Veracity: Practical Secure Network Coordinates via Vote-Based Agreements

Micah Sherr, Matt Blaze, and Boon Thau Loo
University of Pennsylvania

USENIX Technical
June 18th, 2009

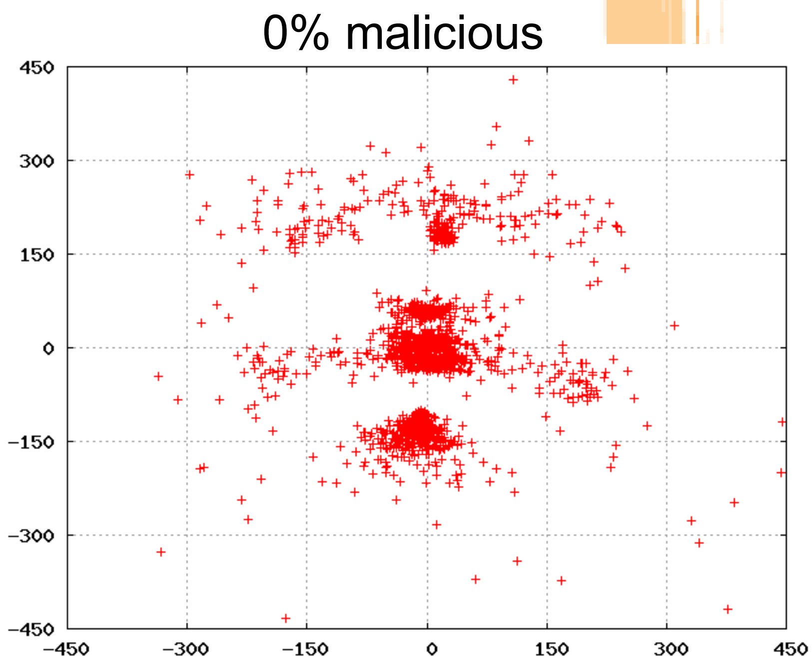
Network Coordinate Systems

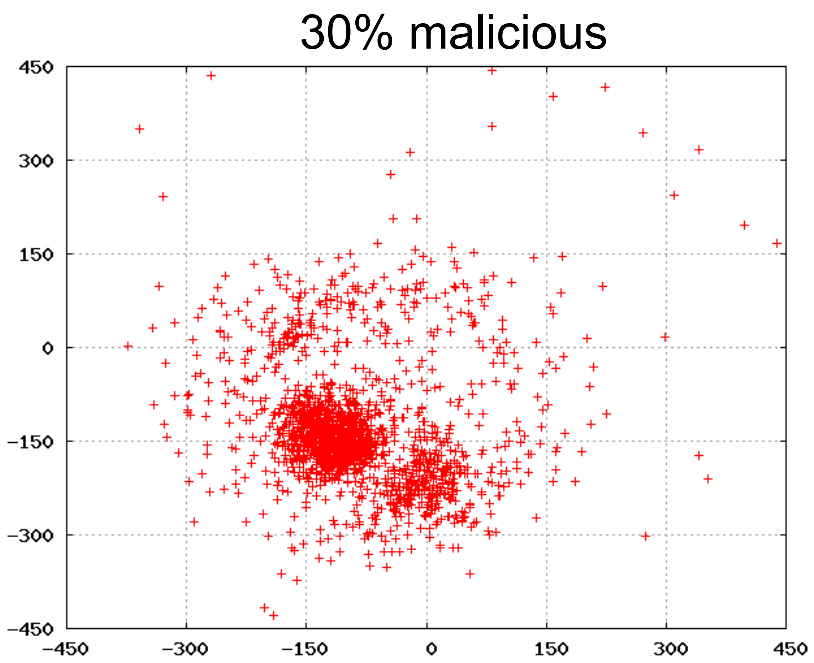
- Network coordinate systems enable efficient network distance estimations without requiring pairwise measurements
- Coordinate system maps nodes to n-dimensional coordinates
- Distance between two peers' coordinates represents actual network distance (e.g., RTT) between them

Applications

- Support wide range of network services:
  - Proximity-based routing
  - Neighbor selection in overlays
  - Network-aware overlays
  - Replica placement
  - Anonymous path selection
  - Detour routing
- E.g., Vuze BitTorrent client maintains million+ node coordinate system for efficient DHT traversal

Vulnerability to Attack





- Distributed coordinate systems easy to manipulate
  - 10% malicious nodes → 4.9X decrease in accuracy
  - 30% malicious nodes → 11X decrease in accuracy

Veracity

- Security protection layer for coordinate systems
  - Lightweight
  - No *a priori* trust required
  - Amenable to realistic network conditions
  - Fully distributed

- Intuition: Truthfulness of coordinates can be accurately assessed by independent peers with different vantage points

Related Work

	Assumes no TIVs	Fully distributed (no a priori trusted nodes or PKI)	Supports dynamic neighborsets	Does not depend on temporal or spatial locality heuristics
PIC	✗	✓	✓	✓
Secure coordinates [Kaafar et al.]	✓	✗	✓	✓
RVivaldi	✓	✗	✓	✓
Zage et al. CCS07	✓	✓	✗	✗
Veracity	✓	✓	✓	✓

Coordinate Systems 101

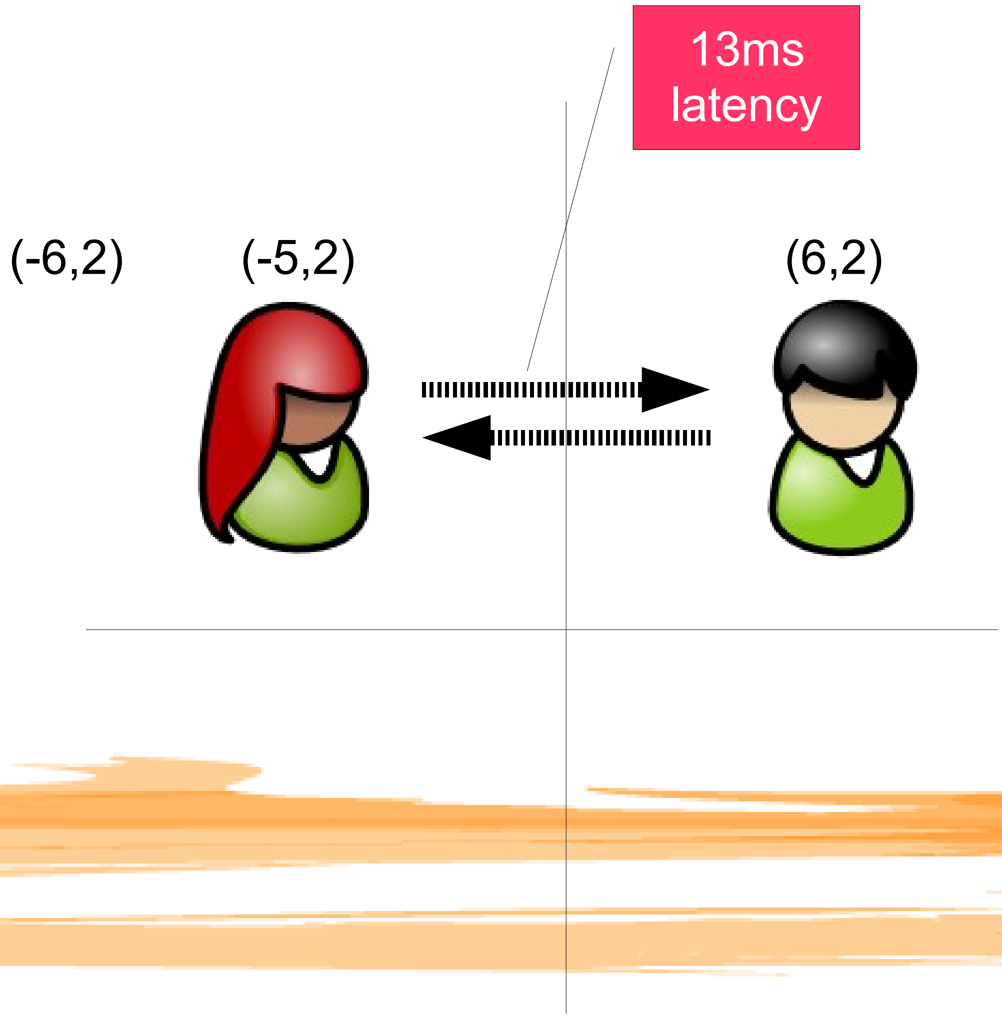
- Many flavors: Vivaldi, PIC, etc.
- Iterative update mechanism:

