Private Social Network Analysis: How to Assemble Pieces of a Graph Privately

Keith B. Frikken
Miami University

Philippe Golle
Palo Alto Research Center

Motivation

- Graph Model fits nicely to
  - Distributed systems
  - Peer-to-peer networks
  - Social networks
- They are used for
  - Marketing
  - Epidemiology
  - Psychology
- However
  - Knowledge of the whole graph is usually distributed
  - Out of privacy concern, a subject can refuse to provide his local knowledge of the graph

Solution

- Protocols to reconstruct the whole graph privately
  - Hide the correspondence between the nodes and edges in the graph and the real-life entities and relationships that they represent.
  - Assume subjects can be malicious in order to de-anonymize the reconstructed graph.
  - Severely restrict the ability of adversaries to compromise the privacy of the honest subjects.

Pseudonyms: a simple solution

- The approach (an example)
  - Each student chooses a pseudonym and lets her friends know the pseudonym.
  - Friendship links can be reported to the authorities pseudonymously.
  - The authorities construct the friendship graph on vertices labeled with pseudonyms.
  - Strip the vertices of their pseudonymous labels and publish the resulting anonymized graph.

Pseudonyms: the limitations

- The authorities can still learn the relationship between pseudonyms and real identities.
- A malicious student, in collusion with one or more malicious authorities, can learn which node in the anonymized graph corresponds to his pseudonym.
- He can then learn which nodes represent his friends, the friends of his friends, etc.

Model

- A set S of subjects, denoted $S= \{S_1,S_2,..,S_n\}$
- Each S_i has a set R_i which contains all subjects to which S_i has a relationship.
- These R-sets imply a graph $G=(V,E)$ where $V=\{S_1,\ldots,S_n\}$ and $\{S_i,S_j\}$ belongs to E if and only if S_j belongs to R_i.
- The graph is directed, but the protocols proposed can be trivially extended to undirected graphs.
- Distributed graph: assume the graph must be reconstructed from partial knowledge of edges distributed among the nodes.

Goal

- To learn a graph AG (i.e., anonymized graph) that is isomorphic to G without revealing the isomorphism that relates AG to G.
- AG is constructed by a number of authorities, none of whom (below a quorum) should be able to learn the isomorphism.

Cryptographic building blocks

- Threshold ElGamal encryption
  - The authorities hold shares of the corresponding decryption key, such that a quorum consisting of all parties can decrypt.
- ElGamal re-encryption

In ElGamal encryption, we consider the usual group of integers $\mathbb{Z}_p^*$ under multiplication and g, a generator of $\mathbb{Z}_p^*$.

$x \in \mathbb{Z}_p^*$ is the secret key.

$y = g^x$ is the public key.

$E(m) = (g^r, my^r)$ is the randomized encryption function with $r \xleftarrow{R} \mathbb{Z}_{p-1}^*$.

$D(c_1, c_2) = (c_1^x)^{-1} * c_2$ is the decryption function on an ElGamal pair.

(Note how the randomized factor is divided out: as long as the pair is a correct ElGamal pair, decryption is straight-forward).

