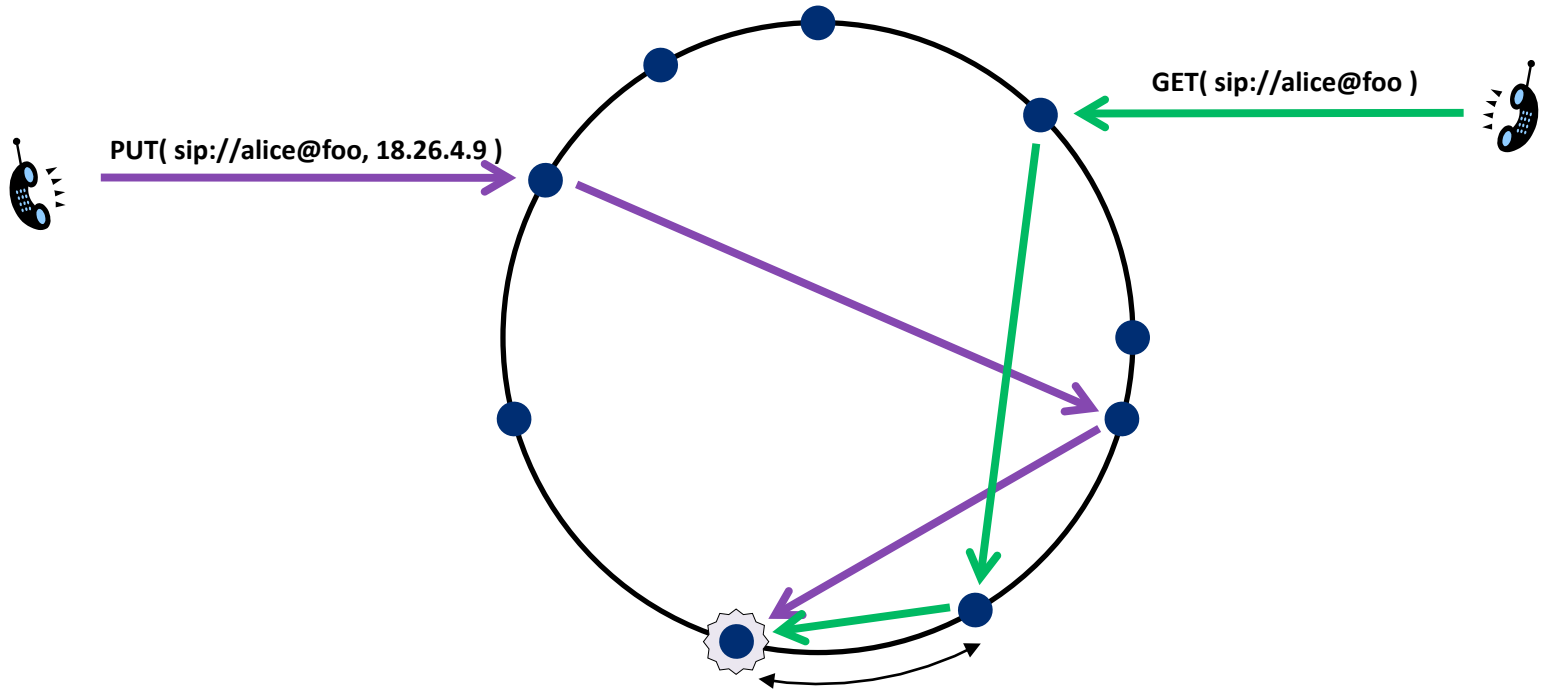
A Sybil-Proof Distributed Hash Table

Chris Lesniewski-Laas M. Frans Kaashoek
MIT

28 April 2010
NSDI
http://pdos.csail.mit.edu/whanau/slides.pptx

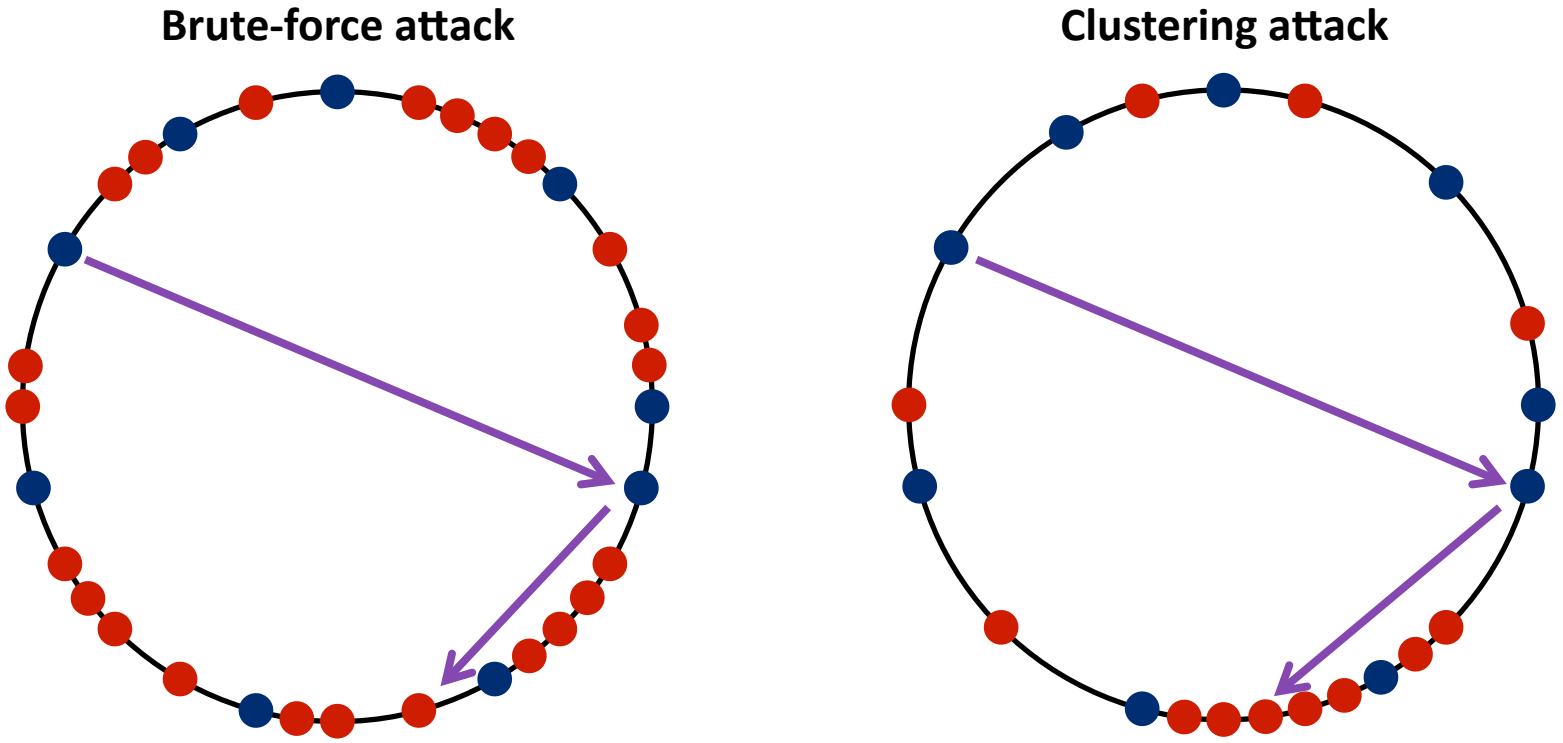
Distributed Hash Table

- Interface: PUT(*key*, *value*), GET(*key*)→*value*
- Route to peer responsible for key