–Node retrieves coordinate of random neighbor

–Node measures metric between itself and neighbor

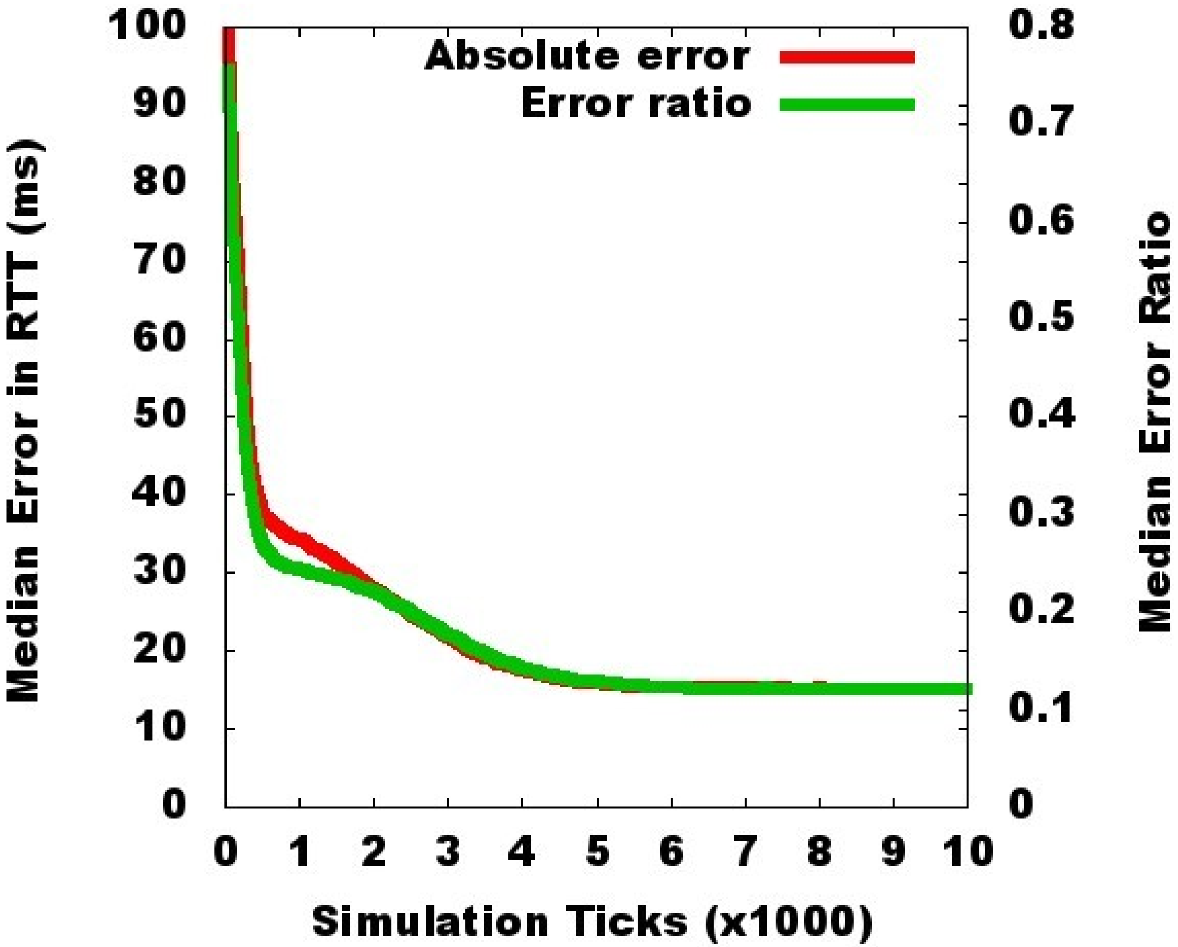
–Updates local coordinate to minimize error function

- Embedding errors due to network triangle-inequality violations (TIVs)



Coordinate Systems 101

- Embedding errors due to network triangle-inequality violations (TIVs)
- **Median error ratio:** median of percentage difference between virtual and real distances between a node and all other nodes



Attacking Virtual Coordinate Systems

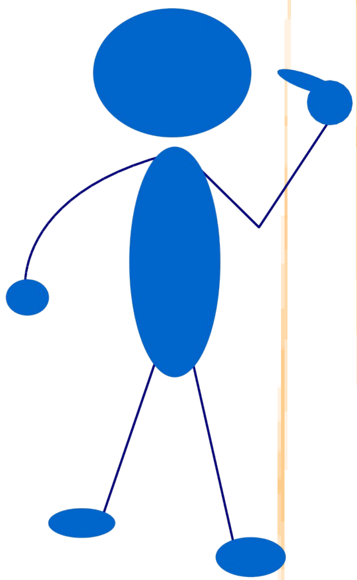
- **Disorder attacks:** decrease accuracy (and utility) of coordinate system
- Attack techniques:
  - When queried, provide false coordinate
  - When probed, delay measurement response
- Possible attack implications:
  - Malicious hosts selected for routes, neighbors, or replicas
  - Requests misrouted; false data returned in CDNs
  - Partitioned DHTs

Veracity:

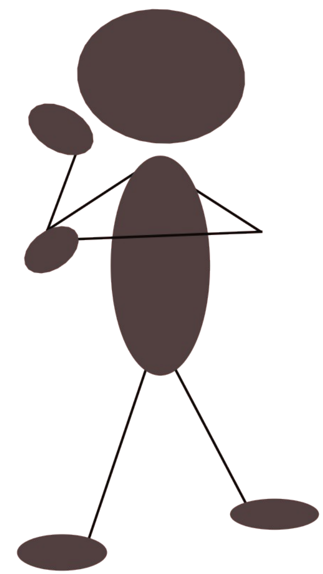
A security layer that protects
the accuracy of coordinate systems

Veracity Participants

Publisher



Investigator



Advertises coordinate.
May or may not be truthful

Wants to use Publisher's coordinate to update local coordinate

Node Discovery

- Fully-distributed *directory service* used to locate peers
- Distributed directory server (e.g., DHT) must support:

DELIVER(g,m) :	deliver message m to node whose globally unique identifier (GUID) is closest to g

- Each node calculates GUID as `HASH(ip|port)`

Veracity's Two Protection Phases

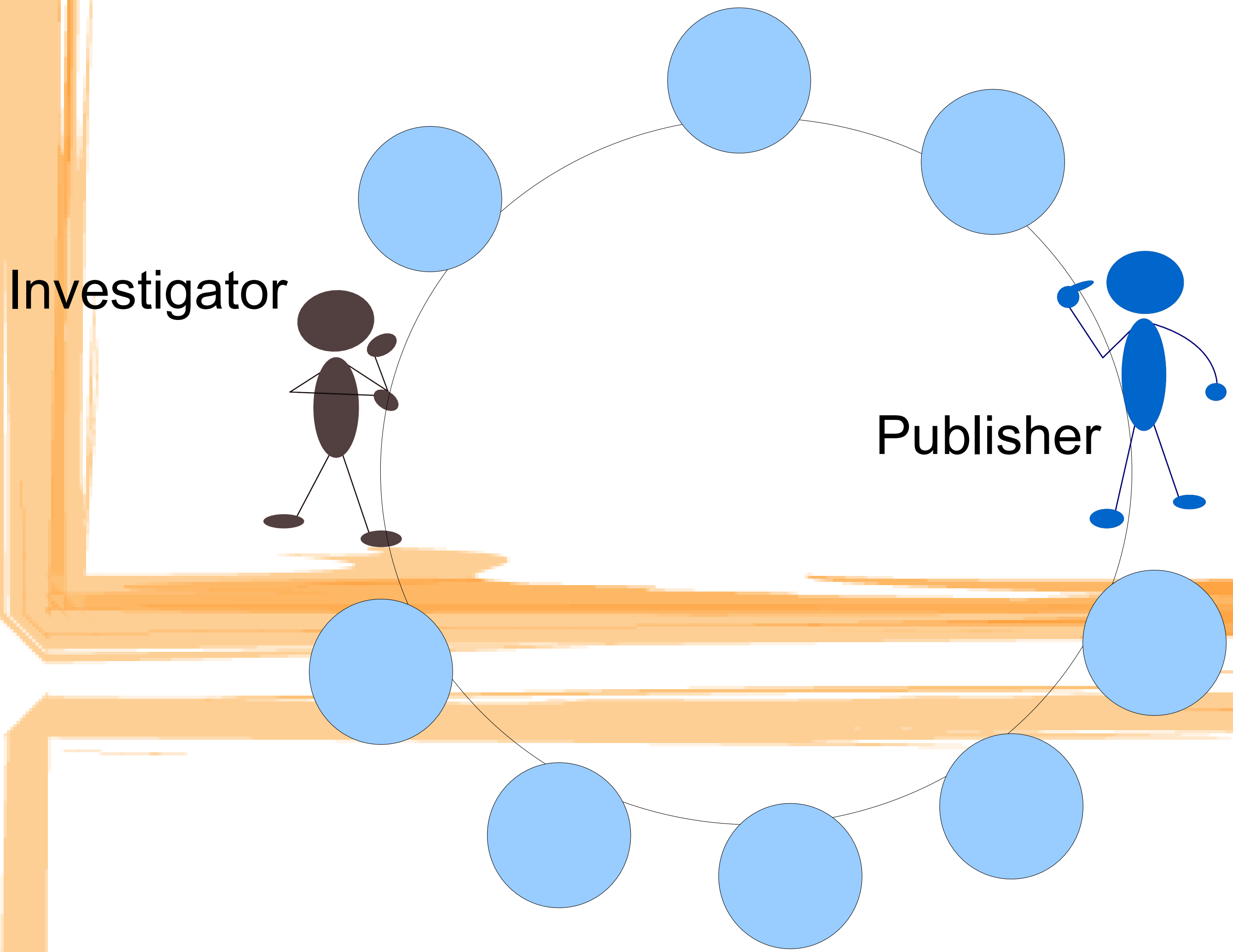
– **Phase I: Publisher coordinate verification**

Rejects inconsistent or inaccurate coordinates

– **Phase II: Candidate coordinate verification**

Prevents delayed measurements after coordinate passes publisher coordinate verification

