How far removed are you?

Scalable Privacy-Preserving Estimation of Social Path Length with Social PaL

N. Asokan

joint work with Marcin Nagy, Thanh Bui, Emiliano De Cristofaro, Ahmad-Reza Sadeghi, Jörg Ott
Problem

How can you find if you have common friends with someone (nearby)?

… in a privacy-preserving way
Applications

- Intuitive means for specifying access control
  - Ride sharing
  - Tethering Internet access
  - ...

- Information
  - Friend radar

- ...

...
Requirements

❖ **privacy:**
  ❖ no more info. to participants than about common friends
  ❖ no additional info. to anybody else (e.g., “trusted server”)

❖ **authenticity:**
  ❖ no false claims of friendship

❖ **efficiency:**
  ❖ applicable for mobile usage
  ❖ minimize expensive crypto operations
Current approach: Using a trusted server

- FourSquare, Tencent, ...

<table>
<thead>
<tr>
<th>User</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>(x,y)</td>
</tr>
<tr>
<td>Bob</td>
<td>(x,y)</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Any friends nearby?

Any friends nearby?

Privacy ✗
Authenticity ✓
Efficiency ✓

Alice

Bob

“Alice”
Alternative: Private Set Intersection Protocols

Secure in the honest-but-curious model
$O(|S_I|+|S_R|)$ modular exponentiations

[De Cristofaro et al, FC’10, Asiacrypt ’10, CANS’12]
Finding Common Friends using PSI naively

Friend ID
Carol
Bob
Tom
Alice
“Carol”
Efficiency ×
Authenticity ×
Privacy ?
Approach

• Make use of widely deployed online social networks
  – user authentication, social graph
• But don’t cede even more information to them
Finding Common Friends using PSI with capabilities

1. Distribute (short-lived) bearer capability to friends

2. Private Set Intersection on capability sets to find common friends

<table>
<thead>
<tr>
<th>User</th>
<th>Capability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Carol</td>
<td>●</td>
</tr>
<tr>
<td>Tom</td>
<td>●●</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Privacy: ✓
Authenticity: ✓
Efficiency: ✗
Can we build a fast “PSI”?  

• Why are classic PSIs slow?  
  ✷ Designed to work even when input sets are enumerable  
    ✷ i.e., elements are predictable  
  ✷ Naive hash-each-element approach fast, but insecure for enumerable input sets  

• However, bearer capabilities are random  
  ✷ Hash-each-element approach is safe  
  ✷ Still O(n) communication complexity  

✦ Idea: use a Bloom Filter to represent input set
What is a Bloom Filter?

Efficient data structure for testing set membership
Map each element to k positions in a bit vector

No false negatives; false positives possible

Bloom Filter PSI Protocol

Challenges
- Insecure channel
- Man-in-the-middle
- False positives

Initiator I
- Secure channel establishment
- Channel binding
- Insert elements into BF
- Check each element for presence in BF
- False positives removal e.g., challenge-response

Responder R

Privacy ✓
Authenticity ✓
Efficiency ✓

Not a replacement for PSI in general!

“Common Friends”: Nagy et al, ACSAC 2013
Comparison: execution time

A protocol execution time comparison is shown, with the protocol execution time in seconds. The chart includes data for PSI-CA and BFPSI with different sizes of input sets (100, 200, 300, 400, 500). The chart is divided into communication and computation time.

The data is an average of 30 runs.
Two challenges with Common Friends

• Bootstrapping is a problem

• Limited to social paths of length 2
Bootstrapping the system

- Only participating users upload capabilities

The system can only find common friends who are participating in the system
Fixing bootstrapping: Ersatz profiles

Assumption:
1. App server may query Social Network for list of friends of a participating user

Have App server create replacements for missing profiles
1. Identify friends of participating users
2. Create/maintain capabilities for those missing

Ersatz profile = Social Network identity + server generated capability
Fixing bootstrapping: Ersatz profiles

(1) Bob uploads capability 
(Bob,  )

(2) Server retrieves Bob’s friends
Friends(Bob) = {Carol, John}

(3) Server generates ersatz profiles for missing users
User | Capability
--- | ---
Carol |  
Bob |  
John |  

(4) Bob downloads capabilities
(Carol,  )
(John,  )
Fixing bootstrapping: Ersatz profiles

With ersatz profiles all common friends are always discovered
Finding lengths of longer social paths

How can you find your social graph “distance” to someone (nearby)?

