Lecture 2

Building secure systems on & for Social Networks

Or, Securing the ties that bind us

Nishanth Sastry

Objectives

In this hour, you will learn how to:

- Preserve your privacy in social network systems
- Prevent Sybil attacks in social networks
- Build a secure DHT overlay on the social graph
- Engineer security using offline methods

You should be able to describe/define:

- Privacy breach & de-anonymization in social nets
- Attribute-based encryption
- Fast mixing in social networks, and its uses

Part 1: Keeping you anonymous from Facebook, its advertisers, and Big Brother

PRESERVING YOUR PRIVACY

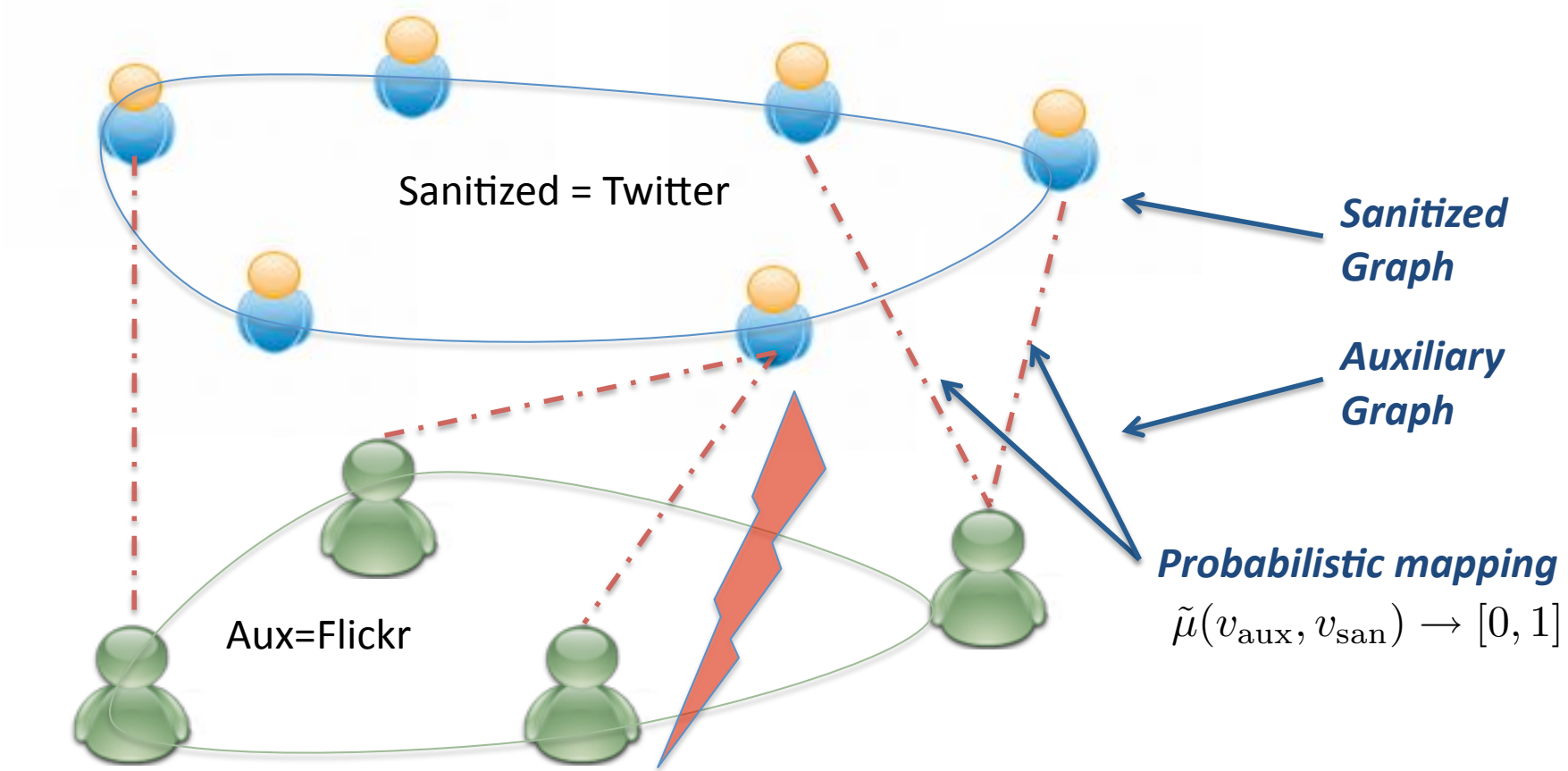
In the good old days, you could be anonymous...

"On the Internet, nobody knows you're a dog."

Now: Attribute data leaks identity!