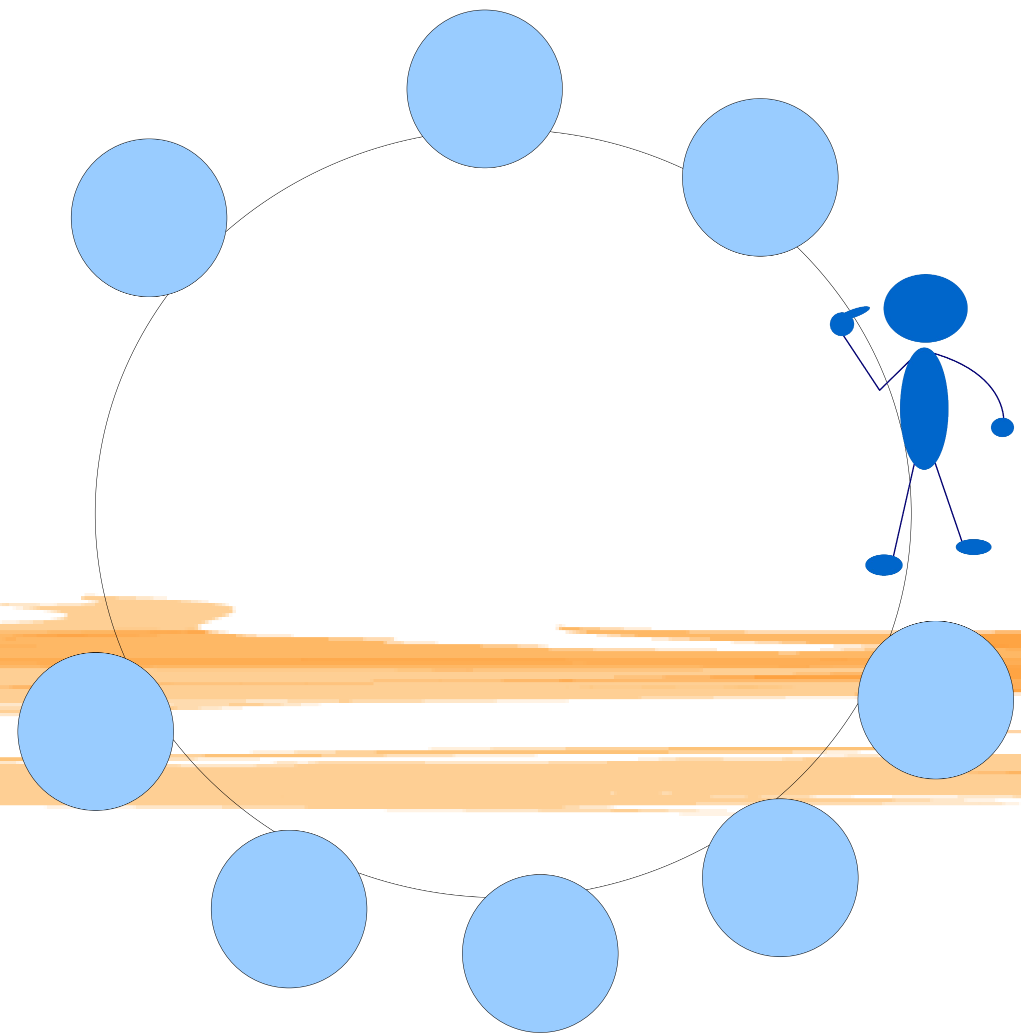
Publisher Coordinate Verification



Publisher Coordinate Verification:
Publisher notifies VSet of coordinate

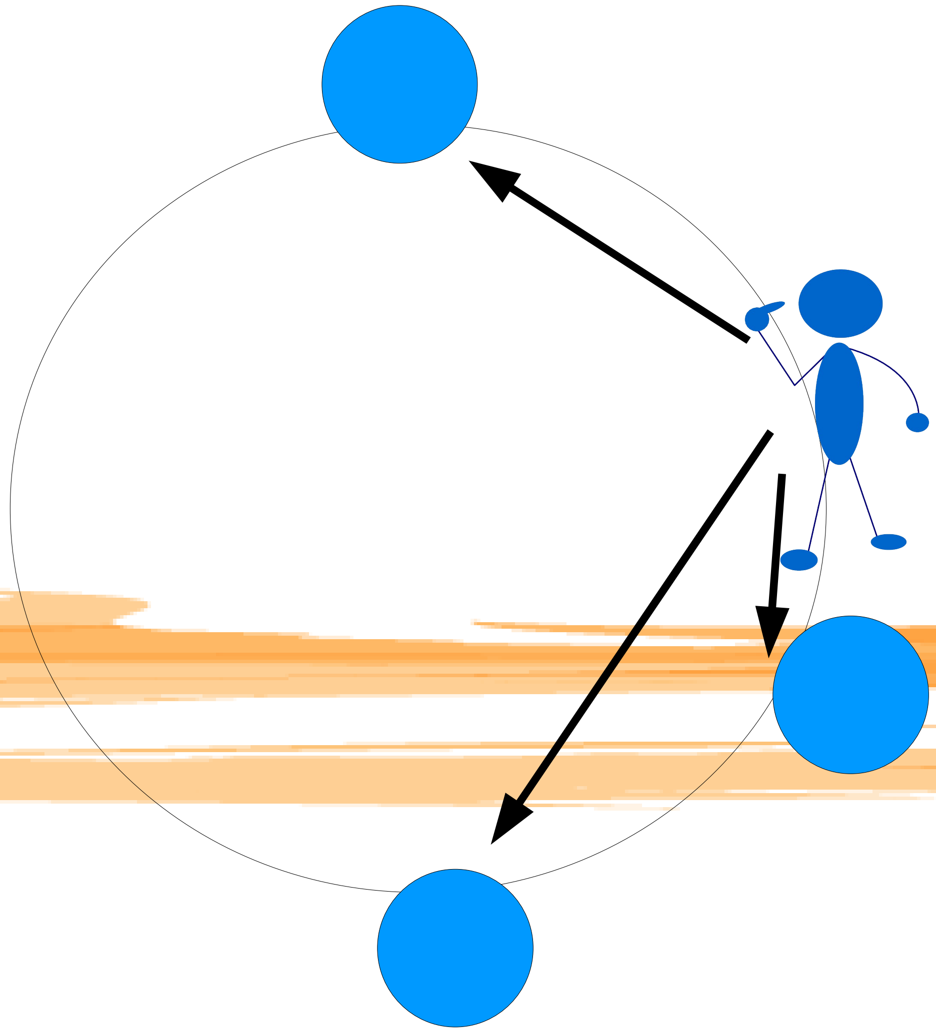
- Publisher updates his coordinate
- Step 1: Publisher computes his ***verification set (VSet)***, consisting of peers whose GUIDs are closest to $h_1, \ldots, h_r$ using the recurrence:

$$h_i = \begin{cases} \text{HASH}(g) & \text{if } i=1 \\ \text{HASH}(h_i\text{-}1) & \text{if } i>1 \end{cases}$$



Publisher Coordinate Verification:
Publisher notifies VSet of coordinate

- Step 2: Publisher sends its GUID *g* and new coordinate *C* to each VSet member via deliver



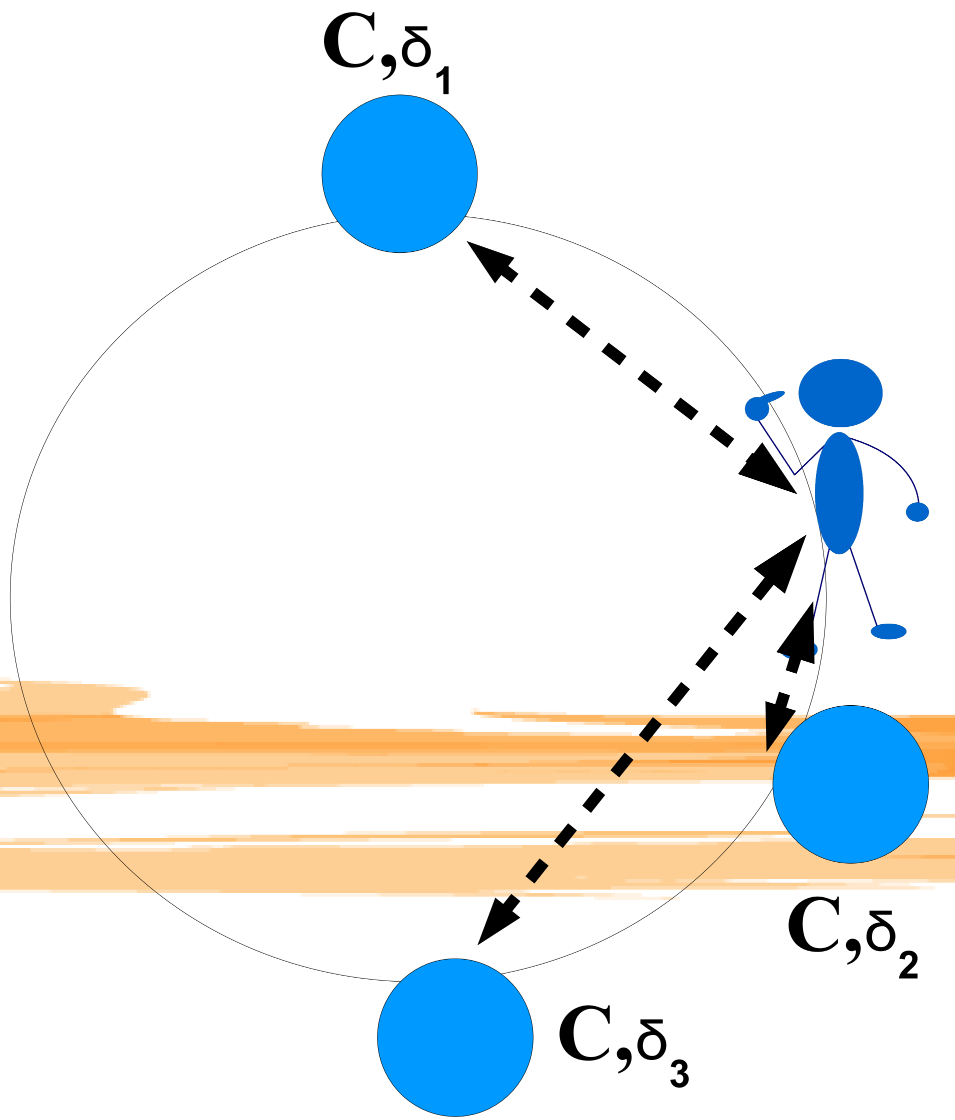
Publisher Coordinate Verification:
VSet members assess Publisher's coordinate