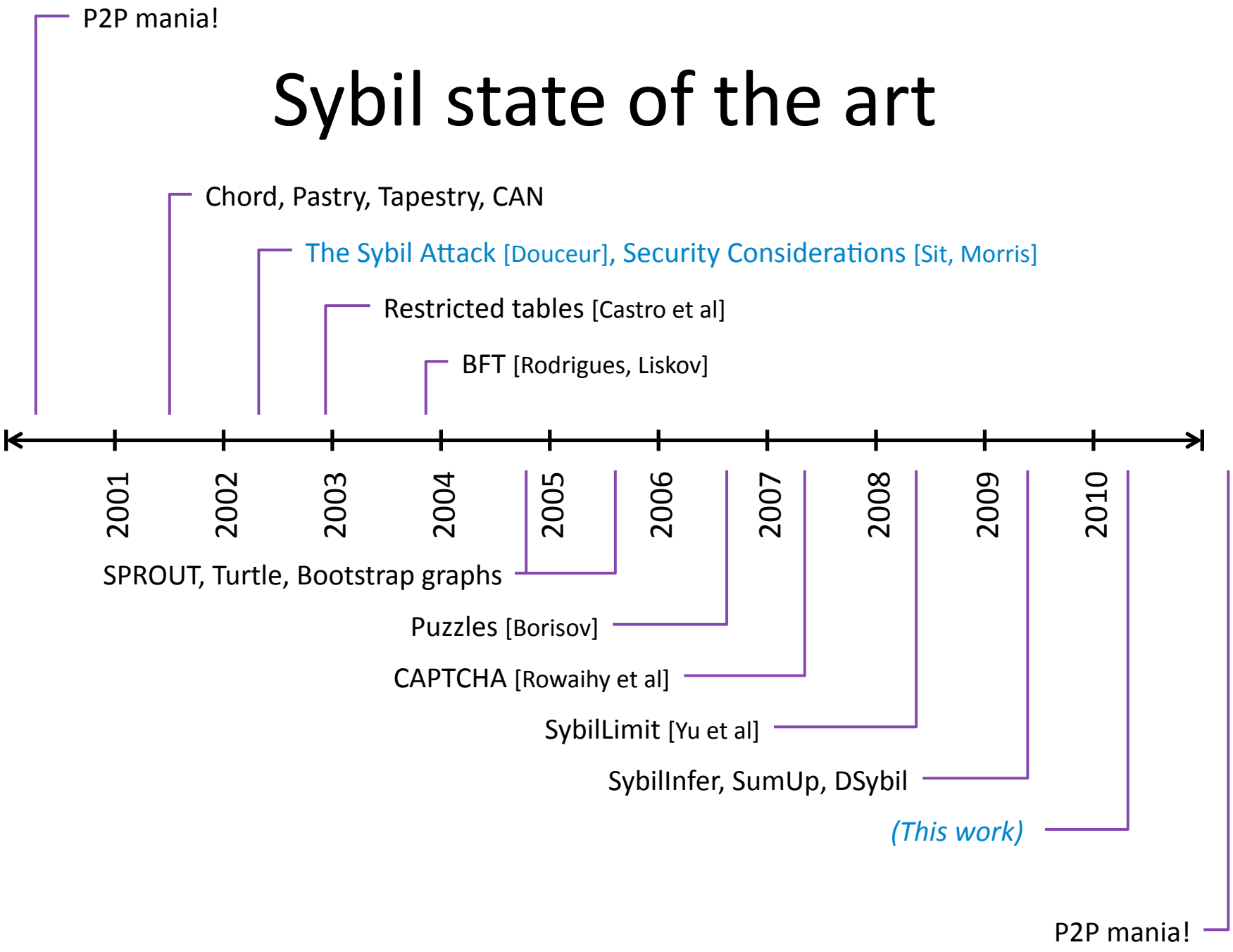


The Sybil attack on open DHTs

- Create many pseudonyms (Sybils), join DHT
- Sybils join the DHT as usual, disrupt routing



Sybil state of the art



Contribution

- Whānau: an efficient Sybil-proof DHT protocol
  - GET cost: O(1) messages, one RTT latency
  - Cost to build routing tables: O(√N log N) storage/ bandwidth per node (for N keys)
  - Oblivious to number of Sybils!

- Proof of correctness
- PlanetLab implementation
- Large-scale simulations vs. powerful attack

Division of labor

- Application provides **integrity**
- Whānau provides **availability**

- E.g., application signs values using private key
- Proc GET(*key*):

  Until valid *value* found:

  Try *value* = LOOKUP(key)

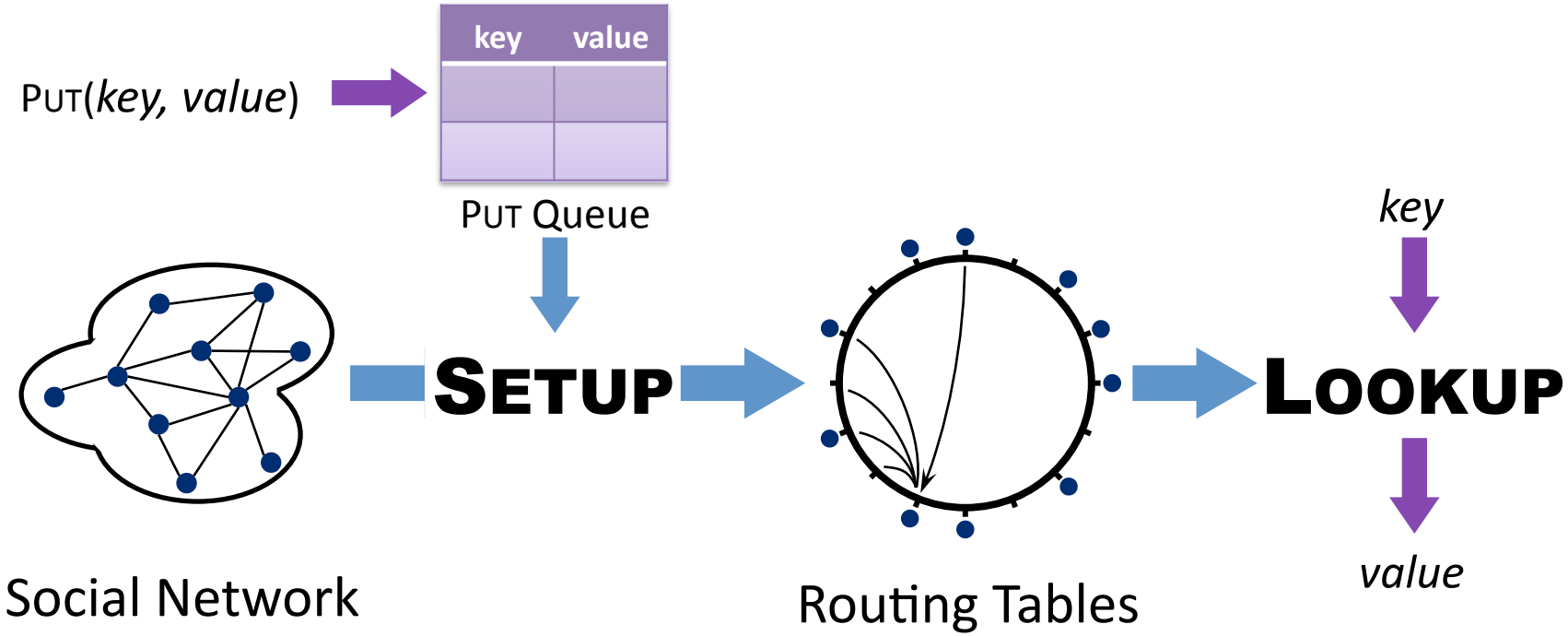
  Repeat

Approach

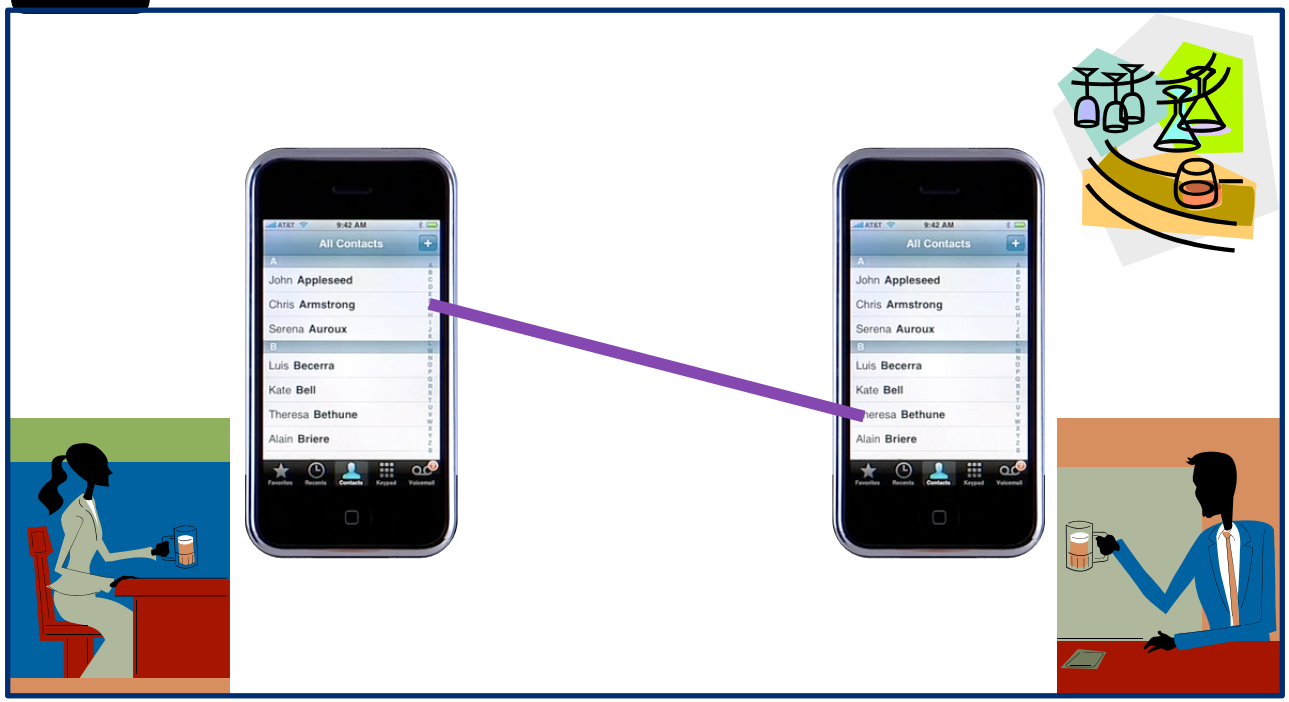
- Use a social network to limit Sybils
  - Addresses brute-force attack
- New technique: *layered identifiers*
  - Addresses clustering attacks

Two main phases

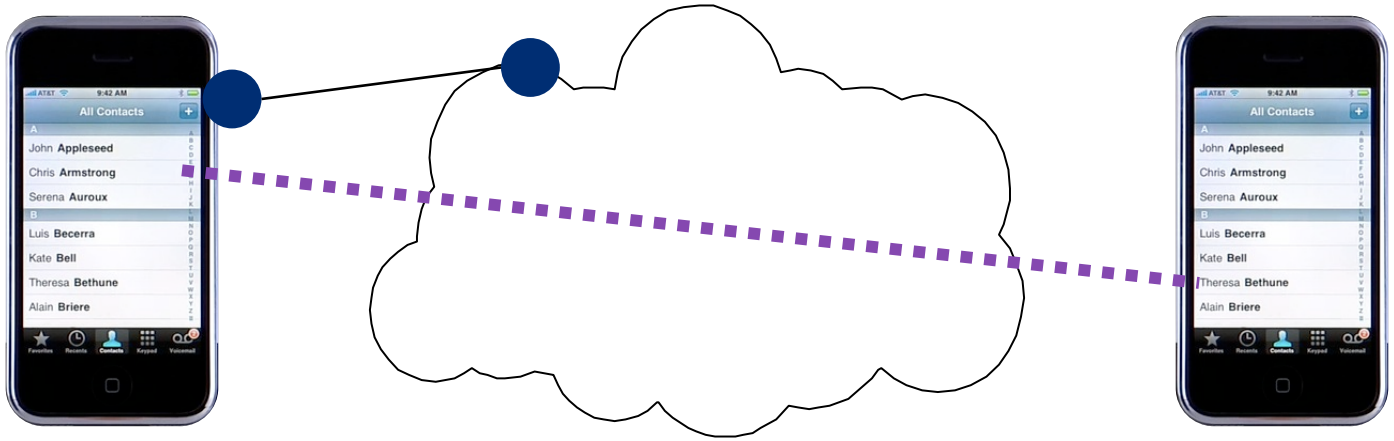
- SETUP: periodically build tables using social links
- LOOKUP: use tables to route efficiently