- AOL releases "anonymized" user search histories
  - ID 4417749 traced to a 62 year old woman in Georgia
- 87% of Americans uniquely identified using just 3 attributes: zip, sex, date of birth (1990 census)
- "8 friends are enough" [BASA09]: searchable public listings of facebook users can be attacked to obtain aggregate statistics about graph

What is a privacy breach? [NS09]



$$P_{\text{posterior}}[X(v_{\text{aux}}) = x] - P_{\text{prior}}[X(v_{\text{aux}}) = x] > \delta$$

Privacy breach: Prior belief in value of some (node/edge) attribute X in Auxiliary Graph is improved using topological information from the Sanitized Graph

Social network de-anonymization

- Identify seed nodes
  - k-cliques whose degree distributions correspond in the two graphs
- Propagation (Add heuristics to taste):
  - If threshold no. of neighbours of unmapped node *u* are mapped to neighbours of node *v*, then map *u* to *v*
  - *Reverse AUX←→SANITIZED. Is mapping is stable?*
  - Lather, rinse, repeat…
- Truth testing: common strings map each other?
  - Mapping Tamedfalcon213 in two networks is valuable, but joe19, not so much…

Securing online social networks (OSN)

Problem: Users trust OSN providers, developers of 3rd party apps, advertisers etc. to not misuse their attribute data

Solution: Hide 1. topology or 2. attributes:

1. Dispense with the OSN. Keep the social apps. Use crypto for distributed access control.

   Persona [BBS+09] (**our focus**), Lockr [TSGW09]

2. Hide attributes in plain sight. NOYB [GTF08]

   Use a cipher to pick another user's attribute, preserving the overall distribution of attribute values.

   - Alice, F, 25 takes her age from Bob, M, 28.
   - OSN providers and advertisers see Alice, F, 28
   - Only auth users can do the reverse mapping to Alice, F, 25.

OSN Access Control Challenges

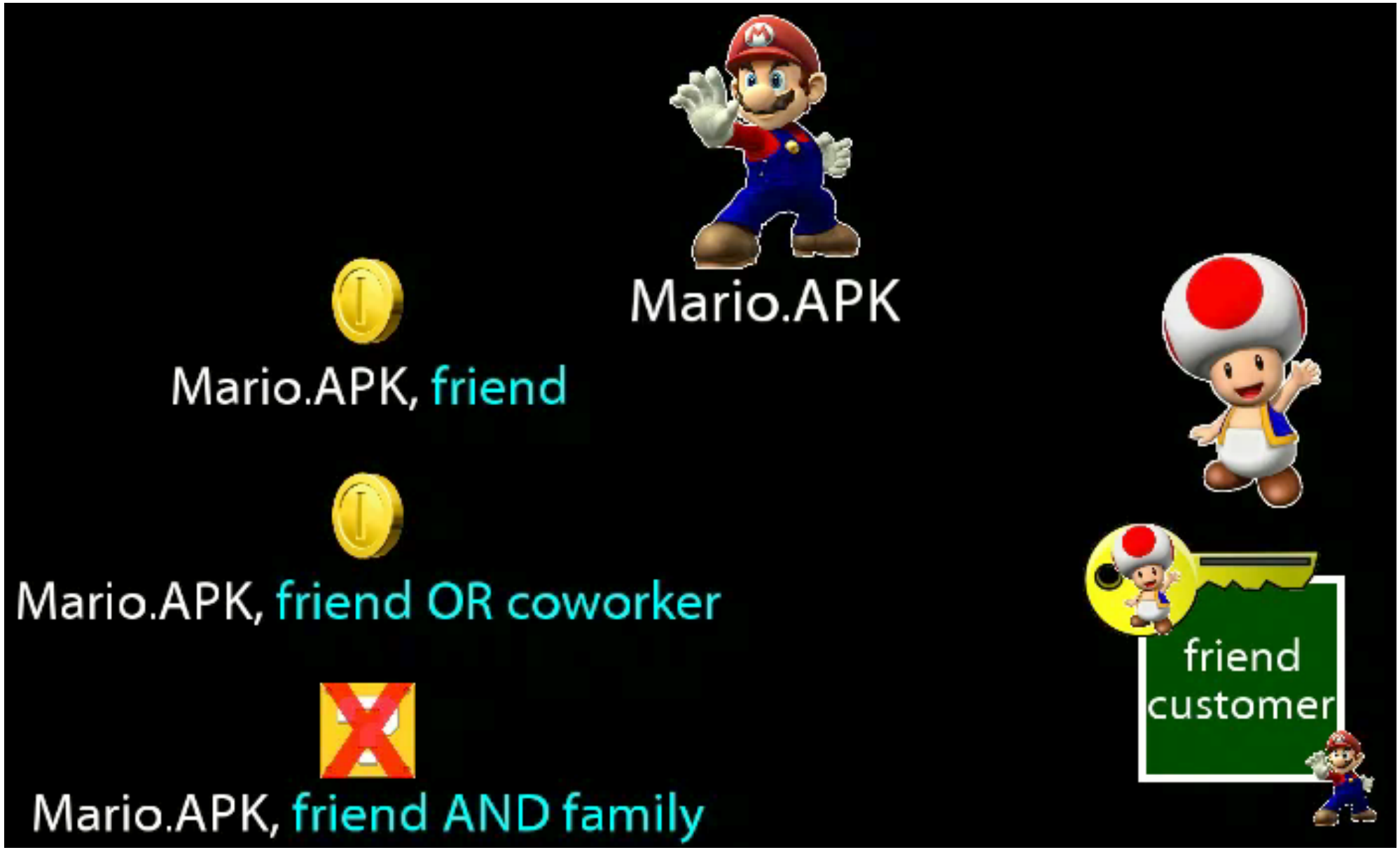
- "Called in sick to work. Let's go skiing"
  - Status intended for friends only, not co-workers
  - Need fine-grained access control
- Alice wants to send "Surprise party for Bob" to all friends of Bob, who are also local to Cam
  - Bob has pre-defined groups "Friends" & "in-Cam"
  - Need friend-of-friend interactions & flexible groups