- Each VSet member measures the RTT between itself and Publisher
- Each computes the **error ratio:** the % difference between the empirical (RTT) and coordinate-based distances:

$$\delta_{(v_i,g)} = \frac{\left|RTT(v_i,g) - ||C - C_{v_i}||\right|}{RTT(v_i,g)}$$

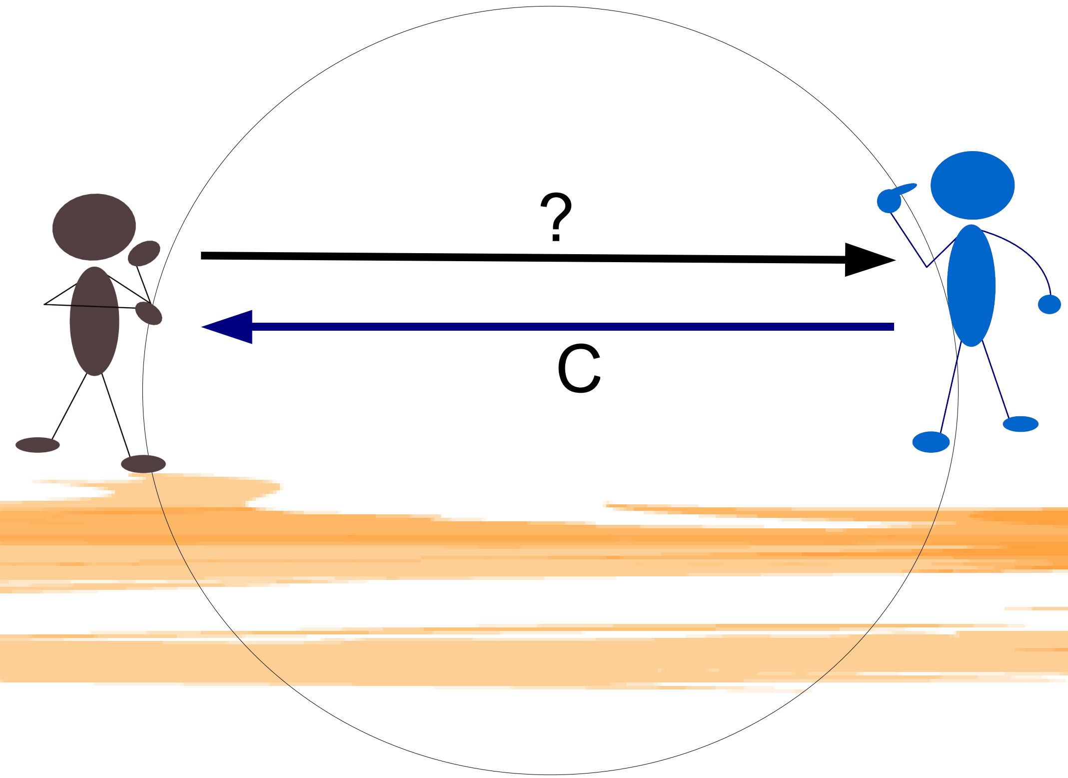
–indicates VSet member's belief in the publisher's advertised coordinate

- VSet members store Publisher's advertised coordinate and its error ratio as **evidence tuple**



Publisher Coordinate Verification:
Investigator queries Publisher for coordinate

- Investigator queries Publisher for its coordinate
- Publisher returns its coordinate C



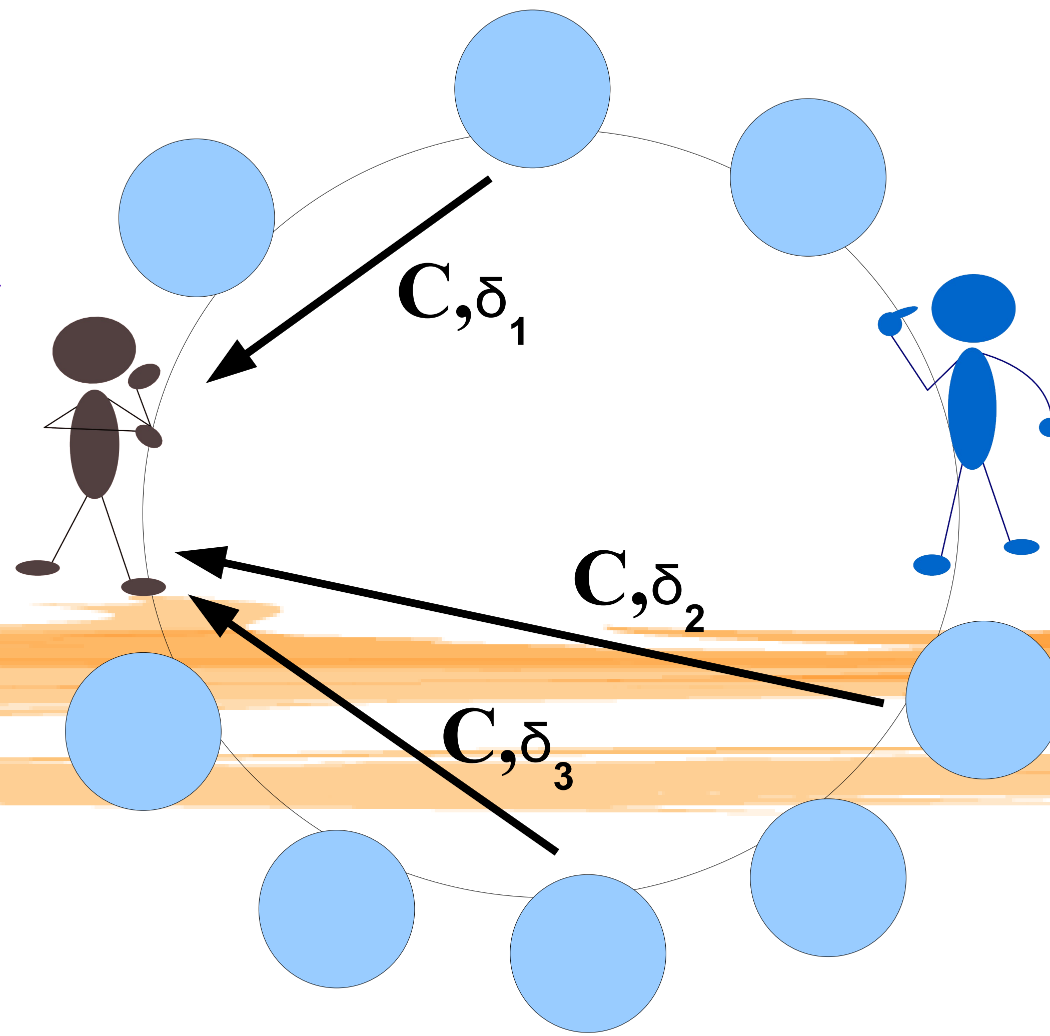
Publisher Coordinate Verification:
Investigator computes Publisher's VSet, requests evidence

- Investigator uses the same recurrence

$$h_i = \begin{cases} HASH(g) & \text{if } i=1 \\ HASH(h_i\text{-}1) & \text{if } i>1 \end{cases}$$

to compute Publisher's VSet

- Investigator requests *evidence tuple* from each VSet member
- Evidence tuples with incorrect coordinate are discarded