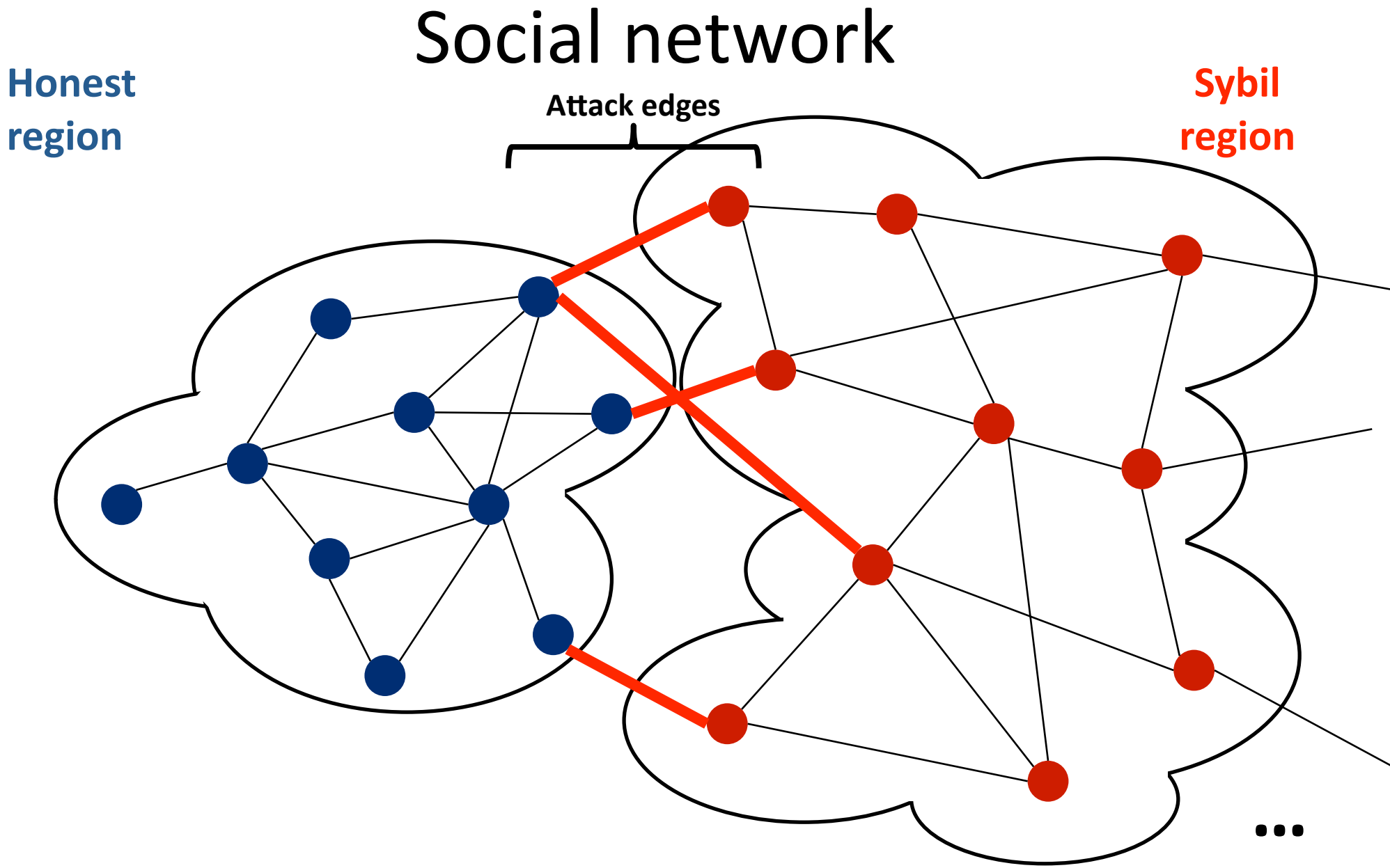
Social links created



Social links maintained over Internet

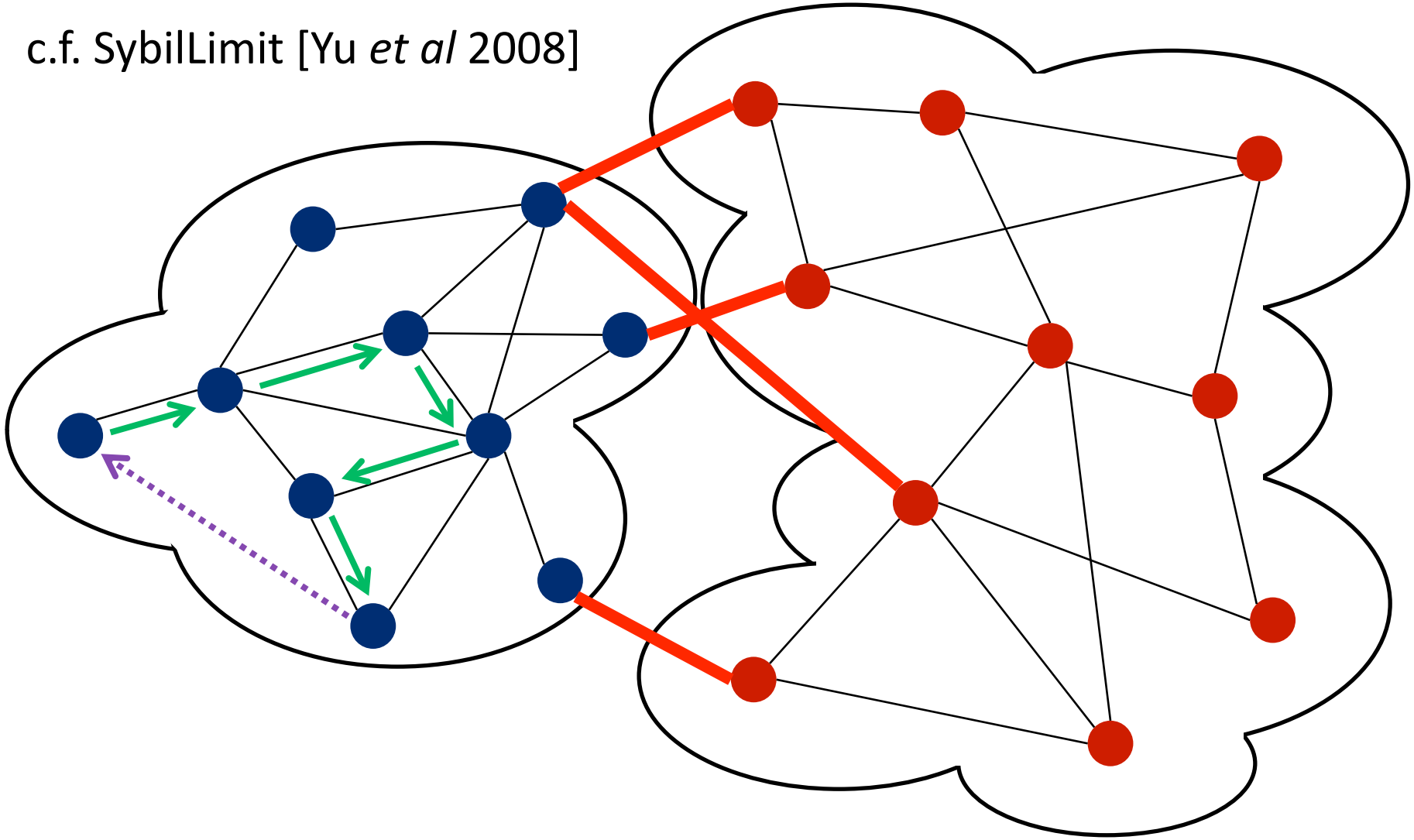


Social network



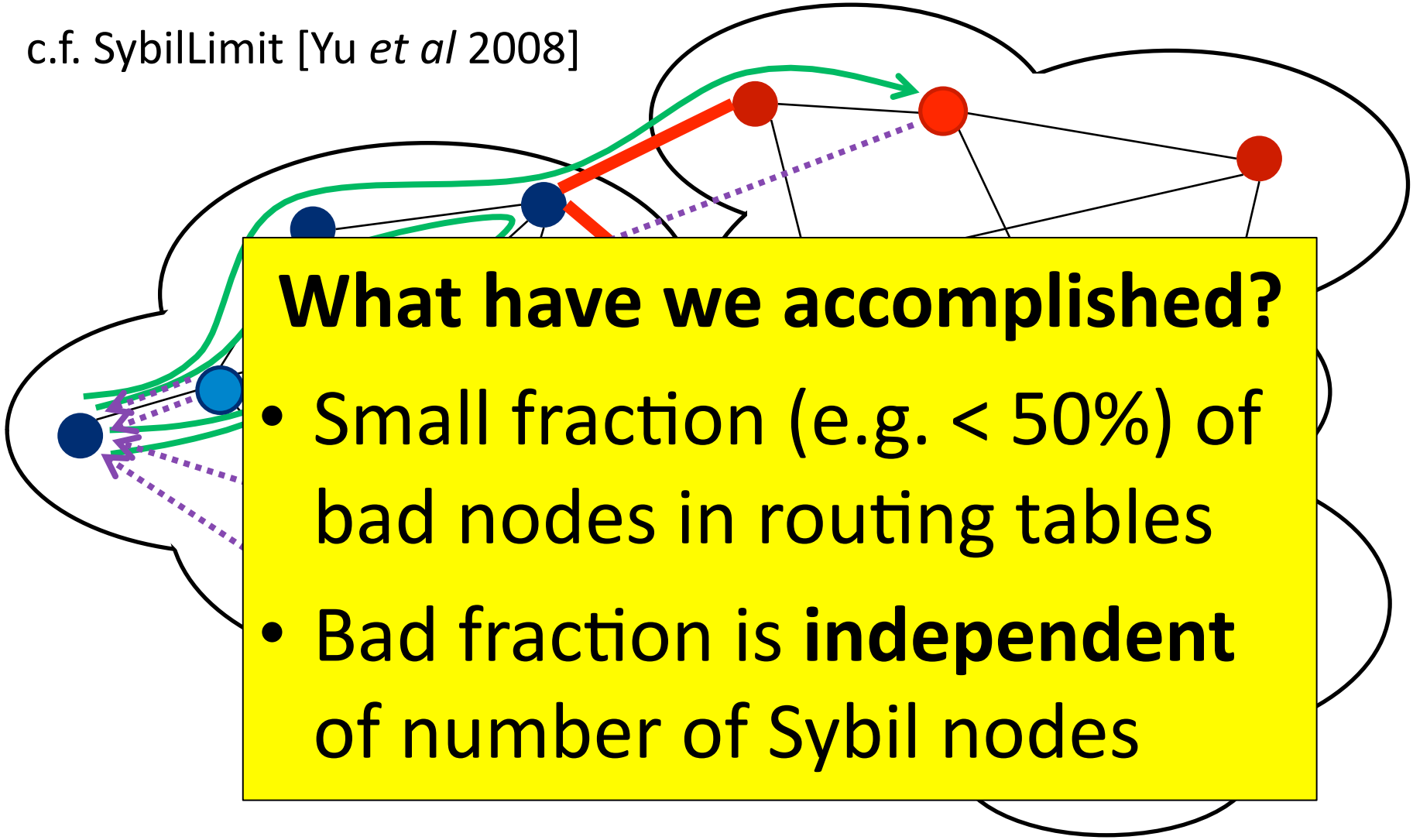
Random walks

c.f. SybilLimit [Yu *et al* 2008]



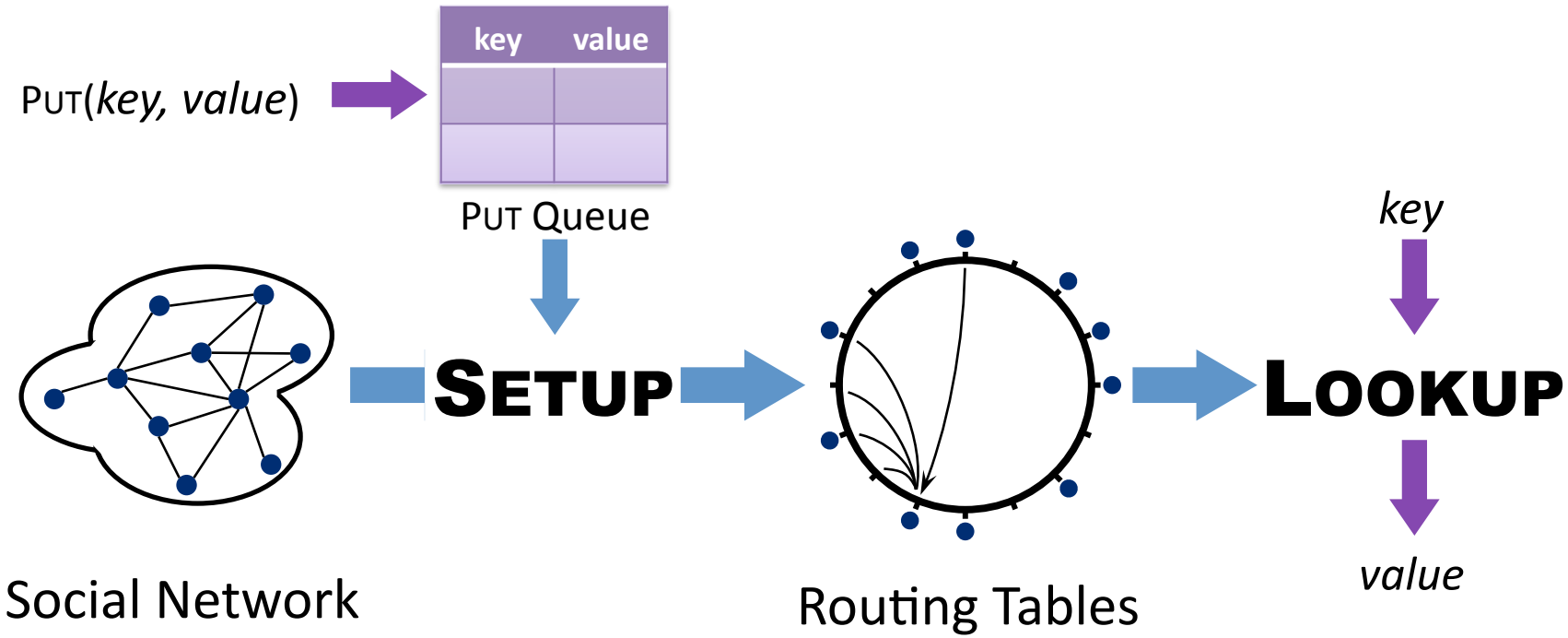
Building tables using random walks

c.f. SybilLimit [Yu *et al* 2008]



What have we accomplished?