- Social Apps involve multi-party read-write with writers not always knowing reader set
- Many groups can be defined:
  - friend, foe, friend AND in-cam, foe OR customer

Simple solutions have drawbacks

- Individual keying for intended readership
  - Won't work without writers knowing audience
- Traditional group keying (symmetric keys) requires writer to be member of group
- Too many groups (friend, foe, friend AND in-cam, foe OR customer) → Group keying is not scalable
- Possible solution: Composable groups
  - Alice needs to send to friend AND in-cam: Double encrypt, first for friend and then for in-cam
  - Susceptible to collusion! Foe in-cam AND friend in London can collude to decrypt.

Cryptonite: Attribute-based Encryption



OSN in Persona Architecture

- Persona separates storage from social apps
- Storage has encrypted data. Apps have refs to data
- Attribute-based secret keys distributed out-of-band
  - User has a master key to generate new keys for friends
- Writer
  - stores encrypted data on their storage server
  - Posts reference to data on social app
- Reader get reference to data from social app
- Anyone can fetch encrypted data
- … But can decrypt only if access structure allows

Part 2: Making users on social network accountable

PREVENTING SYBIL ATTACKS

Sybils: "schizophrenic" identities

If # of sybil nodes accepted	Then applications can do
$< n$	majority voting
$< n/2$	byzantine consensus
$< n/c$ for some constant c	secure DHT [Awerbuch'06, Castro'02, Fiat'05]

Assuming social graph with n nodes and m edges

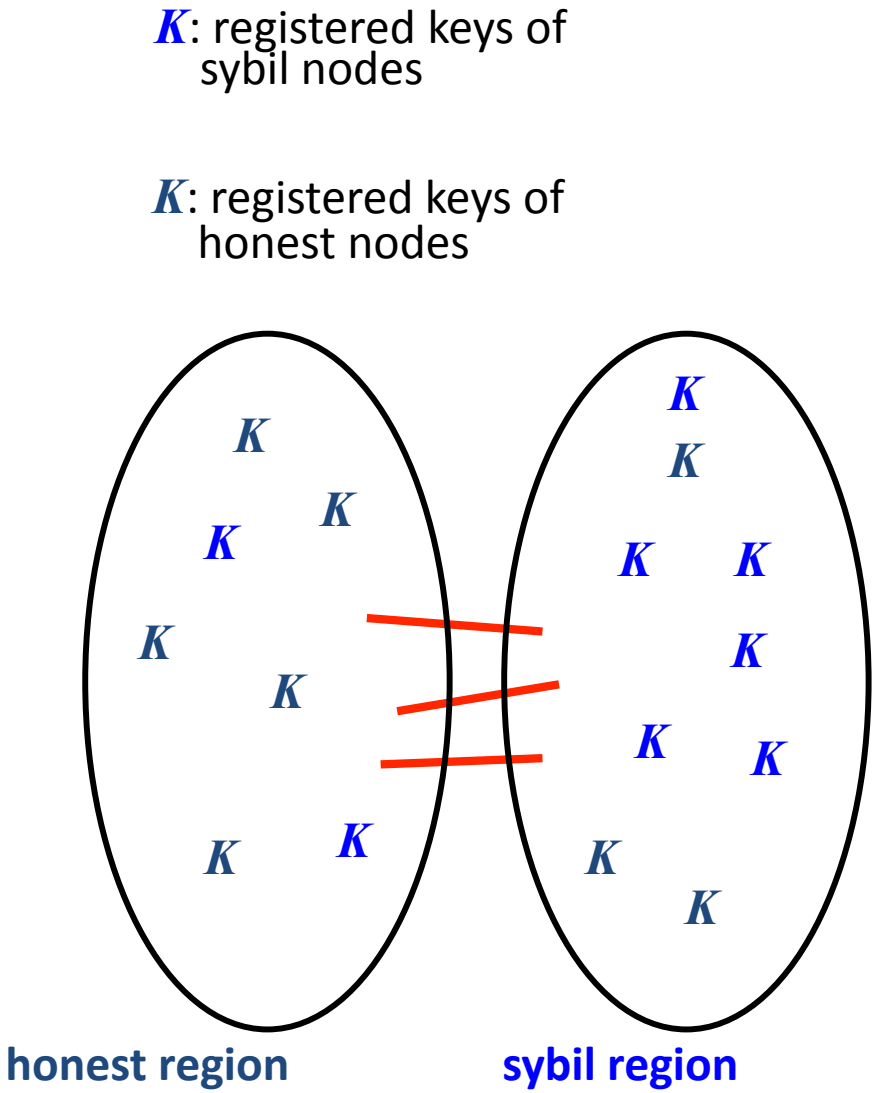
Sybil threat and solution

- Sybils can create any number of false nodes
- Any number of false relations between nodes
  - Sybil net can look identical to real social sub-graph