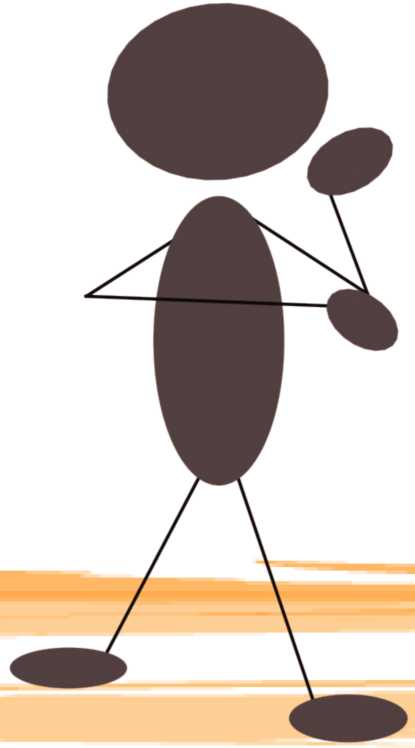


Publisher Coordinate Verification: Investigator considers evidence

- If the number of evidence tuples having

$\delta < \max\delta$

is at least R, then coordinate is *accepted.*

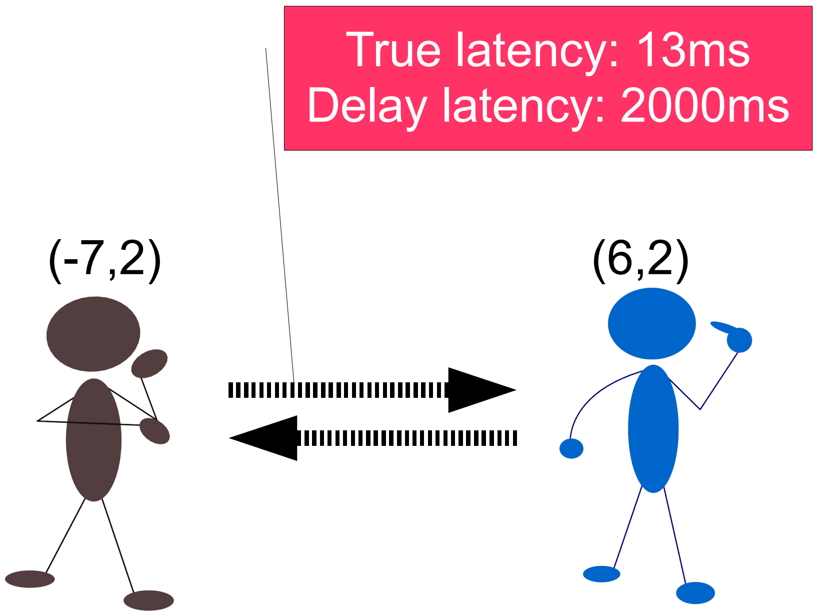


- Publisher Coordinate Verification ensures that:

–Publisher must advertise consistent coordinate to VSet members and Investigator

–Publisher's coordinate must match VSet members' empirically measured RTTs

(-934,2)



- But this is insufficient to protect a virtual coordinate system...

–Publisher behaves honestly, allowing coordinate to pass Publisher Coordinate Verification

–After verifying coordinate, Investigator measures RTT to Publisher

–Publisher delays Investigator's RTT probe

Candidate Coordinate Verification

- Investigator queries coordinates of random nodes (RSet)
- Conducts RTT measurement to each RSet member
- Computes new candidate coordinate C' using Publisher's verified coordinate
- Using current (C) and candidate coordinate (C'), computes error ratios E and E'

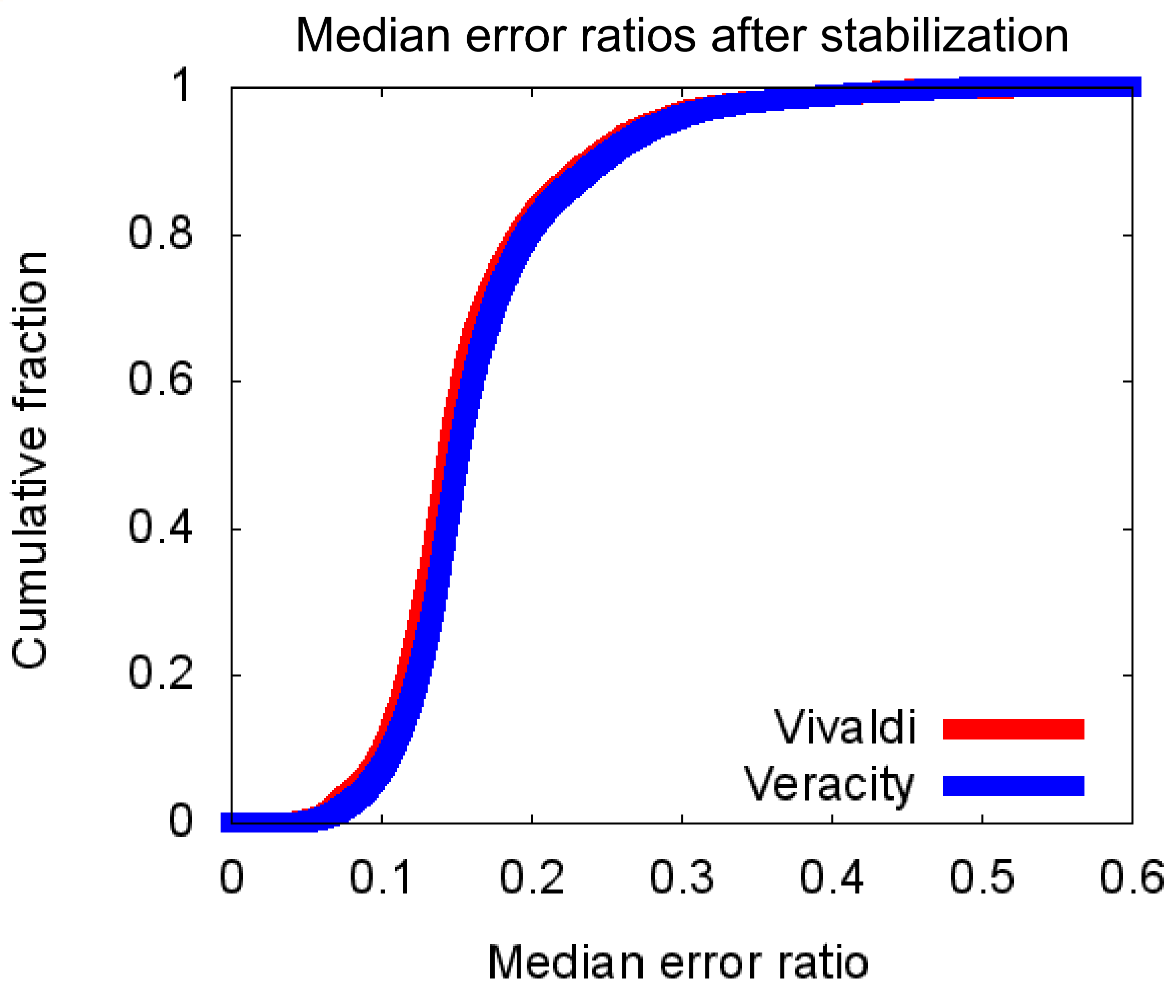
$$E = \sum_{R_i \in R} \frac{\left| RTT(I, R_i) - ||C - C_{R_i}|| \right|}{RTT(I, R_i)} \qquad E' = \sum_{R_i \in R} \frac{\left| RTT(I, R_i) - ||C' - C_{R_i}|| \right|}{RTT(I, R_i)}$$

- If $E' - E \leq \Delta$, Investigator replaces C with C'

Evaluation

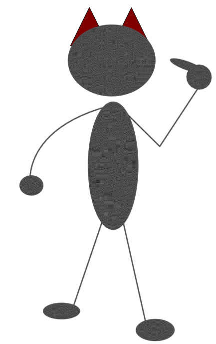
Accuracy in Absence of Attack

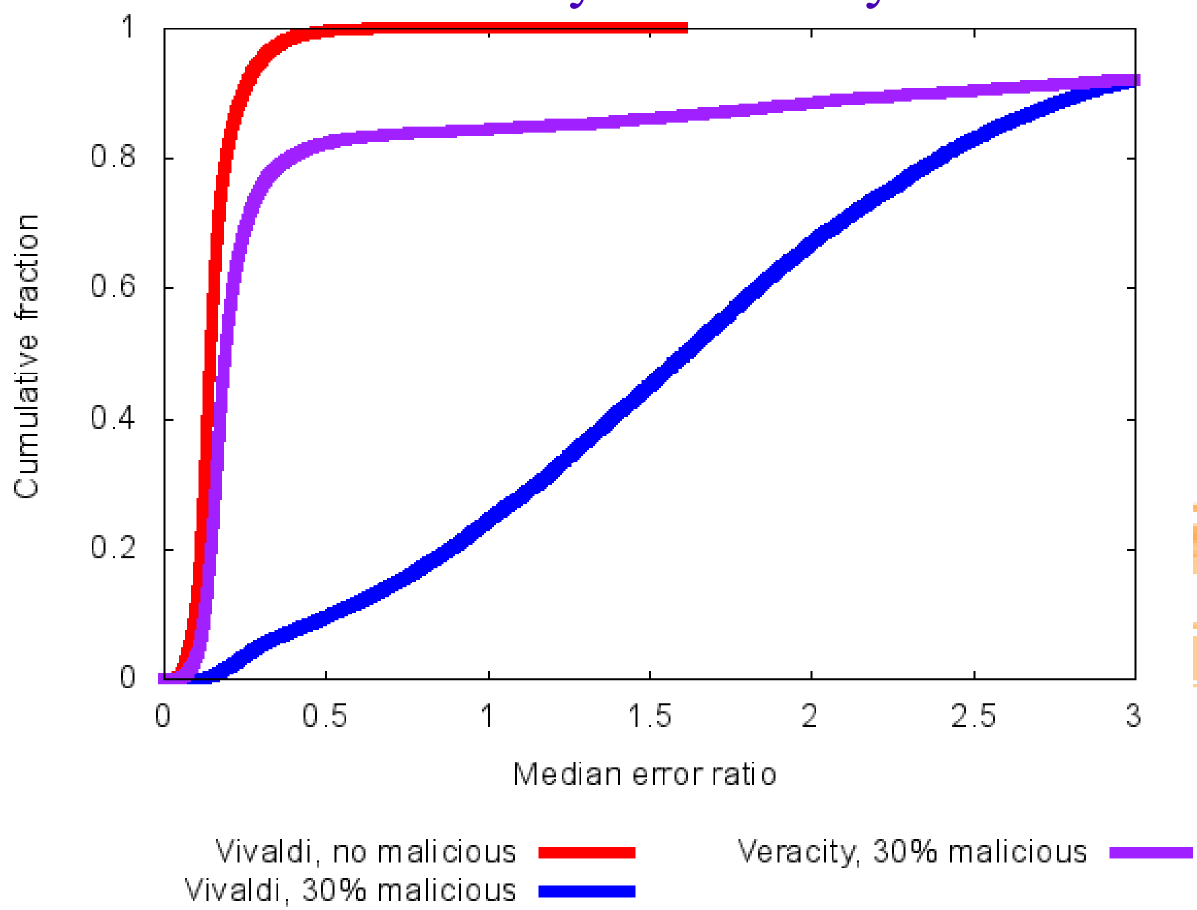
- Veracity functionality added to Bamboo DHT
- Median error ratios of 500 nodes from the King (pairwise latency) dataset
- **Veracity increases median of median error ratios by just 4.6% (0.79ms)**