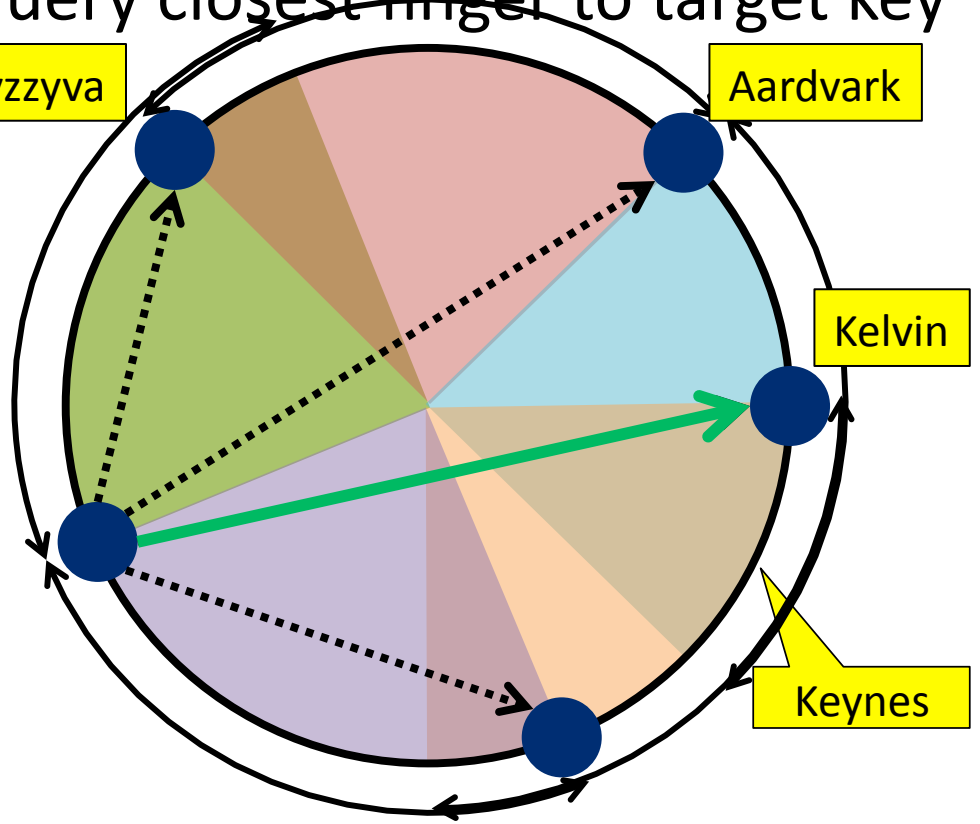
- Small fraction (e.g. < 50%) of bad nodes in routing tables
- Bad fraction is **independent** of number of Sybil nodes


PUT(*key, value*)
key
value
PUT Queue
SETUP
LOOKUP
key
value
Social Network
Routing Tables

Routing table structure

- O(√n) fingers and O(√n) keys stored per node
- Fingers have random IDs, cover all keys WHP
- Lookup: query closest finger to target key