- Solution: It is more difficult to create link between an honest node and a sybil node
- Exploit scarcity of such attack edges

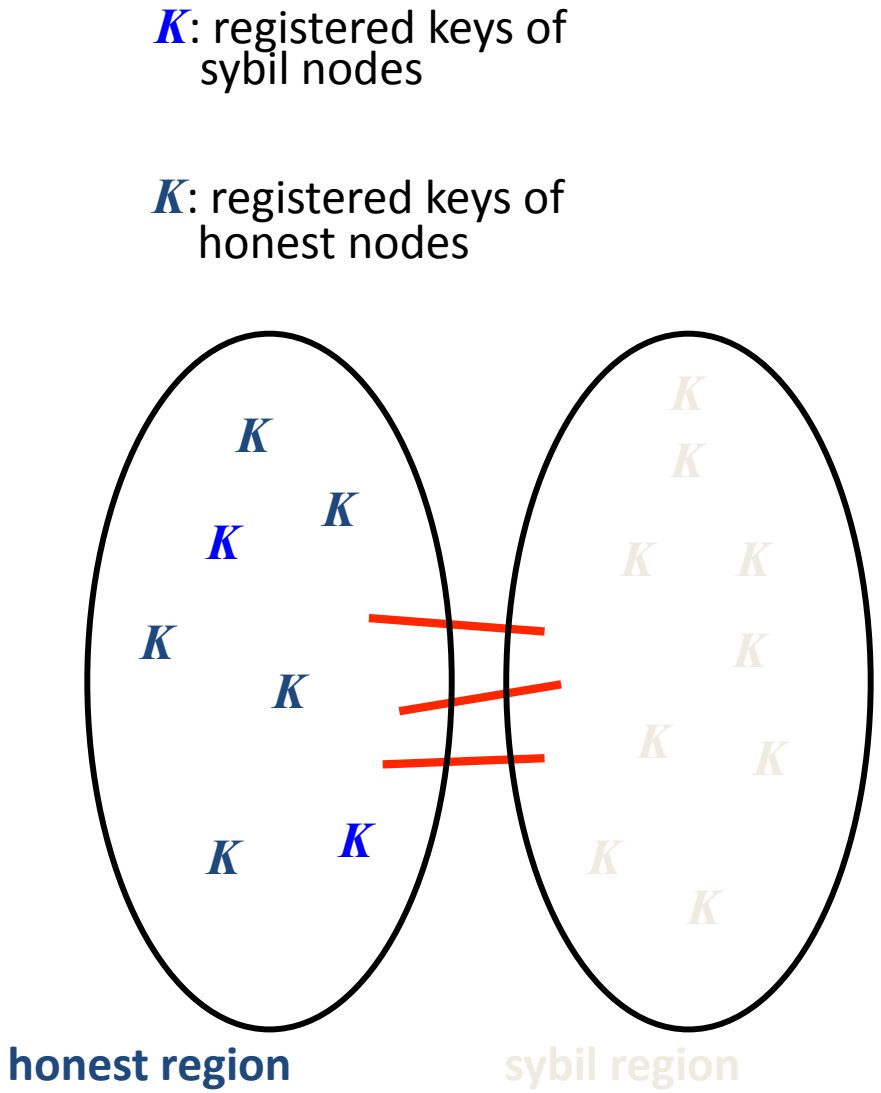
SybilLimit: Strawman Design – Goal 1

- Ensure that sybil nodes (collectively) register only on limited number of honest nodes
  - Still provide enough "registration opportunities" for honest nodes