Resilience to Naïve Attack

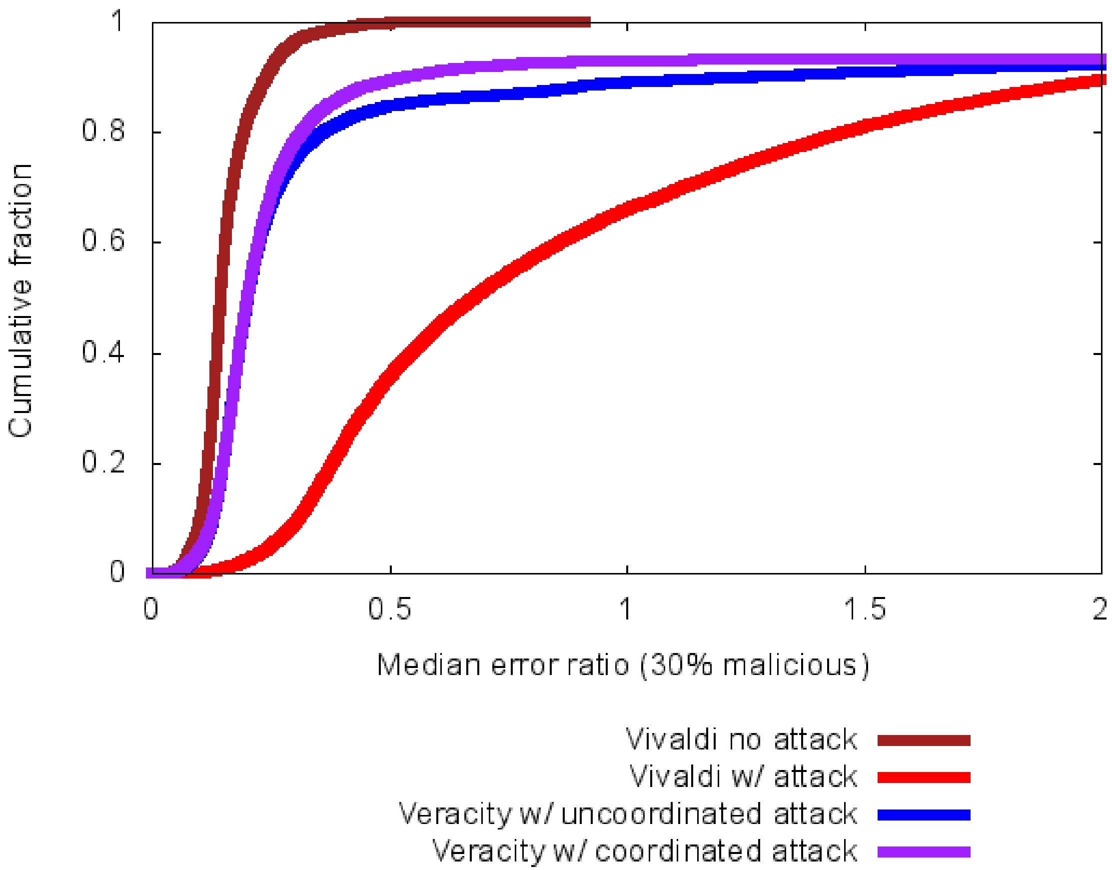
- Malicious nodes report <u>inconsistent</u> and random coordinates and delay RTT probes by up to 2000ms
  - Worst case for Vivaldi
  - Inconsistent coordinates easily detected by VSet





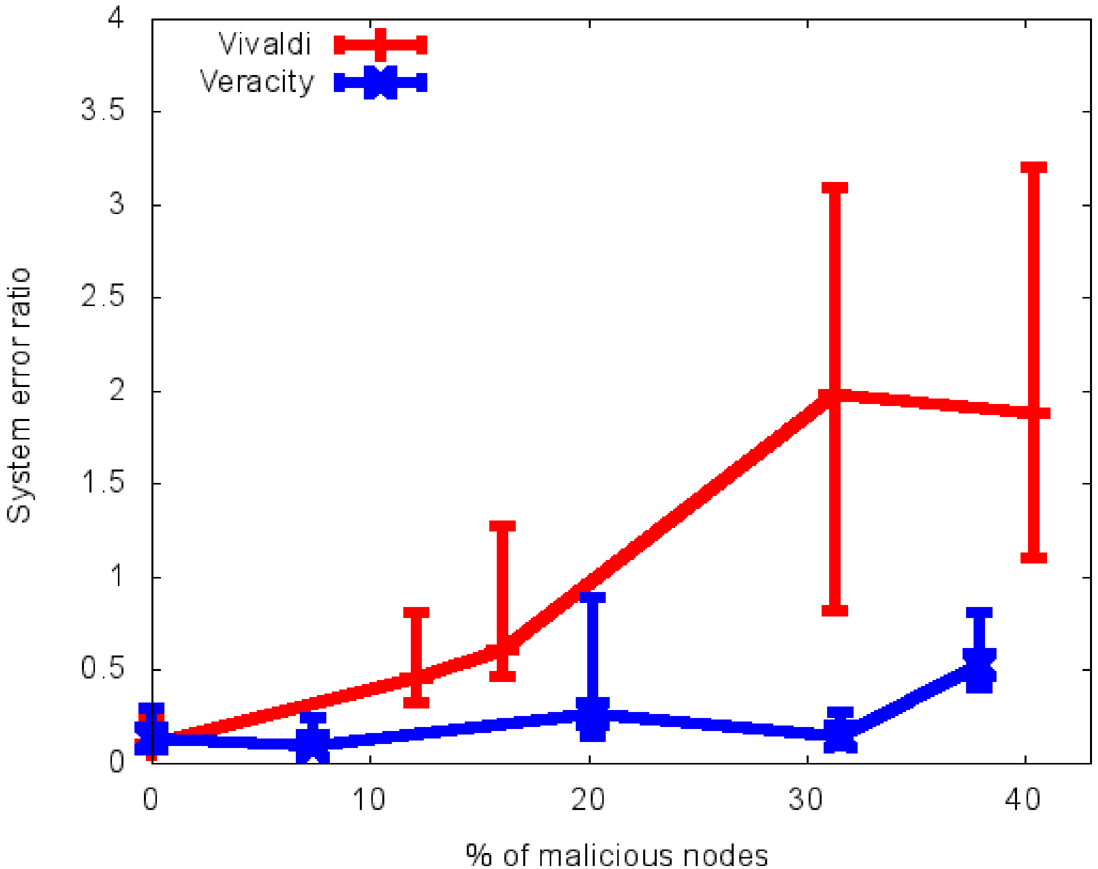
Resilience to Coordinated Attack

- Malicious nodes (30% of network) randomly delay RTT probes and advertise false coordinates
- Malicious nodes offer supporting evidence (low error ratios) for other malicious nodes, no evidence for honest nodes