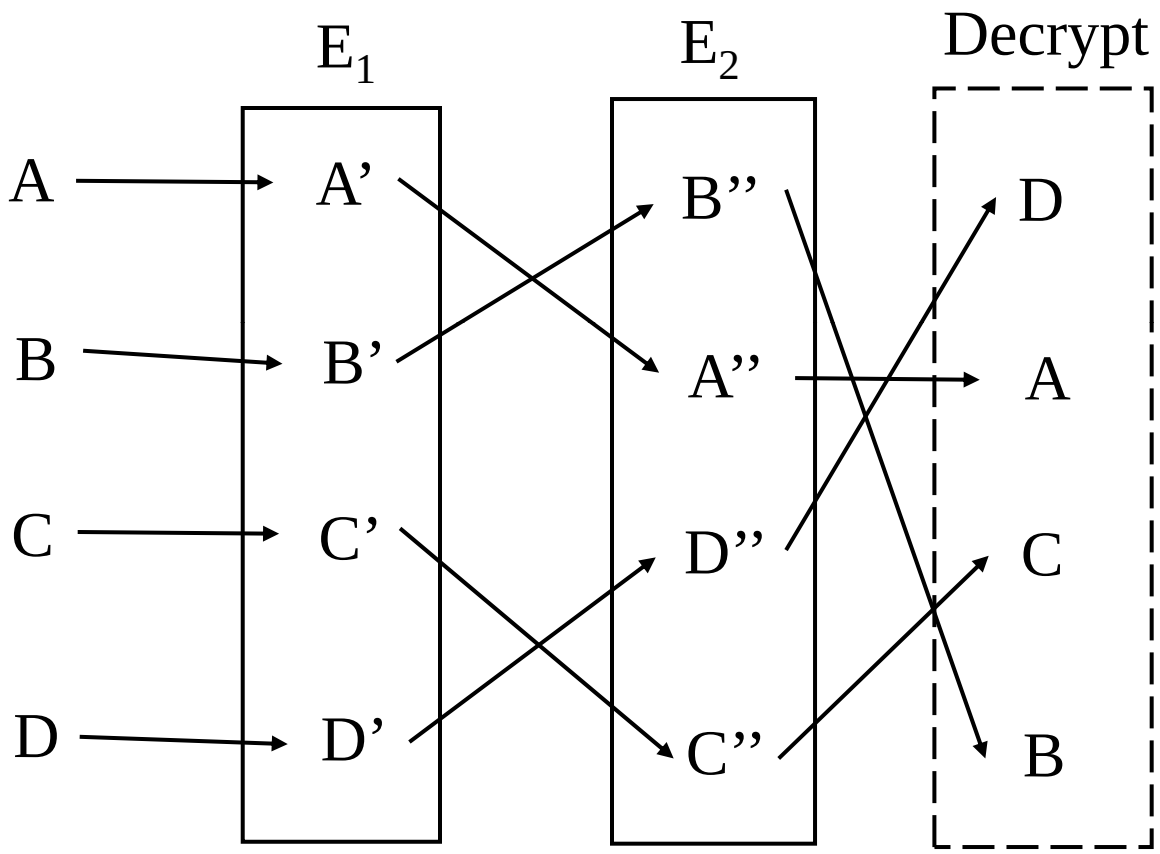
We now describe ElGamal reencryption (i.e. re-randomization):

$ReEnc(c_1, c_2) = (c_1 * g^s, c_2 * y^s) = (g^{(r+s)}, my^{(r+s)})$ for $s \xleftarrow{R} \mathbb{Z}_{p-1}^*$.

The reencryption does not affect the decryption process, nor does it require knowledge of the secret key!

Cryptographic building-blocks

- Re-encryption mix network



Cryptographic building blocks

- Re-encryption mix network

We are now ready to define the El-Gamal based re-encryption mix net:

1. An El-Gamal public-key (p, g, y) is generated (in some distributed manner).

2. The initial encryption phase E simply encrypts all the ballots $B_1, \ldots, B_n$ by applying the El-Gamal encryption algorithm with the public-key (p, g, y). It then posts all the resulting ciphertexts $(C_{1,0}, \ldots, C_{n,0})$ on a bulletin board.

3. The i'th mix phase, on input a set of ciphertexts $(C_{1,i-1}, \ldots, C_{n,i-1})$, re-encrypts each ciphertext and permutes the resulting ciphertexts using a secretly chosen random permutation.

4. The final decryption phase D, given a set of ciphertexts $(C_{1,k}, \ldots, C_{n,k})$, simply decrypts all the ciphertexts in some distributed manner (in order to achieve robustness).

Cryptographic building blocks

- Mixing ciphertext tuples
  - Input a batch of vectors
  - A vector consists of multiple ciphertexts as its components
  - Mix servers re-encrypt individually every ciphertext component
  - Mix servers mix the vectors
  - The order of the components (ciphertexts) within a vector is fixed.
- Oblivious test of plaintext equality
  - $E(m_1)$ and $E(m_2)$ be two ElGamal ciphertexts
  - The joint holders of the decryption key can check whether $m_1=m_2$ via Jakobsson and Schnorr protocol without revealing any other information (e.g. what's the actual plaintext)

The Protocol

- Setup
  - The authorities create a list of n ciphertexts E(1),…,E(n).
  - They mix the list to obtain a permuted list: $E(\pi(1)), \ldots, E(\pi(n))$
  - They then append n copies of the value E(-1) to the end of the list to form a list L of 2n ciphertext elements.
  - L is posted to a bulletin board accessible by the subjects / authorities.

The Protocol

- Data Collection. Subject S_i does the following steps:

1. S_i chooses a random permutation σ_i (over $2n$ items) such that for $1 \leq j \leq n$:

   - If $S_j \in R_i$, then $\sigma_i^{-1}(j) \in [1, n]$.
   - If $S_j \notin R_i$, then $\sigma_i^{-1}(j) \in [n+1, 2n]$

   Note that such a permutation always exists. The permutation σ_i maps to the range $[1, n]$ the encrypted nyms of all subjects to whom S_i wants to report a relationship. When S_i has fewer than n relationships, σ_i also maps "dummy" entries (i.e., encryptions of -1) to the range $[1, n]$. The encrypted nyms of subjects to whom S_i does not want to report a relationship are mapped to the range $[n+1, 2n]$, as are left-over dummy entries (if any).