… in a privacy-preserving way

Social PaL: Social Path Length Finder
More applications

- Intuitive means for specifying access control
  - Ride sharing
  - Tethering Internet access
  - ...

- Information
  - Friend radar

- Routing in “dense” ad-hoc environment
- Place familiarity estimation
- ...

...
Social Path Length

Definition:

**minimum** number of hops in social graph between two users
Additional requirements

- **Privacy:**
  - Two users can’t learn more than by gathering information using standard social network interfaces available to them

- **Functional:**
  - Maximize number of paths discovered between two users
  - Determine exact path length between two users
Capabilities as path length proofs

Intuition:
1. Capability distributed to friends used as friendship proof
2. Use hash chains to generate higher order capabilities

From capability c generate $i^{th}$ order capability:

$$h^i(c) = c^i$$

1. Distribute $c^i$ to contacts $i+1$ hops away
2. Recipient includes $c^i$, $c^{i+1}$, …, $c^n$ in input to PSI
Social PaL graph building

Social PaL only learns friend lists of actual users
  - users explicitly authorize Social PaL

If relationships in the social network are reciprocal
  - Partial view of friend lists of ersatz profiles possible
Social PaL capability distribution

1. Friends’ capabilities returned with identities
2. Higher order capabilities returned w/o identities
Social PaL path length discovery

D(Bob, Alice) = 4

D(Bob, Thomas) = 3
Coverage of social path discovery

- **Theorem:** If Social PaL discovers a path between A and B, then both A and B can determine its **exact length**.

- **Coverage:** probability that A & B will discover a k-hop path that exists between them in the social network.
Dataset for estimating coverage

• Social Filter dataset
  – By Sirivanos et al
  – Derived from dataset by Gjoka et al (UC Irvine)
  – 500 000 users; 30 connections on average
Simulation for estimating coverage

1. Test set: randomly choose x% of users
   - $x = 20, 40, 60, 80$ (represents fraction using Social PaL)
2. Pick 50k pairs randomly from “test set” w/ k-hop path
   - $k = 2, 3, 4$
3. Compute fraction for which Social PaL discovers path

Repeat steps 1-3 ten times; average results
Coverage: Social Filter dataset

![Graph showing coverage of SocialPal discovery versus fraction of OSN with SocialPaL](image)
Datasets for estimating coverage

• Social Filter dataset
  – By Sirivanos et al
  – Derived from dataset by Gjoka et al (UC Irvine)
  – 500 000 users; 30 connections on average
  – Sampling did not preserve node degree
Dataset for estimating coverage

- MHRW dataset
  - Sampled using Metropolis Hastings random walk
  - 95,700 “sampled users”, 175 connections on average
  - 72.2 million “outside users”
  - among sampled users: 3 connections on average

- From Gjoka et al (Infocom 2010)
Simulation for estimating coverage

1. Test set: randomly choose x% of “sampled users”
   - x = 20, 40, 60, 80 (represents fraction using Social Pal)
2. Pick 50k pairs randomly from “test set” w/ k-hop path
   - k = 2, 3, 4
3. Compute fraction for which Social PaL discovers path

Repeat steps 1-3 ten times; average results
Coverage: MHRW dataset (random walk)

X-axis shows fraction of sampled users
Sampled users: 957,000
Outside users: 72.2 million
Dataset for estimating coverage

- BFS dataset
  - Sampled using breadth-first search
  - 2.2 million sampled users, 310 connections on average
  - among sampled users: 53 connections on average
- Also from Gjoka et al (Infocom 2010)
Coverage: Breadth-first search dataset

X-axis shows fraction of sampled users
Sampled users: 2.2 million
Total users: 93.8 million
Coverage analysis summary

• Use of ersatz profiles significantly increases coverage
  – Always 100 % coverage for 2-hop paths (detects all)
  – Only 20% users with Social PaL: coverage > 40%
    • Except for MHRW dataset
  – 80% users with Social PaL: coverage > 80%, always

• Coverage is better in datasets with higher connectivity
  – BFS dataset ~ Social Network in regions with high penetration

• 4-hop paths more readily discovered than 3-hop paths!
Example App: nearbyPeople

nearbyPeople
Example App: SpotShare

SpotShare (Google Play)
Summary

• Privacy-preserving, scalable protocols for finding
  – common friends
  – lengths of social paths
• Used in two applications (available for download)
  – Easy-to-use tethering (“SpotShare”)
  – Friend radar (“nearbyPeople”)
• Source code available for research use
• More info at https://se-sy.org/projects/pet/