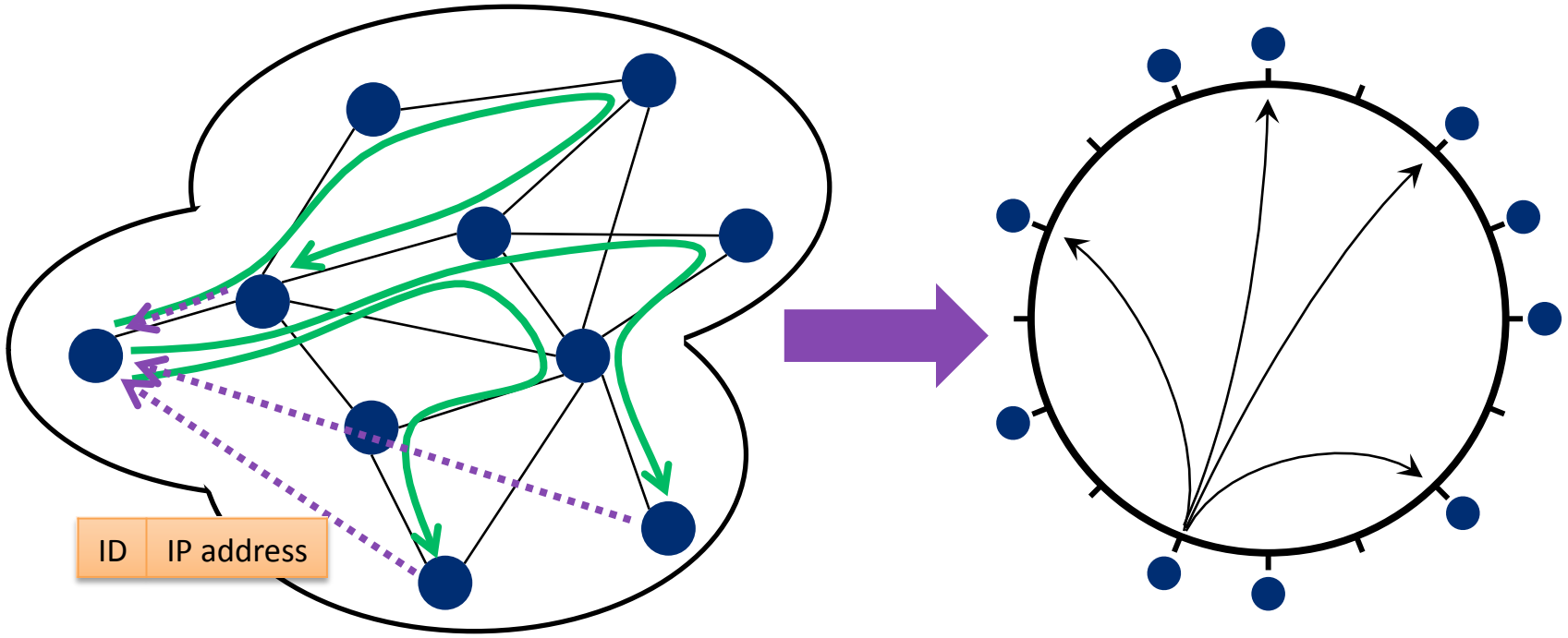
Finger tables:
(*ID*, *address*)



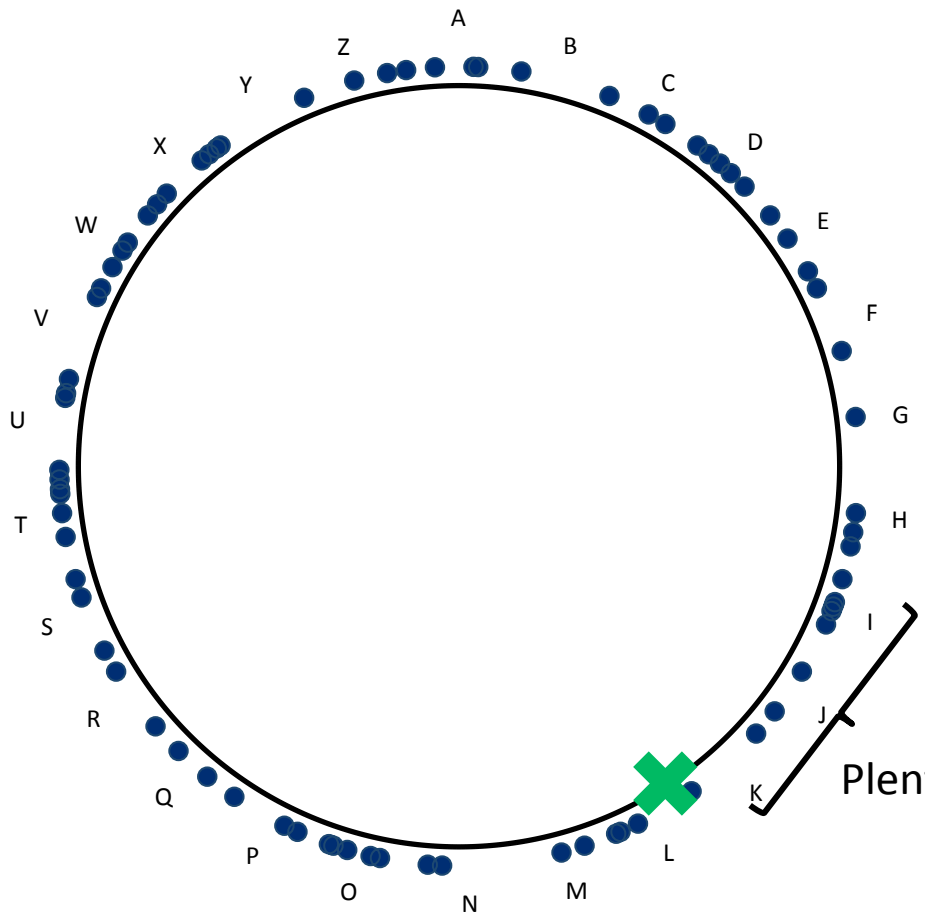
Key tables:
(*key,value*)

From social network to routing tables

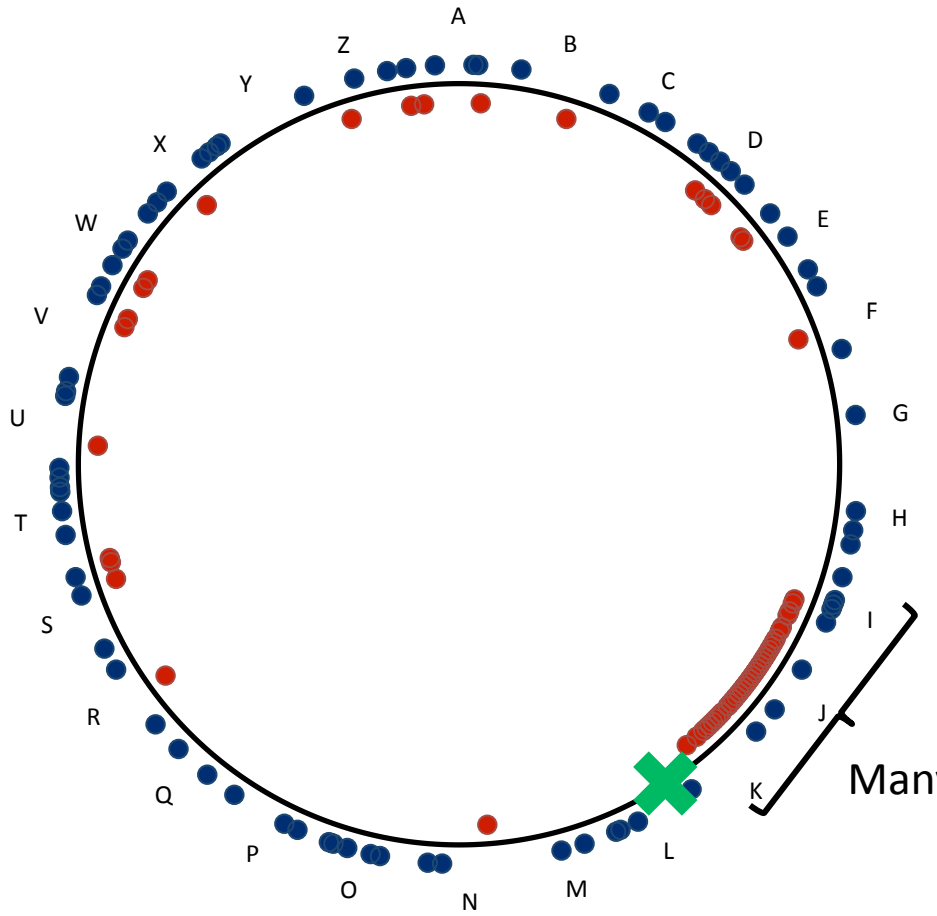
- Finger table: randomly sample $O(\sqrt{n})$ nodes
- Most samples are honest