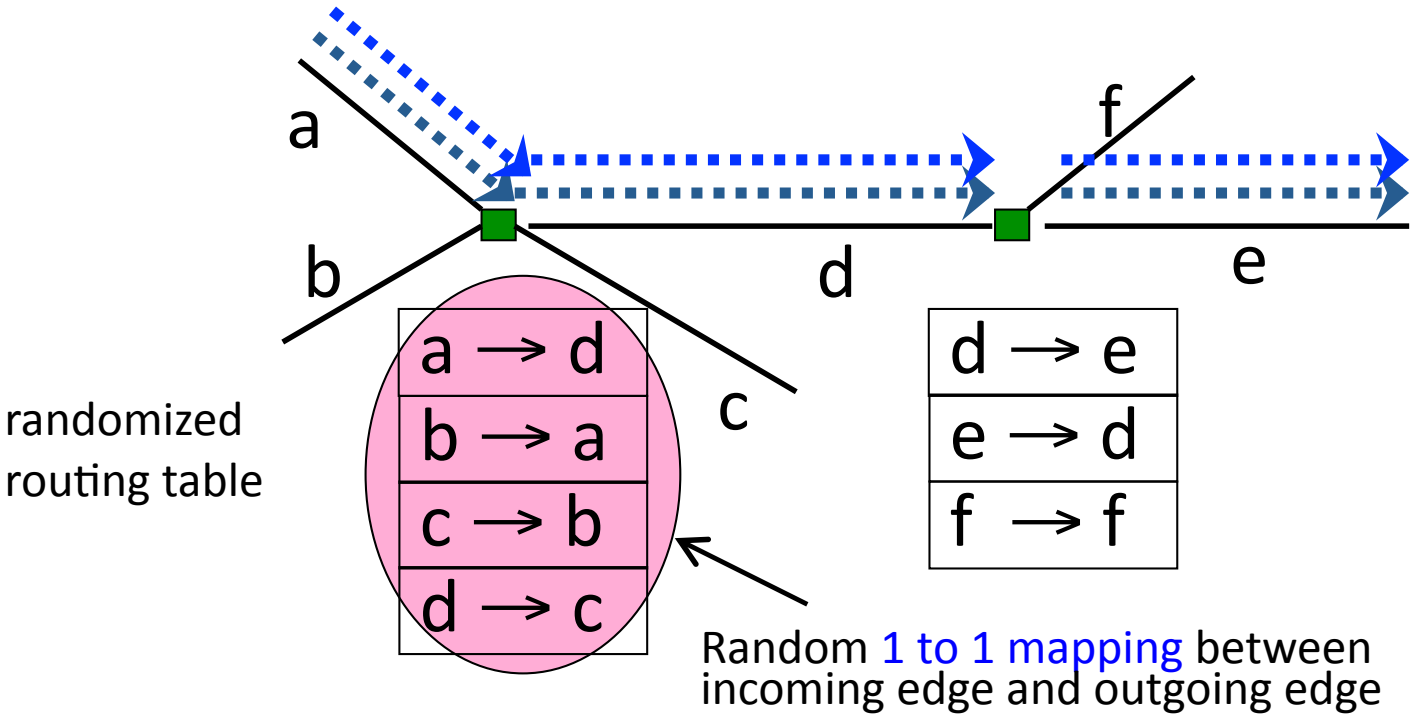


SybilLimit: Strawman Design – Goal 2

- Accept S iff K_S is registered on sufficiently many honest nodes
  - Without knowing where the honest region is !
  - Circular design? We can break this circle…