The Protocol

2. S_i mixes L with σ_i and submits this to the authorities along with a proof of proper mixing. We denote the list submitted by S_i as $L_i = L[\sigma_i(1)], \ldots, L[\sigma_i(2n)]$.

The Protocol

- Graph Reconstruction
  - To reconstruct the graph the authorities create a set T and initialize to the empty set. They then do the following:

1. For each subject S_i:
   (a) The authorities verify the proof of correct mixing. If the proof is invalid, the authorities discard L_i and continue without S_i's values.
   (b) They create n tuples: $\left(E(\pi(i)) \ ; \ L[\sigma_i(j)]\right)$ for $1 \leq j \leq n$, and they add all of these tuples to T.
2. The authorities mix the list of tuples in T and prove correct mixing to obtain a new list of tuples T'.

The Protocol

3. For each tuple $\Big(E(x)\ ;\ E(y)\Big)$ in T', the authorities jointly decrypt $E(y)$. If the value is not in $[1, n]$, they discard the tuple. Otherwise, they jointly decrypt $E(x)$ and record an edge from x to y in AG.

Security Properties

- The number of edges that a subject reports is hidden from an adversary.
- Subjects do not learn their own nym or other participants' nyms as part of the setup process.
- A subject can report only edges from itself to other subjects
- Multi-edges are disallowed.
- Does not require any form of consent to report an edge to an subject.

Deal with multi-edges

- Setup. (no difference)
- Data Collection.
  - In addition to the previously submitted information, n sets of m+1 encrypted values are submitted. Subject S_i constructs its j^{th} set of values as follows:

1. Suppose $\sigma_i(j) = k$. If $k > n$ then the $m+1$ values are encryptions of randomly chosen values. If $k < n$, and S_i wants to report $\bar{m} \leq m$ edges with S_k, with edge values $v_1, \ldots, v_{\bar{m}}$, then the jth set consists of the values $E(\bar{m}), E(v_1), \ldots, E(v_m)$ followed by $m - \bar{m}$ encryptions of random values.

2. Along with the list of $m+1$ values, S_i submits a proof of plaintext knowledge for each of these values.

Deal with multi-edges

- Graph Reconstruction

1. The authorities verify all proofs provided by subject S_i. If any proof fails then S_i's tuples are discarded.
2. For every valid tuple $\left(E(y)\ ;\ E(v_0)\ ;\ \ldots\ ;\ E(v_m)\right)$ that subject S_i reports, the authorities build a tuple $\left(E(\pi(i))\ ;\ E(y)\ ;\ E(v_0)\ ;\ \ldots\ ;\ E(v_m)\right)$
3. The authorities mix all tuples and verify correct mixing.

Deal with multi-edges

4. For each tuple $\Big(E(x)\ ;\ E(y)\ ;\ E(v_0)\ ;\ \ldots\ ;\ E(v_m)\Big)$ the authorities do the following:

   (a) They decrypt v_0. If it is not in $[1, m]$, they discard the tuple.

   (b) They decrypt the values $v_1, \ldots, v_{v_0}$. If any of these values is not a valid edge value then they discard this tuple.

   (c) They decrypt y. If $y = -1$, they discard the tuple.

   (d) They decrypt x and add v_0 edges from x to y in AG with respective values $v_1, \ldots, v_{v_0}$.

Consent

Setup. Same as in Section 5.1.

Subject Setup. Suppose subjects S_i and S_j would like to report an edge from S_i to S_j. They agree on a random value $r_{i,j}$ chosen uniformly from a range large enough that collisions between values generated by distinct pairs of nodes are highly unlikely. For example, $r_{i,j}$ could be a 160-bit integer.

Consent

Data Collection. Subject S_i does the following steps:

1. It creates permutations $\sigma_{i,out}$ and $\sigma_{i,in}$ to represent the sets $R_{i,out}$ and $R_{i,in}$ respectively using Step 1 of the protocol in section 5.1. It permutes L using $\sigma_{i,out}$ and $\sigma_{i,in}$ to obtain $L_{i,out}$ and $L_{i,in}$ respectively. It also generates proofs of proper mixing.

2. It generates n encrypted values $E(q_{i,1,out}), \ldots, E(q_{i,n,out})$ where $q_{i,j,out} = r_{i,k}$ (from the subject setup phase) if $\sigma_{i,out}(j) = k$ and $1 \leq k \leq n$ and is a random value otherwise. It submits these values to the authorities along with proofs of plaintext knowledge.