Honest nodes pick IDs uniformly

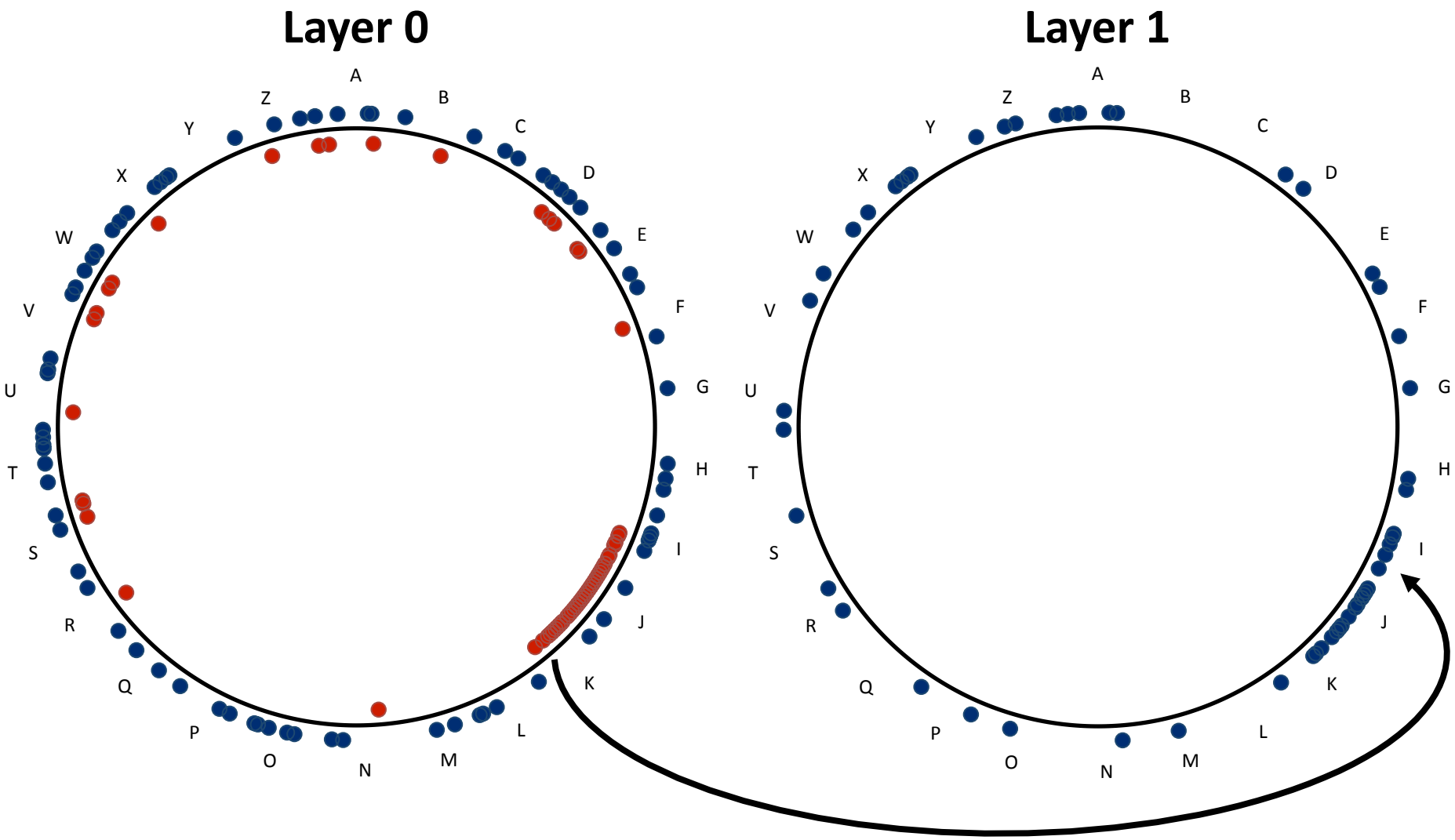


Sybil ID clustering attack

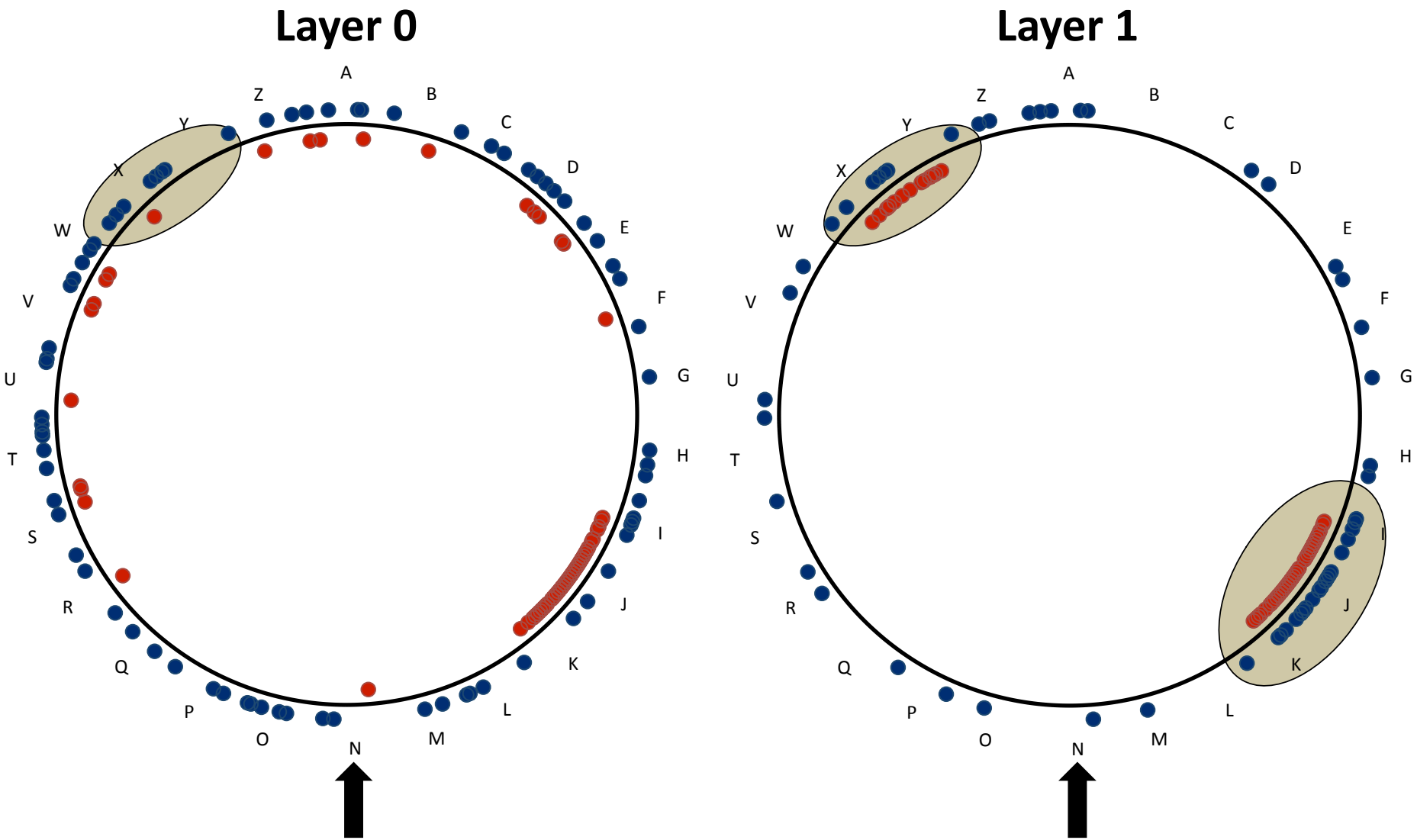


[Hypothetical scenario: 50% Sybil IDs, 50% honest IDs]