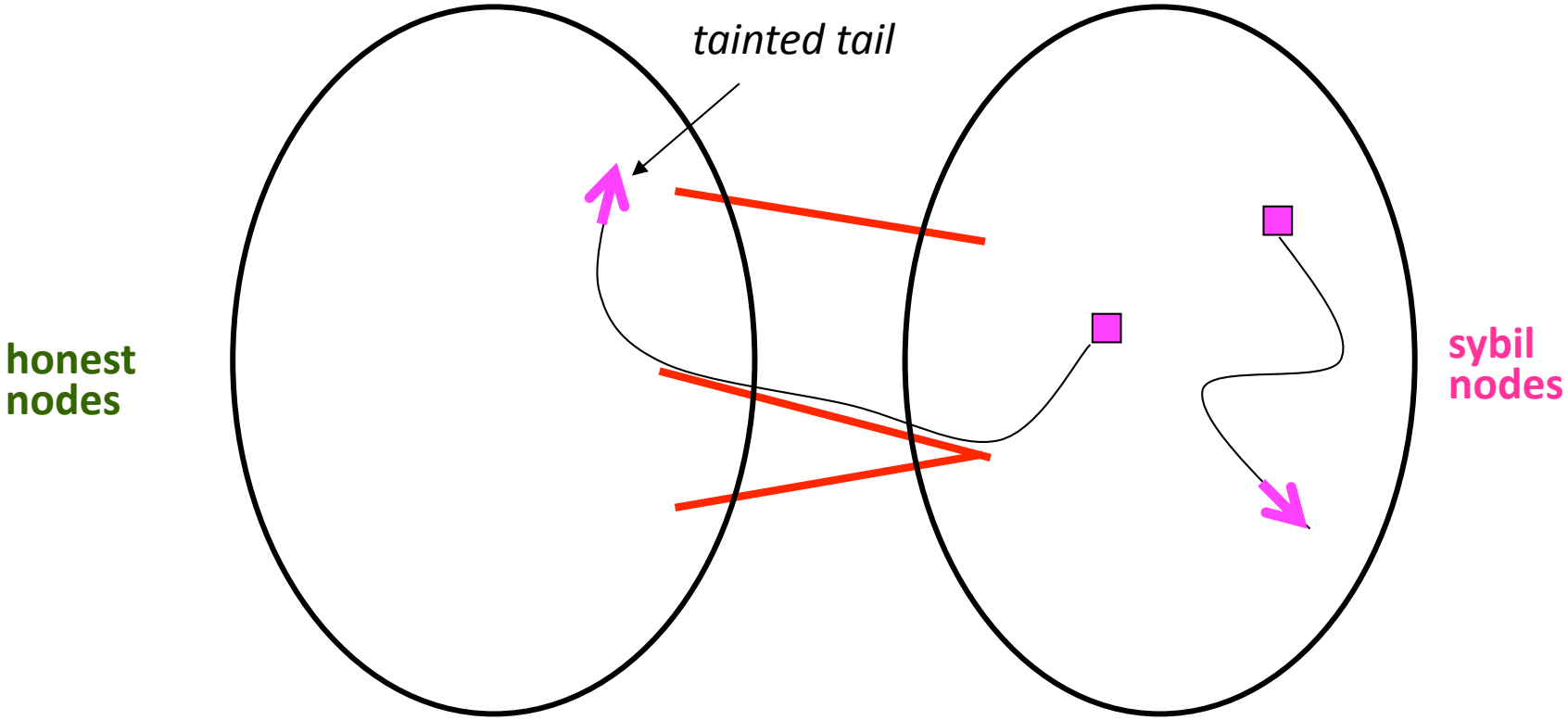
Random Route: Convergence



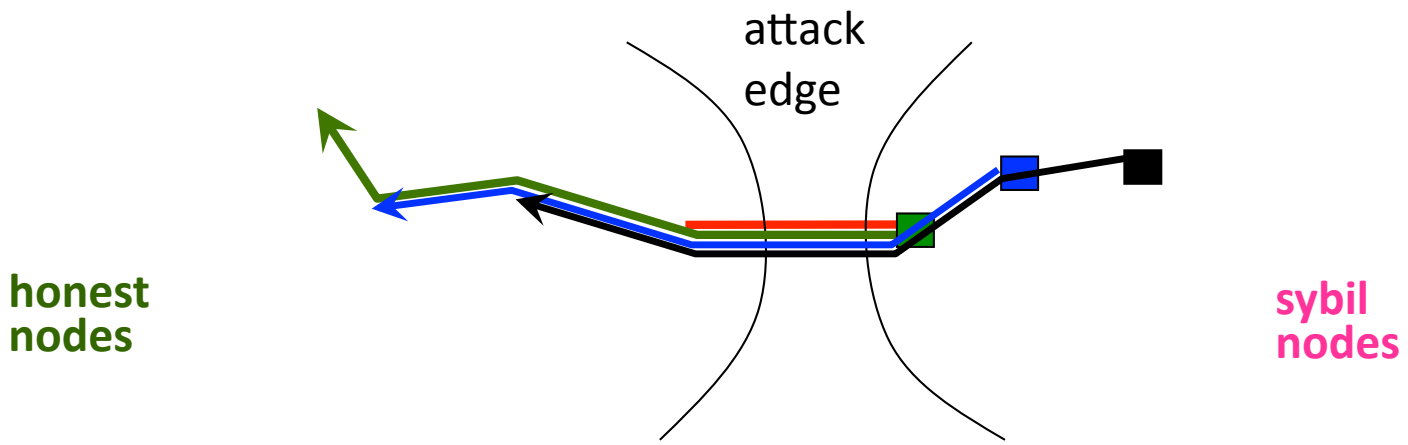
Using routing table gives Convergence Property:
Routes merge if crossing the same edge

Tails of Sybil Suspects

- Every sybil suspect initiates a random route from itself and registers its key at the tail