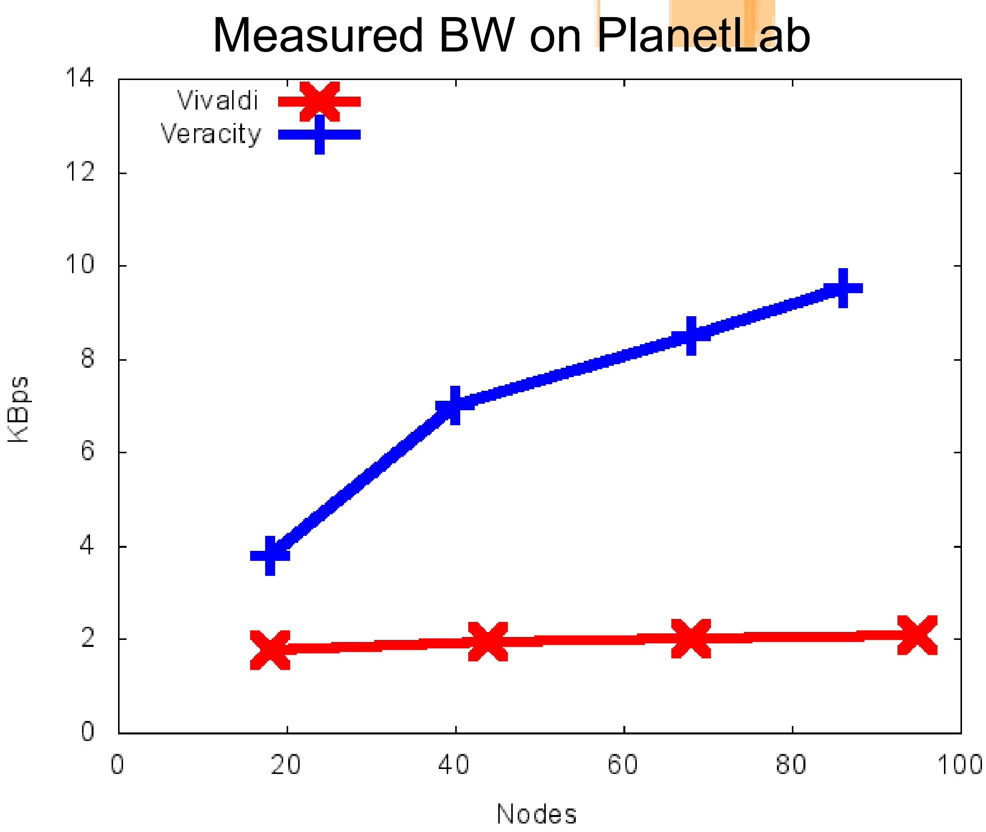
PlanetLab Deployment

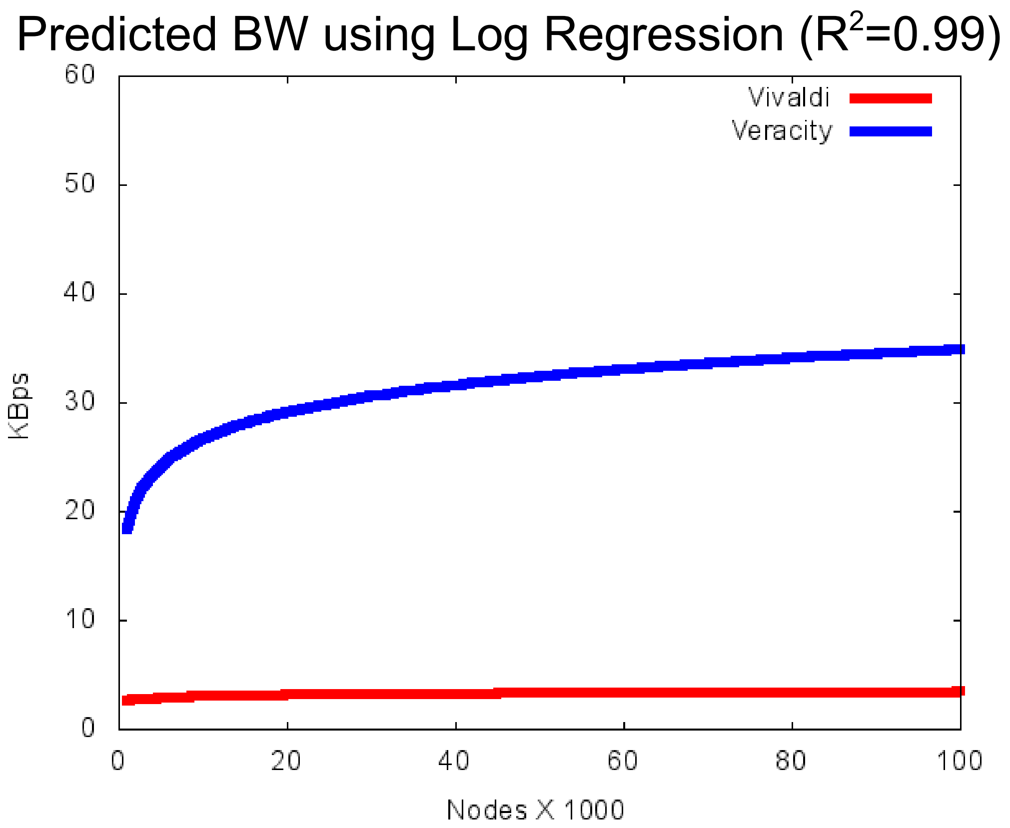
- Installed on ~100 geographically diverse PlanetLab nodes



Communication Cost

- Publisher Coordinate Verification and Candidate Coordinate Verification both impose linear communication overheads
- Cost of each **deliver** request is O(log N)





Summary

- Veracity effectively mitigates disorder attacks
  - Reduces Vivaldi's median error ratio by 88% when 30% of nodes are malicious and uncoordinated
  - Even against coordinated attacks, Veracity reduces Vivaldi's error ratio by 70% when 30% are malicious
- Unlike existing approaches, Veracity
  - Does not rely on TIV assumptions
  - Requires no centralized infrastructure
  - Does not require *a priori* trust
- Veracity incurs minimal communication overhead and can be practically deployed

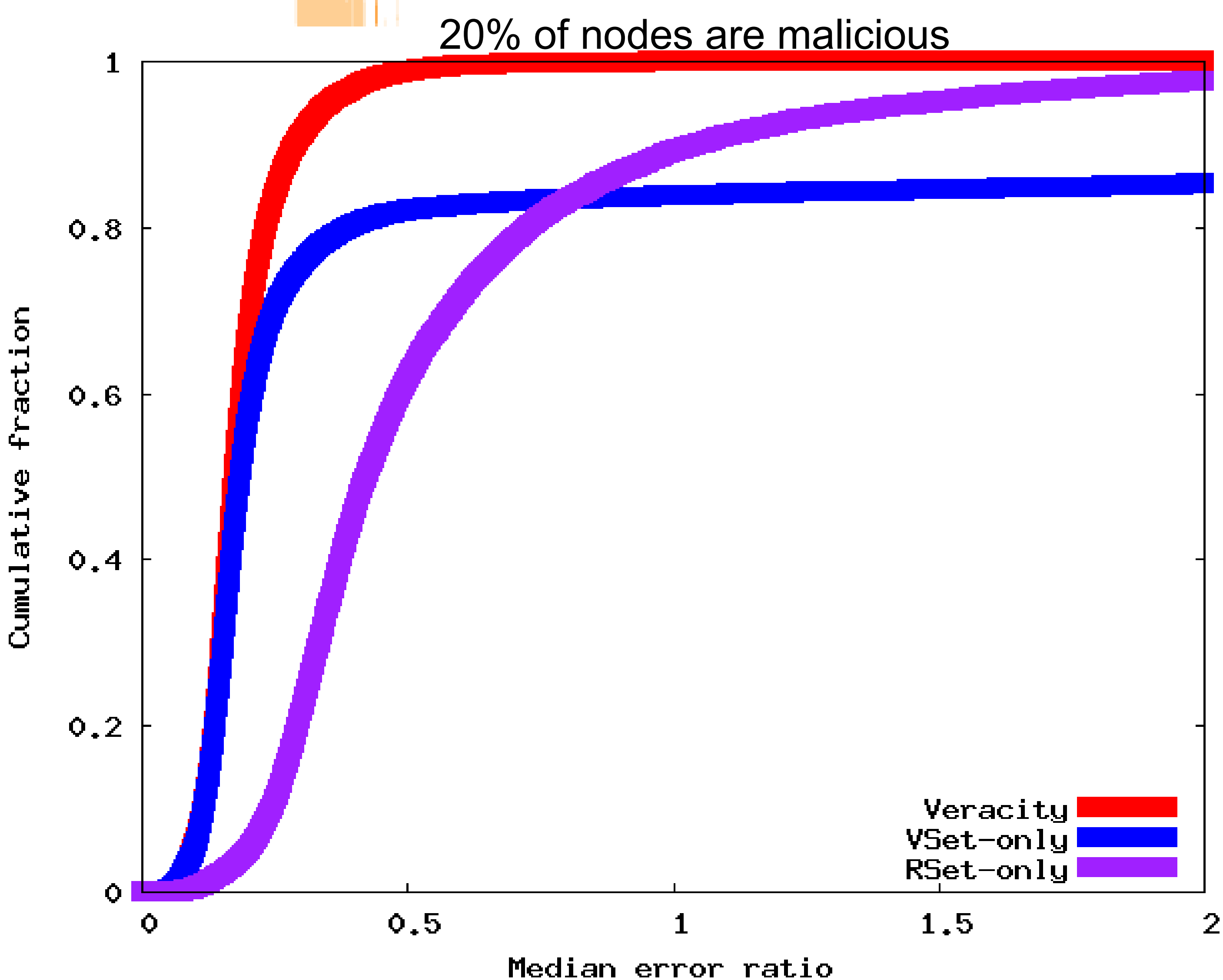
Veracity: Practical Secure Network Coordinates via Vote-Based Agreements