Honest layered IDs mimic Sybil IDs

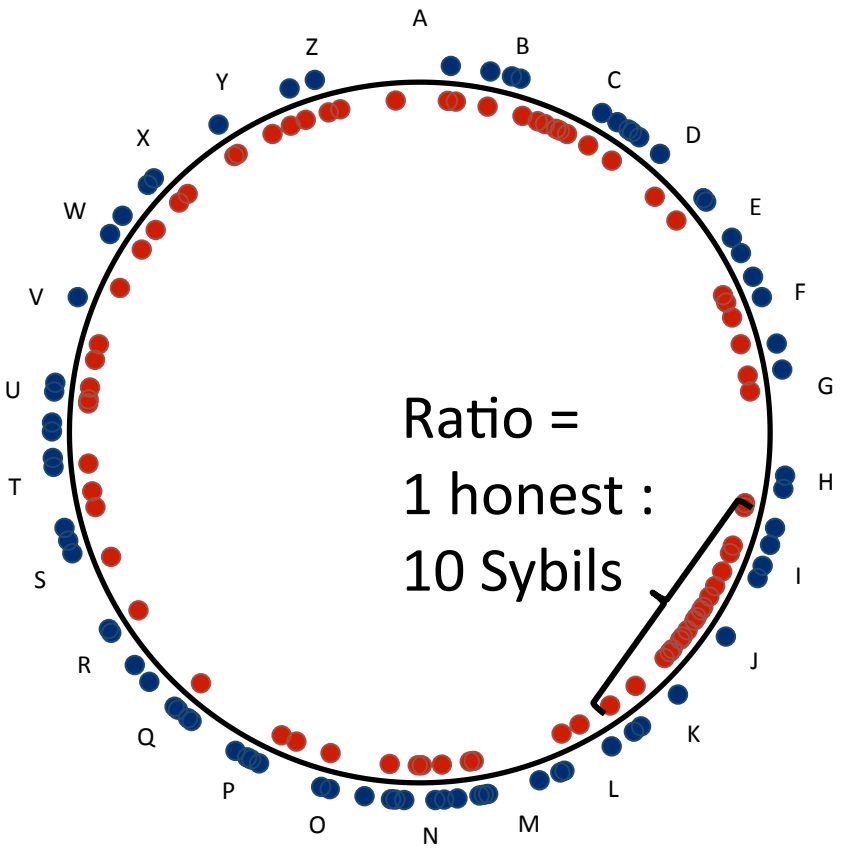


Every range is balanced in some layer

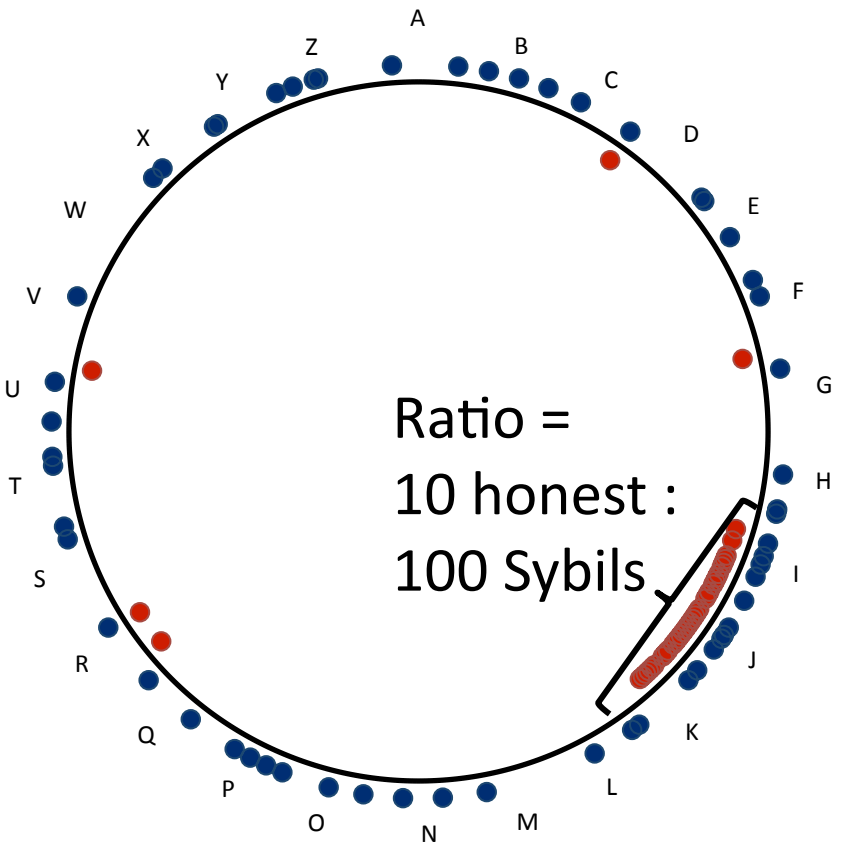


Two layers is not quite enough

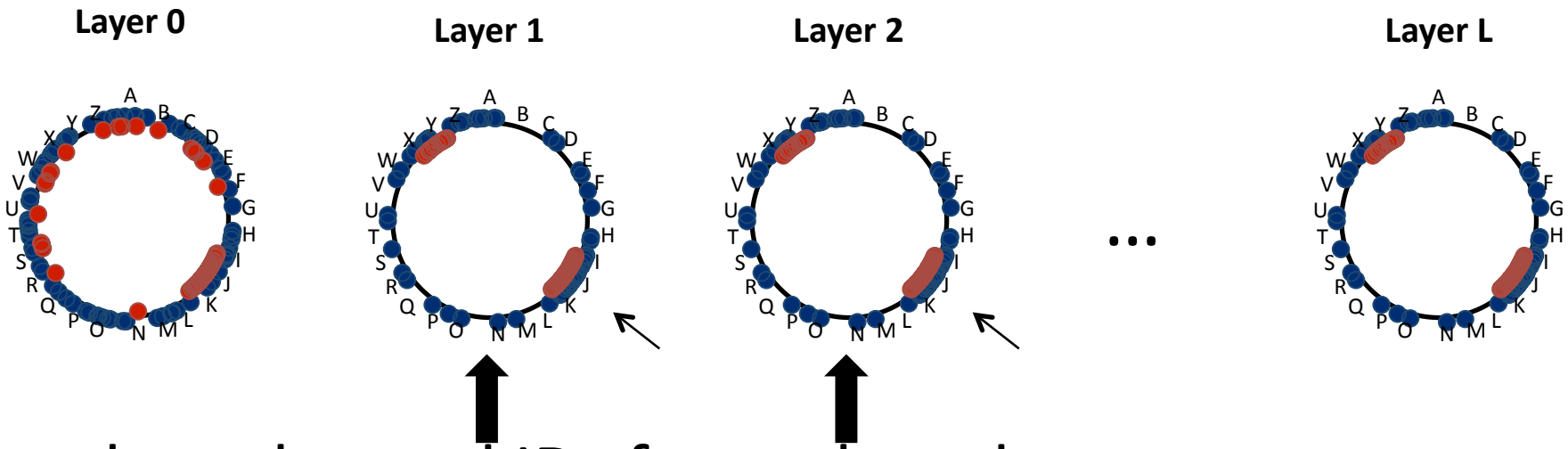
Layer 0



Layer 1



Log n parallel layers **is** enough



- log n layered IDs for each node
- Lookup steps:
  1. Pick a random layer
  2. Pick a finger to query
  3. GOTO 1 until success or timeout

Main theorem: secure DHT routing

If we run Whānau's SETUP using:

1. A social network with walk length = $O(\log n)$ and number of attack edges = $O(n/\log n)$
2. Routing tables of size $\Omega(\sqrt{N} \log N)$ per node

Then, for any input key and all but εn nodes:

- Each lookup attempt (i.e., coin flip) succeeds with probability $\Omega(1)$
- Thus GET(*key*) uses $O(1)$ messages (expected)

Evaluation: Hypotheses

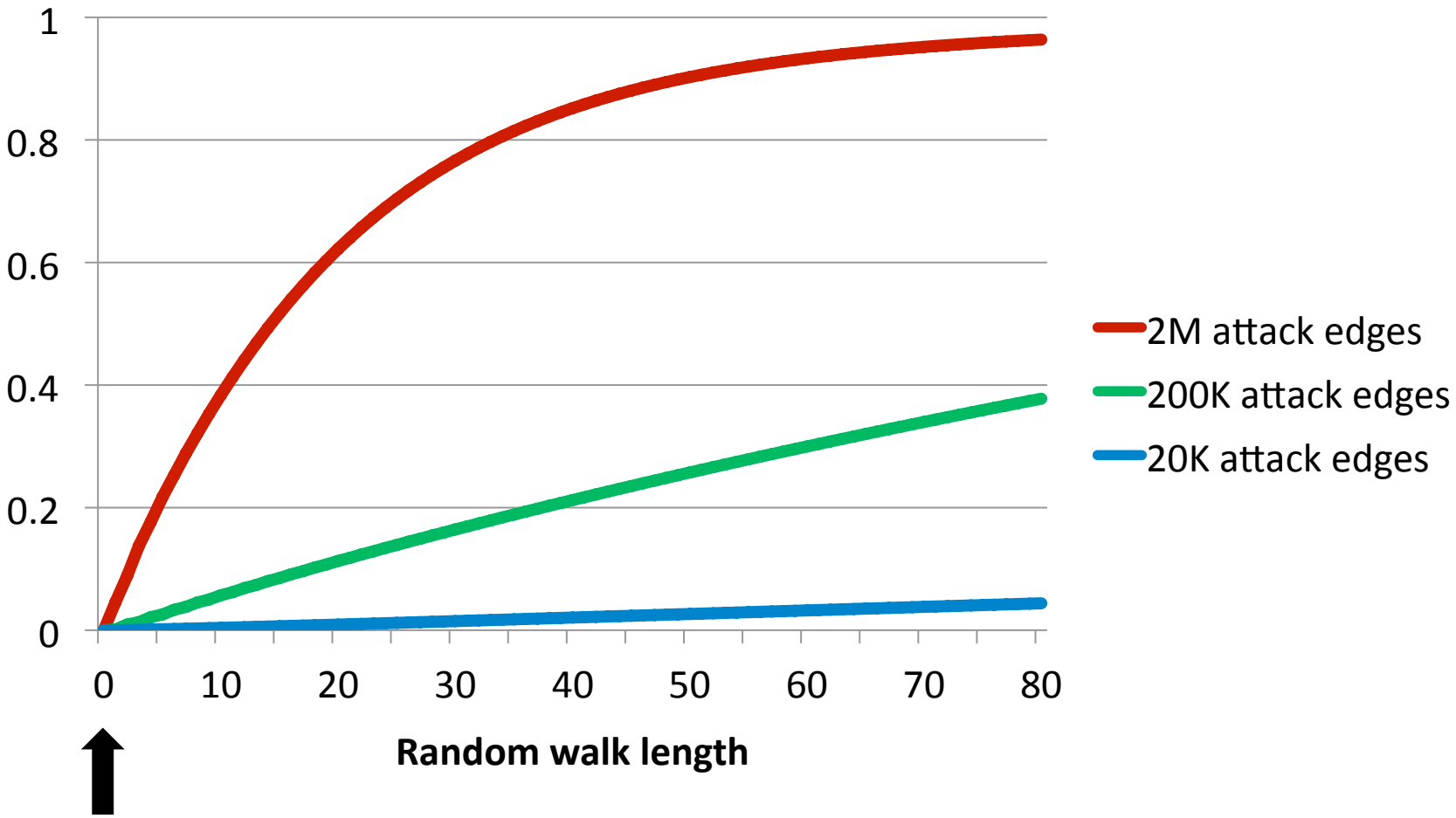
1. Random walk technique yields good samples
2. Lookups succeed under clustering attacks
3. Layered identifiers are necessary for security
4. Performance scales the same as a one-hop DHT
5. Whānau handles network failures and churn

Method

- Efficient message-based simulator
  - Social network data spidered from Flickr, Youtube, DBLP, and LiveJournal (n=5.2M)
  - Clustering attack, varying number of attack edges