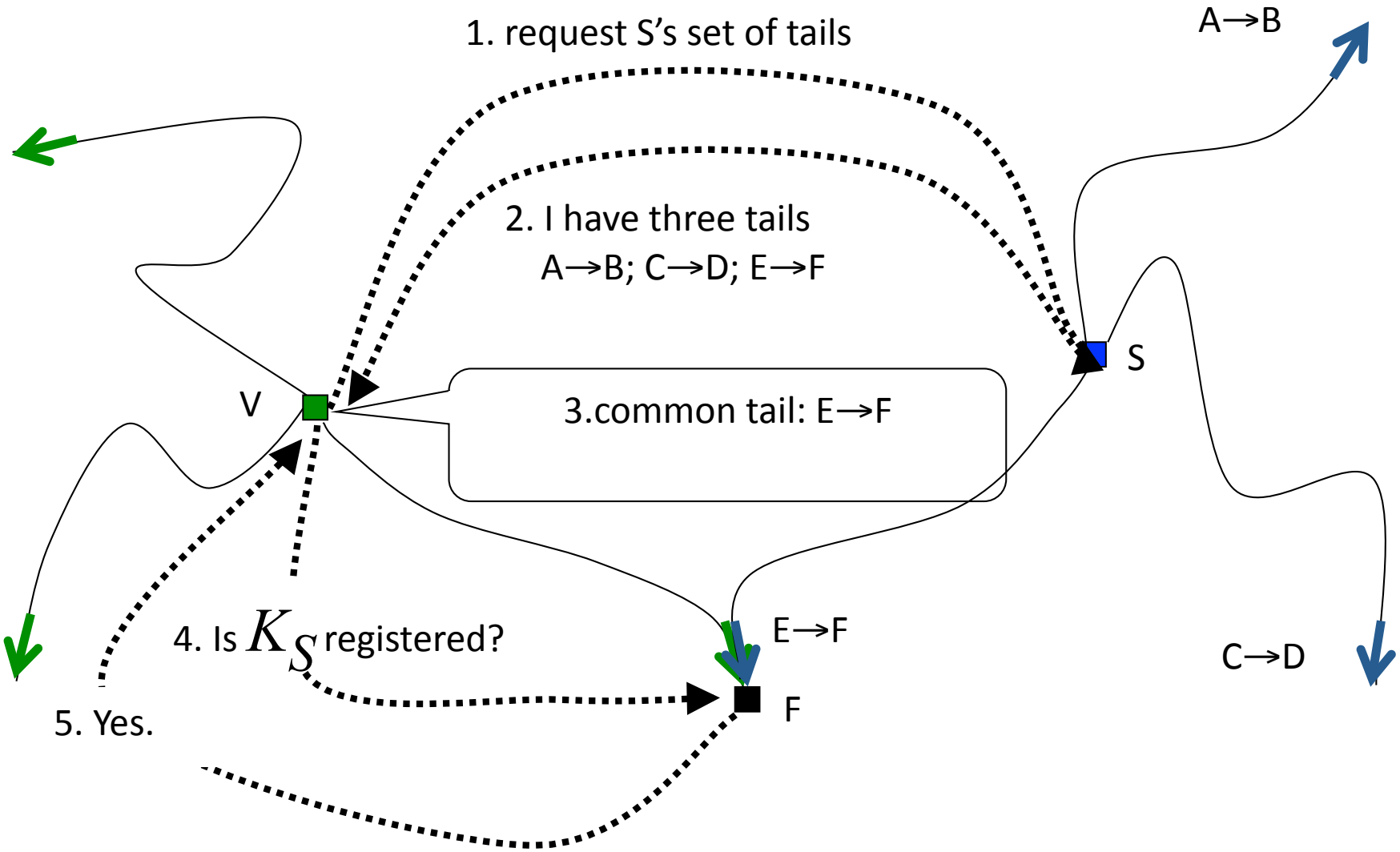


The number of tainted tails is small



- Suppose we do w step random routes from each node
- Claim: There are at most w tainted tails per attack edge
  - Proof: By the Convergence property
  - Regardless of whether sybil nodes follow the protocol
  - Will show w can be chosen to be $O(\log(n))$ in social networks
- Attack edges are few → #tainted edges ($=gw$) is small!
- Probability of honest tails "escaping" to sybil region=gw/n
  - If $g = \#attack\ edges$ is < $o(\log(n)/n)$, escape probability = $o(1)$

