j+1}
  - $s_{j+1} - s_j \bmod n = v_j$
  - so they can learn the value of j
- Various methods of prevention, e.g.
  - divide v_i randomly into shares v_{im} such that

$$\sum_m v_{im} = v_i$$

  - sum over i in a different order for each m.

$$\sum_{i=1}^{N} v_{im} = V_m$$

  - sum V_m normally $V = \sum_m V_m$

Millionaire problem

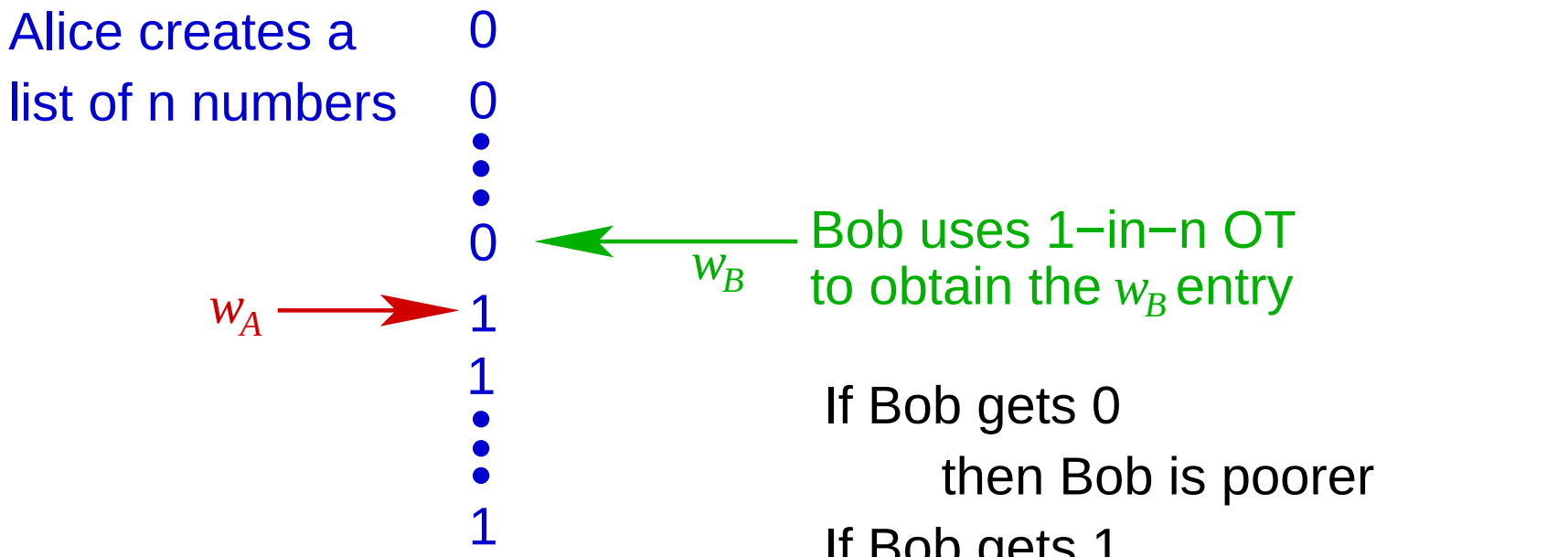
- Bill Gates and Warren Buffet are trying to decide who should put more money into the Gates foundation $(*)$
  - they want to know who is richer
- But they are feeling rather secretive, and don't want to reveal their true wealth.
- how can they decide?

Oblivious transfer [4, 5]

- there are various versions
- consider 1-in-n Oblivious Transfer (OT)
  - Alice has a list of numbers $\{a_1, a_2, \ldots, a_n\}$
  - Bob has an index β
  - Bob wants to learn a_β
  - Alice must not learn β, and Bob must not learn a_i for any $i \neq \beta$.
- Bob learns exactly one item from Alice's list, without Alice learning which item Bob discovered.

Applications

- the millionaires problem
  - more generically: calculating a minimum
- Assume Alice has wealth $w_A \in [1,n]$, and Bob has $w_B \in [1,n]$, where n is known to both



OT – how it works

THE UNIVERSITY OF ADELAIDE AUSTRALIA

1-in-2 Oblivious Transfer

- Alice has a pair of bits (a_0, a_1), and Bob has β
- trapdoor permutation f
  - Given key k, can choose permutation pair (f_k, f_k^{-1})
  - Given f_k it is hard to find f_k^{-1}
  - Easy to choose random element from f_k's domain
- random Bit B_{f_k} is a poly.-time Boolean function
  - $B_{f_k} = 1$ for half of the objects in f_k's domain
    $B_{f_k} = 0$ for other half
  - no probabilistic polynomial time algorithm can make a guess for $B_{f_k}(x)$ that is correct with probability better than $1/2 + 1/\text{poly}(k)$

1-in-2 Oblivious Transfer

- A randomly chooses (f_k, f_k^{-1}), and tells f_k to B
- B randomly chooses x_0 and x_1 in f_k's domain, and computes $f_k(x_i)$
- B sends A the pair

$$(u,v) = \begin{cases} (f_k(x_0), x_1), & \text{if } \beta = 0 \\ (x_0, f_k(x_1)), & \text{if } \beta = 1 \end{cases}$$

- A computes $(c_0, c_1) = \left(B_{f_k}(f_k^{-1}(u), f_k^{-1}(v))\right)$
- A sets $d_i = a_i \text{ xor } c_i$ and sends (d_0, d_1) to B
- B computes $a_\beta = d_\beta \text{ xor } B_{f_k}(x_\beta)$

http://www.cs.ut.ee/~lipmaa/crypto/link/protocols/oblivious.php

SDP – how it works

(1) A and B agree on two numbers m and n

(2) A finds m random vectors $\mathbf{t}_i$ such that

$$\mathbf{a}_1 + \mathbf{a}_2 + \ldots + \mathbf{a}_m = \mathbf{a}$$

B finds m random numbers $r_1, r_2, \ldots, r_m$.