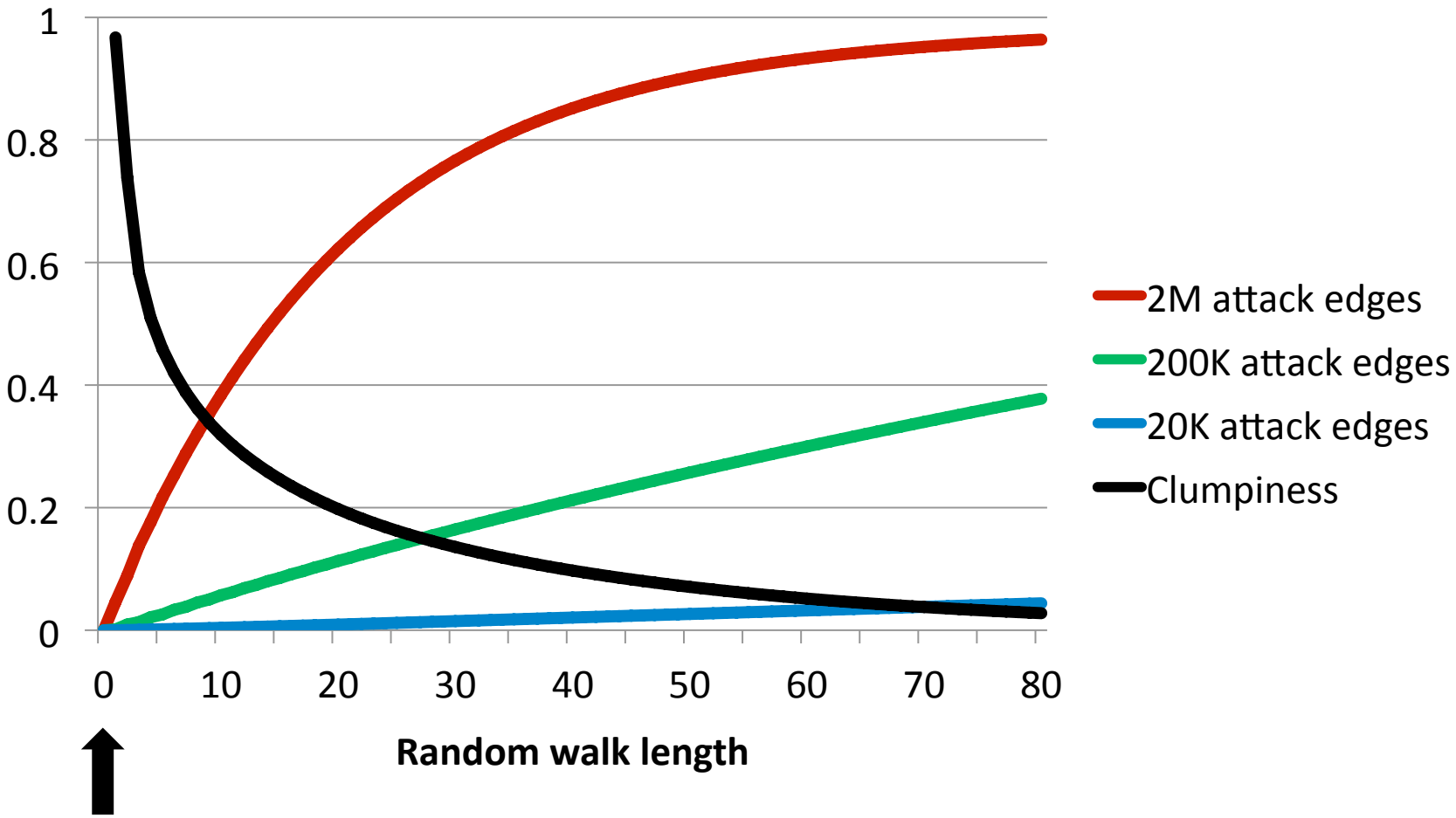
- PlanetLab implementation

Escape probability



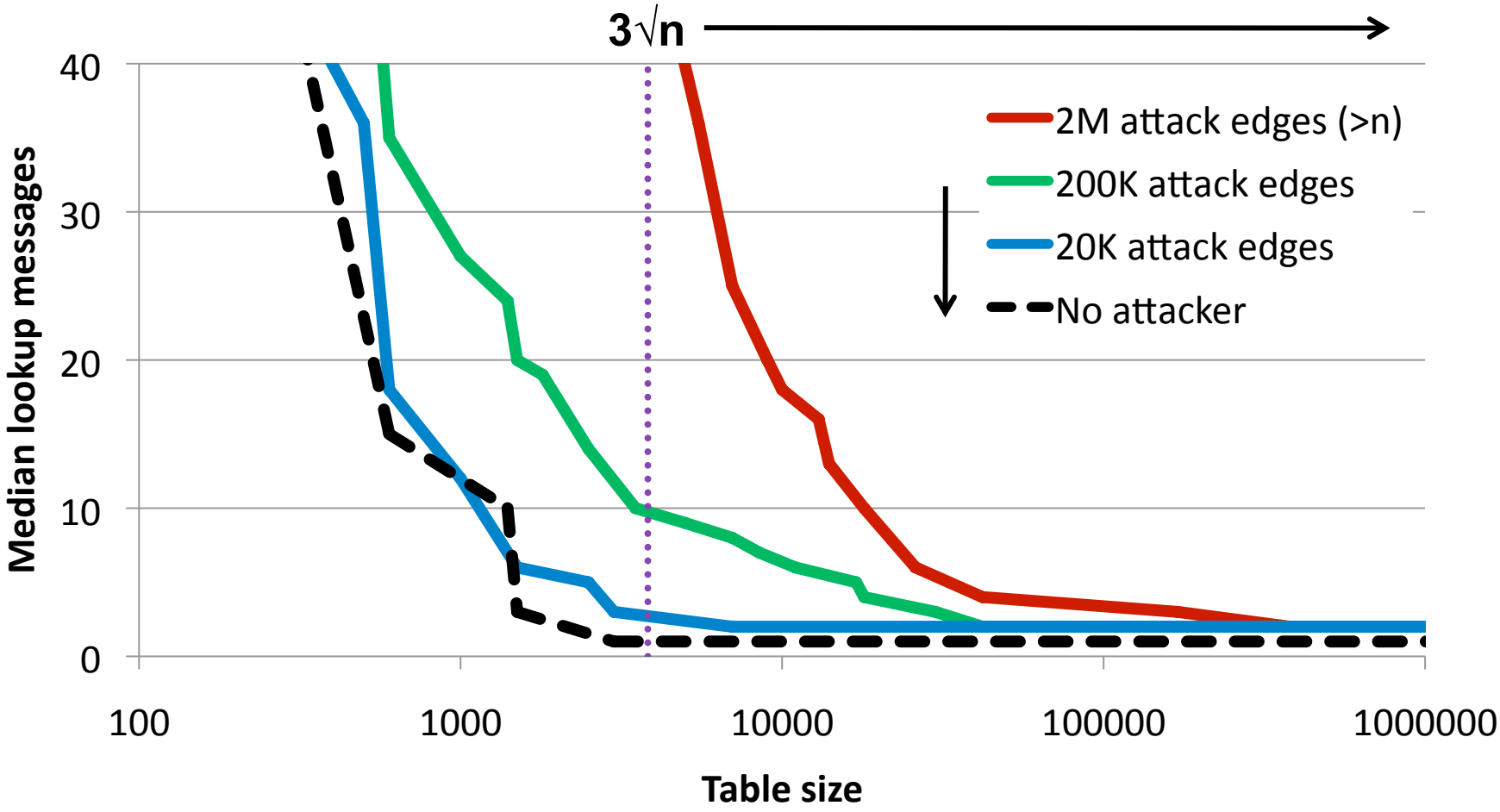
[Flickr social network: n ≈ 1.6M, average degree ≈ 9.5]

Walk length tradeoff



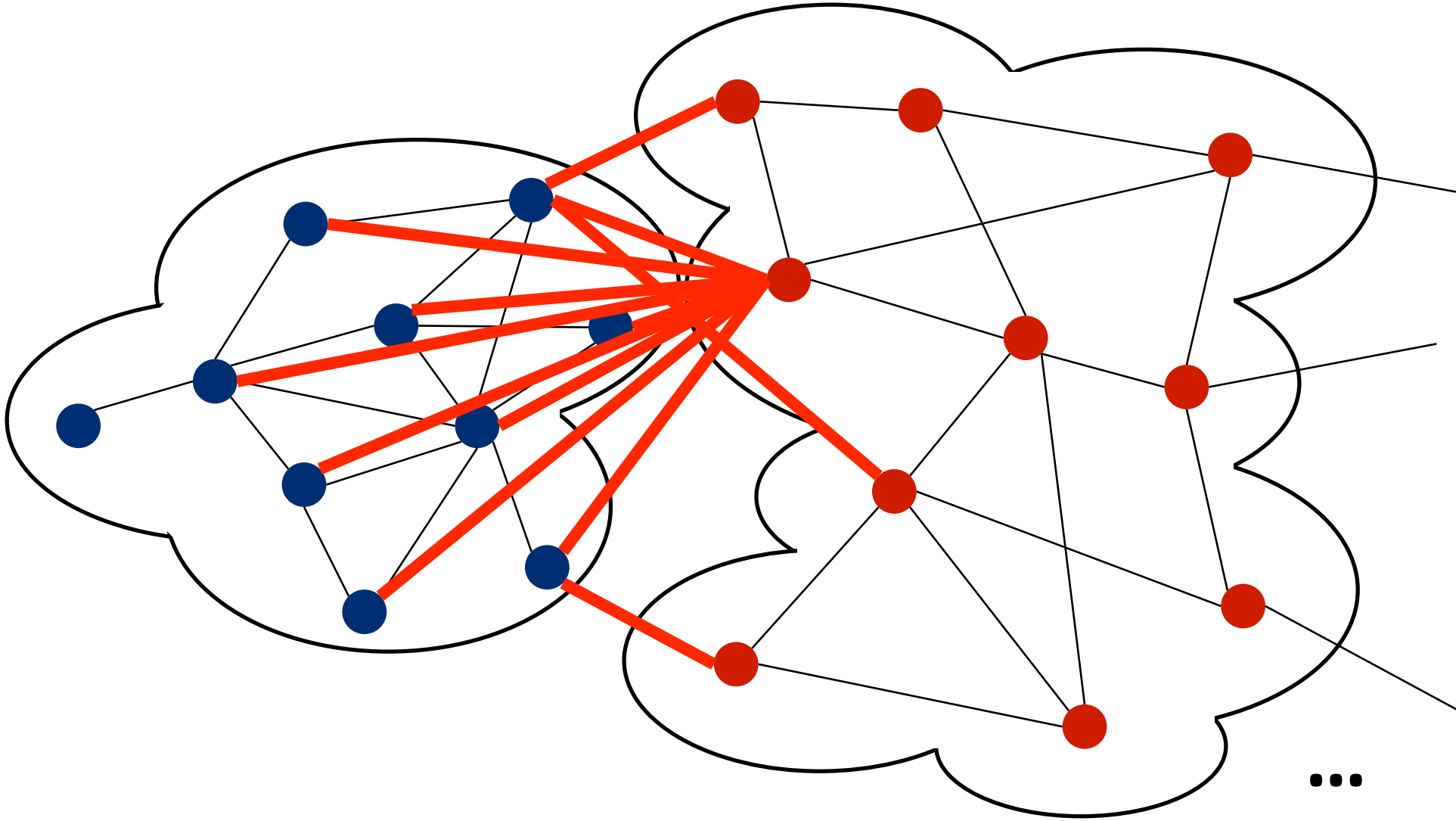
[Flickr social network: n ≈ 1.6M, average degree ≈ 9.5]

Whānau delivers high availability



[Flickr social network: n ≈ 1.6M, 3√n ≈ 4000]

Everything rests on the model…



Contributions

- Whānau: an efficient Sybil-proof DHT
  - Use a social network to filter good nodes
  - Resist up to $O(n/\log n)$ attack edges
  - Table size per node: $O(\sqrt{N} \log N)$
  - Messages to route: $O(1)$

- Introduced layers to combat clustering attacks