Micah Sherr, Matt Blaze, and Boon Thau Loo
University of Pennsylvania

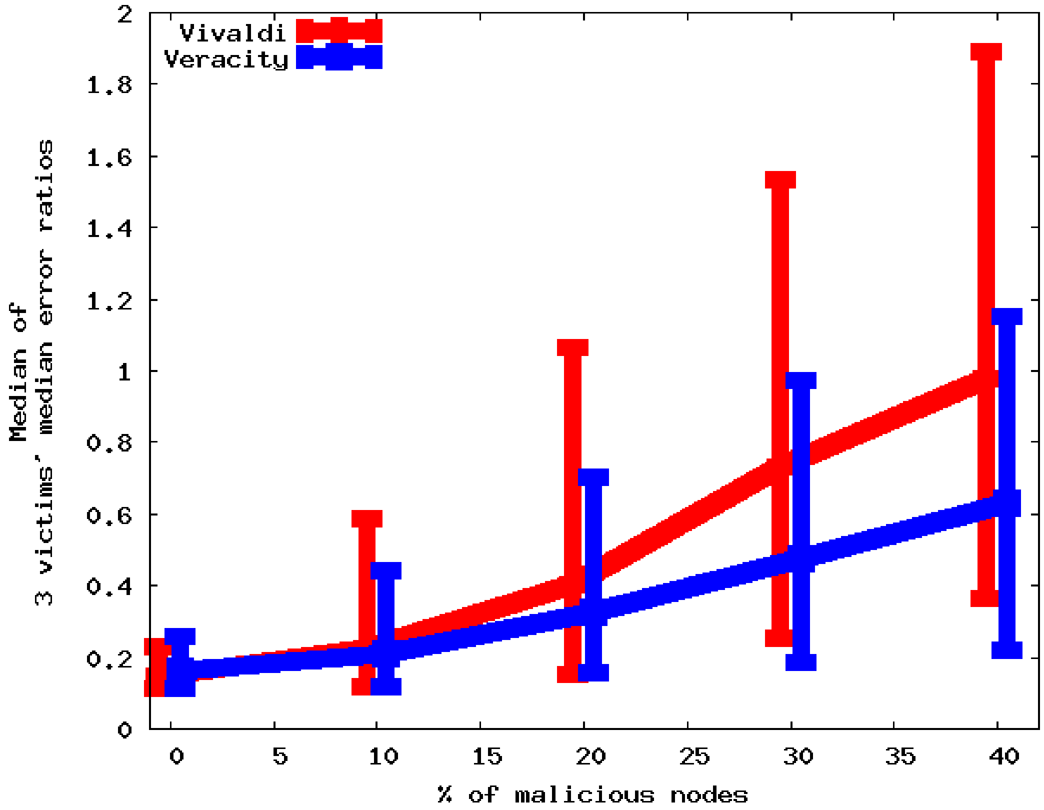
USENIX Technical
June 18th, 2009

Backup slides

Rejected: VSet-only and/or RSet-only Veracity



Resilience to Repulsion and Isolation Attacks



- Malicious nodes partitioned into 3 coalitions
- Each coalition attempts to move victim node to far coordinate (-1000 in all dimensions)

DHT Security

- Veracity relies on reliability of deliver requests
- DHT attacks:

–**Sybil**: register multiple identities to increase influence in network

–**Eclipse**: falsify routing update messages to corrupt DHT routing tables

–**Routing**: misroute or modify requests, or forge responses

DHT Security (2)

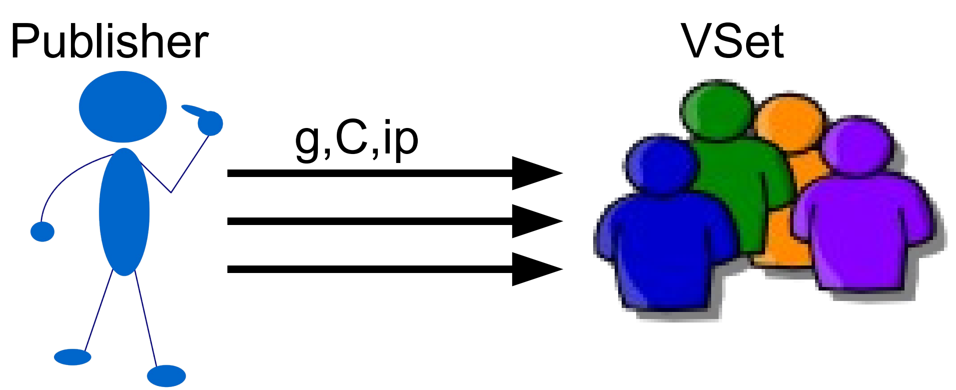
- Sybil attack countermeasures:
  - Distributed registration in which nodes vote on whether IP is allowed to join [Dinger'06]
  - Use bootstrap graphs to generate trust profiles [Danezis'05]
  - Cryptopuzzles [Borisov'06]
- Eclipse and Routing attack countermeasures:
  - Organize network into swarms; forward message only if lookup sent from majority of members of previous swarm [Fiat'05]
  - Send via redundant routes [Castro'02]

Publisher Coordinate Verification: Publisher notifies VSet of coordinate

- Each publisher assigned a *Verification Set* (VSet) of peers whose GUIDs are closest to $h_1, \ldots, h_r$ determined using the recurrence:

$$h_i = \begin{cases} HASH(g) & \text{if } i = 1 \\ HASH(h_{i-1}) & \text{if } i > 1 \end{cases}$$

- After updating his coordinate, publisher sends tuple to each VSet member via `deliver`



g – publisher's GUID

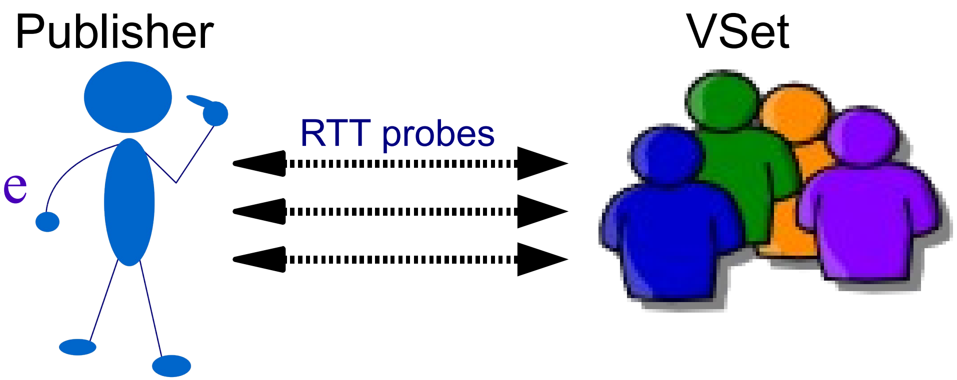
C – publisher's coordinate

ip – publisher's IP+port

Publisher Coordinate Verification: VSet members assess coordinate

- Each VSet member measures the RTT between itself and the publisher
- VSet members compute the *error ratio*:

$$\delta_{(v_i,g)} = \frac{\left| RTT(v_i, ip) - ||C - C_{v_i}|| \right|}{RTT(v_i, ip)}$$



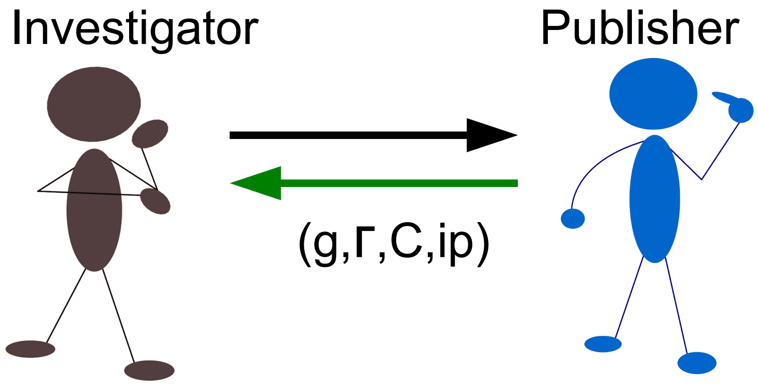
- Error ratio reflects percentage difference between real and estimated distances
- Indicates VSet member's belief in the publisher's advertised coordinate
- VSet member stores *evidence tuple* (g,C,ip,δ)

Publisher Coordinate Verification: Investigator queries Publisher for coordinate

Investigator queries Publisher for its coordinate.

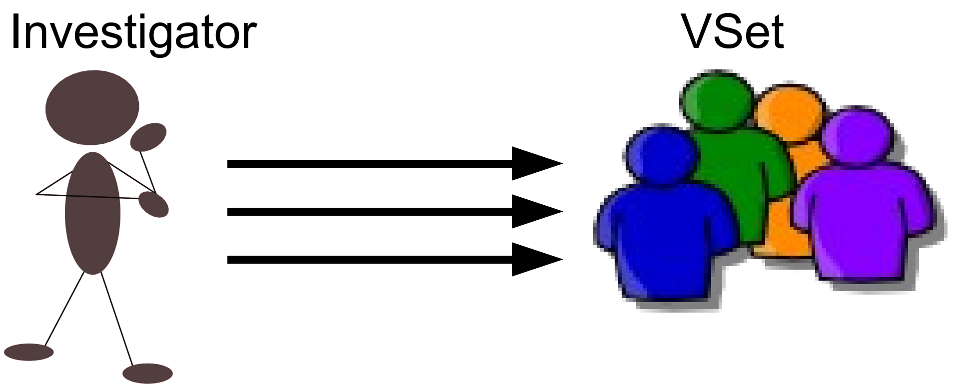
Publisher responds with *claim tuple:*

- g – publisher's GUID
- Γ – publisher's VSet size
- C – publisher's coordinate
- ip – publisher's network address



Publisher Coordinate Verification:
Investigator probes VSet for evidence

- Investigator calculates Publisher's VSet and queries each member for its evidence tuple



Publisher Coordinate Verification:
Investigator considers VSet evidence

- VSet members return evidence tuples to Investigator
- If the number of evidence tuples having $\delta < \max\delta$ is at least R, then coordinate is accepted