(3) for $i = 1$ to m

(3a) A sends B n different vectors:

$$\{\mathbf{a}_i^{(1)}, \mathbf{a}_i^{(2)}, \ldots, \mathbf{a}_i^{(n)}\}$$

where exactly one $\mathbf{a}_i^{(q)} = \mathbf{a}_i$, the other $n-1$ vectors are random

(3b) B computes $\mathbf{a}_i^{(j)} \cdot \mathbf{b} - r_i$

(3c) A uses 1-in-n OT to retrieve

$$v_i = \mathbf{a}_i^{(q)} \cdot \mathbf{b} - r_i = \mathbf{a}_i \cdot \mathbf{b} - r_i.$$

(4) B computes $V_b = \sum_{i=1}^{m} r_i$

(5) A computes

$$V_a = \sum_{i=1}^{m} v_i = \sum_{i=1}^{m} \mathbf{a}_i \cdot \mathbf{b} - r_i = \mathbf{a} \cdot \mathbf{b} - V_b.$$

SDS and Sketches

THE UNIVERSITY OF ADELAIDE AUSTRALIA

Could apply this approach to many sources of data

- number of routers, number of links, or number of links of each type (e.g. OC48, Gig-Ethernet)
- kilometres of fiber, bandwidth-miles of network capacity,
- traffic-miles for carried traffic,
- detailed traffic data (e.g. netflow)
- performance data (packet loss, delay, reordering, ...)

Lots of sorts of data, and in particular for complex data (traffic) the dimensionality of dataset could be very high.

SDS and Sketches

Sketches [9] are an approach to reduce dimensionality of streaming datasets, e.g. Count-Min sketch [10]

- **Data:** a stream of updates (a,u), where $a \in \{1,\ldots,n\}$ is a key, and $u \in \mathbb{R}$ a value.
- **Signal:** a vector $v \in \mathbb{R}^n$, where for each update (a,u), we perform $v_a \mathrel{+}= u$.
- **Sketch:** consists of a $d \times w$ array of counts: $c[1,1] \ldots c[d,w]$, and d random hash functions $h_1, \cdots, h_d : \{1 \cdots n\} \rightarrow \{1 \cdots w\}$, for $w \ll n$
- **Update:** When an update (a,u) arrives, update $c[i,h_i(a)] \mathrel{+}= u$ for all $1 \leq i \leq d$.
- **Query:** When a point query $Q(a)$ arrives, an approximation of v_a is given by $\hat{v}_a = \min_i c[i,h_i(a)]$.

SDS and Sketches

Its almost trivial to extend SDS to sketches:

- agree on common hash functions (and array sizes)
- compute a sketch locally at each party
- use SDS to sum each element in the array
  - the point is that given K updates $\{(a_i^{(n)}, u_i^{(n)})\}_{i=1}^{K}$ from party n

$$\text{Sketch}\left(\cup_{n=1}^{N}\{(a_i^{(n)}, u_i^{(n)})\}_{i=1}^{K}\right) = \sum_{n=1}^{N}\text{Sketch}\left(\{(a_i^{(n)}, u_i^{(n)})\}_{i=1}^{K}\right)$$

- we can use the final sketch as needed, e.g. in anomaly detection

References

[1] "Data-mining moratorium act of 2003." Introduced in Senate of the United States in January 2003. http://thomas.loc.gov/cgi-bin/query/z?c108:S.188:.

[2] A. M. Odlyzko, "Internet traffic growth: Sources and implications," in Optical Transmission Systems and Equipment for WDM Networking II (B. B. Dingel, W. Weiershausen, A. K. Dutta, and K.-I. Sato, eds.), vol. 5247, pp. 1–15, Proc. SPIE, 2003.

[3] A. Yao, "Protocols for secure computations," in Proc. of the 23th IEEE Symposium on Foundations of Computer Science (FOCS), pp. 160–164, 1982.

[4] H. Lipmaa, "Oblivious transfer or private information retrieval." http://www.cs.ut.ee/~lipmaa/crypto/link/protocols/oblivious.php.

[5] B. Pinkas, "Oblivious transfer." http://www.pinkas.net/ot.html.

[6] C. Clifton, M. Kantarcioglu, J. Vaidya, X. Lin, and M. Zhu, "Tools for privacy preserving distributed data mining," SIGKDD Explorations, vol. 4, December 2002.

[7] Y. Lindell and B. Pinkas, "Privacy preserving data mining," Journal of Cryptology, vol. 15, no. 3, 2002.

[8] "Internet Activity, Australia." http://www.abs.gov.au/Ausstats/abs@.nsf/0/6445f12663006b83ca256a150079564d?OpenDocument, 2005.

[9] S. Muthukrishnan, "Data streams: Algorithms and applications," 2003. Manuscript based on invited talk from 14th SODA. Available from http://www.cs.rutgers.edu/~muthu/stream-1-1.ps.

[10] G. Cormode and S. Muthukrishnan, "An improved data stream summary: The count-min sketch and its applications," Proceedings of Latin American Theoretical Informatics (LATIN), pp. 29–38, 2004.

[11] W. Du and M. J. Atallah, "Privacy-preserving cooperative statistical analysis," in Proc. of the Annual Computer Security Applications Conference (ACSAC'2001), (New Orleans, LA, USA), December 2001.