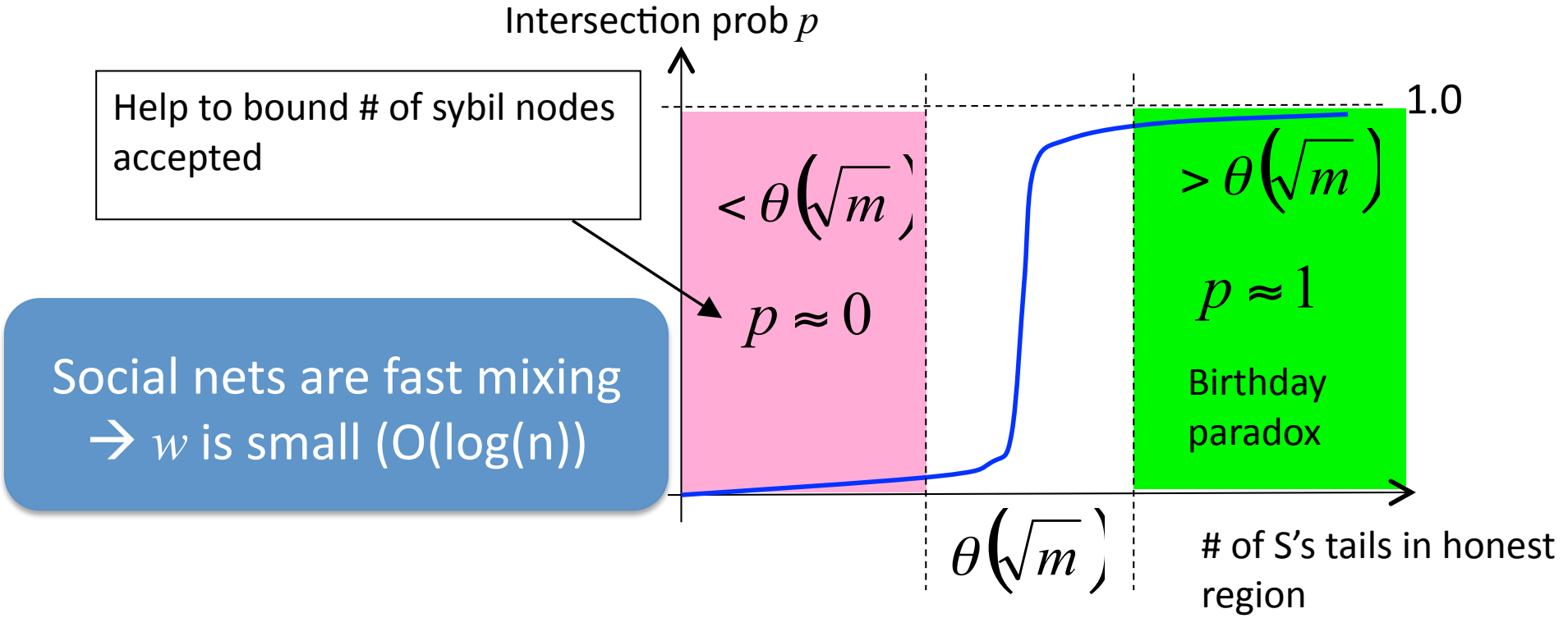
Verification Procedure



Leveraging fast mixing in social nets

If w = *mixing time*, tails (of S & V) are uniformly random

Assuming V has $\theta(\sqrt{m})$ tails in the honest region



SybilLimit Summary

- Goal 1: Ensure that sybils (collectively) register only on limited number of honest nodes
- Tainted tails < gw
- w small due to fast mixing of social nets

- Goal 2: Accept S iff K_S is registered on sufficiently many honest nodes
- "Sufficiently many" = $\theta\left(\sqrt{m}\right)$
- Intersection occurs *iff* S has $\theta\left(\sqrt{m}\right)$ tails in the honest region

Part 3. Application of fast-mixing in social networks

A SYBIL-PROOF DHT

Sybil attack in the DHT context

- DHT primer: nodes arranged in a virtual ring. "Finger tables" route look ups close to target id.
  - e.g. Sergei looks up Tanya's IP from finger Umberto
  - Goal: sub-linear (typ. logarithmic or faster) look ups.

- If Umberto is evil, he could tell Sergei to ask his (sybil) finger Uma, who sends Sergei to Upton, who sends Sergei to Ursula…

A Sub-linear Sybil-proof DHT [L08]

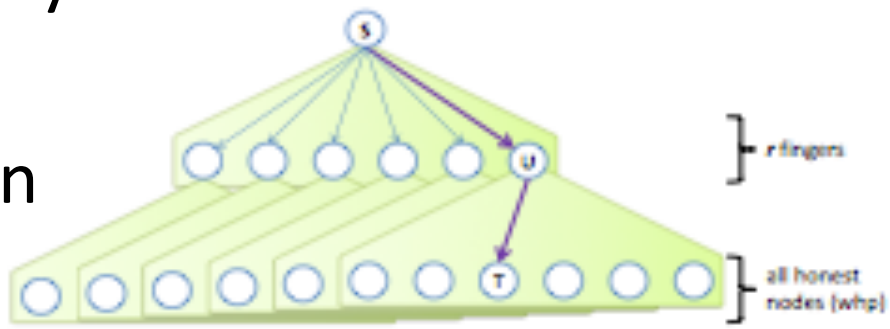
(Precursor to Whanau [LK10])

- Assumes fewer than $g=o(\log(n)/n)$ attack edges

- Routing table setup: Do r independent w-step random walks and record final node as finger
  - $o(1)$ escape probability → $\Omega(r)$ fingers are honest
  - mixing time = w → fingers are ends of random edges
    - target node is a finger with at least $\Omega(r)/m$ probability

Key lookups succeed w.h.p.

- Ask each finger whether they know target
- If honest finger has target in its table, the IP is returned
- Choose $r = \Theta(\sqrt{m \log m})$



$$p_{fail} = \left(1 - \frac{\Omega(r)}{m}\right)^{\Omega(r)} = O\left(\left(1 - \frac{\log m}{r}\right)^{r}\right)$$

$$= O(e^{-\log m}) = O\left(\frac{1}{m}\right)$$

Is social engineering really hard?

- Security relies on hardness of social engineering: Both SybilLimit and Whanau assume inducing edge from honest node to Sybil node is hard
  - At least, much harder than creating new sybils

BUT…

- Many well-networked users have guessable passwords
- People “friend” out of politeness: Brazilians on Orkut
- A Facebook friend is worth about 37 cents: Burger King offered free burger if you removed 10 friends
  - Nearly 200,000 friend links were removed

Part 4. Offline foundations for online security

SOCIAL ENGINEERING FOR ANONYMITY AND ACCOUNTABILITY

Confounding Charlie

- Boston metro introduced the Charlie card (like Oyster), which makes user journeys traceable
- Richard Stallman hated this
- Introduced Charlie card parties: Attendees exchange zero valued Charlie cards to confound Big Brother

Pseudonym parties [FS08]

- Step 2: confound Big Brother **&** prevent Sybils
- One day designated for "Pseudonym party"
- Users obtain/exchange IDs
  - protected by masks to preserve privacy
- Parties are geographically separated, but synchronously scheduled to prevent user from attending two parties → prevent sybils

Summary

- Social graph = who's connected to whom + what are their features/attributes
- This information is
  a) Sensitive and private
  b) Available in unprecedented detail via OSNs
  c) Incredibly useful: to both good and bad guys

Research Questions

1. How do we keep this data from unsafe disclosure?
   Depends on what data & how apps need to use it
2. How do we exploit this data (in good ways)?
   hinges on properties of social graph (fast mixing, homophily)

Answers will always be proviso a threat model