Consent

3. It generates n encrypted values $E(q_{i,1,in}), \ldots, E(q_{i,n,in})$ where $q_{i,j,in} = r_{k,i}$ (from the subject setup phase) if $\sigma_{i,in}(j) = k$ and $1 \leq k \leq n$ and is a random value otherwise. It submits these values to the authorities along with proofs of plaintext knowledge.

Consent

Graph Reconstruction. To reconstruct the graph, the authorities create two sets T_1 and T_2 and initialize them to $\emptyset$. They then do the following:

1. For each subject S_i:

   (a) The authorities check all proofs (of proper mixing and of plaintext knowledge). If any of the proofs fail for a subject S_i then all of S_i's values are discarded.

   (b) For each item in $L_{i,out}[\ell]$ $(1 \leq \ell \leq n)$, the authorities build a triple $\Big(E(q_{i,\ell,out})\ ;\ E(\pi(i))\ ;\ L_{i,out}[\ell]\Big)$ and add this triple to T_1.

   (c) For each item $L_{i,in}[\ell]$ $(1 \leq \ell \leq n)$, the authorities build a triple $\Big(E(q_{i,\ell,in})\ ;\ L_{i,in}[\ell]\ ;\ E(\pi(i))\Big)$ and add this triple to T_1.

Consent

2. They mix the tuples in T_1, and verify correct mixing (call the new list of tuples T_1').

3. They find all tuples with matching r values (i.e., tuples that were reported by two parties), and combine the tuples' information. The authorities decrypt the first element of every tuple and find all matching values. The authorities discard all tuples that do not match any other tuple or that match more than one tuple in the first element.

Consent

The authorities are left with pairs of tuples that match on the first element:

$$\Big(r_{m,n}\ ;\ E(\pi(m_1))\ ;\ E(\pi(n_1))\Big) \text{and}$$

$$\Big(r_{n,m}\ ;\ E(\pi(m_2))\ ;\ E(\pi(n_2))\Big),$$

where $r_{m,n} = r_{n,m}$. They then check that $E(\pi(m_1))$ and $E(\pi(m_2))$ decrypt to the same plaintext, using the oblivious test of plaintext equality. Similarly, the authorities check that $E(\pi(n_1))$ and $E(\pi(n_2))$ decrypt to the same plaintext. If either of these checks fail then both tuples are discarded. If both checks succeed, then the authorities build an ordered pair

$\Big(E(\pi(m_1))\ ;\ E(\pi(n_1))\Big)$ and adds it the the set T_2.

Consent

4. They mix the tuples in T_2, and verify correct mixing (call the new list of tuples T_2').

5. A quorum of authorities then decrypt the pairs in T_2' of the form $\Big(E(x)\ ;\ E(y)\Big)$ and register an edge from x to y in AG.

Conclusion

The complex graphs that represent connections in distributed systems, such as social networks, online communities or peer-to-peer networks are of tremendous interest and value to scientists in fields as varied as marketing, epidemiology and psychology. However, knowledge of these graphs is typically distributed among a large number of subjects, each of whom knows only a small piece of the graph. Privacy concerns make these subjects reluctant to share their local knowledge of the graph.

This paper studies the problem of assembling pieces of a graph privately. We define what it means to reconstruct a graph privately, propose a threat model and cryptographic protocols that allow a group of authorities to jointly reconstruct a private graph.

In future work, we hope to implement these protocols and demonstrate their value with simulations. We hope these protocols will help collect data that privacy concerns previously made difficult or impossible to collect.

Forensic Genomics: Kin Privacy, Driftnets and Other Open Questions

Frank Stajano
University of Cambridge
Computer Laboratory
15 JJ Thomson Avenue
Cambridge CB3 0FD
United Kingdom
fms27@cam.ac.uk

Lucia Bianchi
Studio Legale Bianchi
Via G. di Vittorio
S. Casciano Val di Pesa
50026 Firenze
Italy
luciagbianchi@gmail.com

Pietro Liò
University of Cambridge
Computer Laboratory
15 JJ Thomson Avenue
Cambridge CB3 0FD
United Kingdom
pietro.lio@cl.cam.ac.uk

Douwe Korff
London Metropolitan
University
Dept of Law, Governance and
International Relations
Ladbroke House
62-66 Highbury Grove
London N5 2AD
United Kingdom
d.korff@londonmet.ac.uk

Motivation

- Nowadays, genetic analysis of human samples is commonplace, both in forensics and in medicine.
- With the advancement of technology, we are seeing switch from *genetics* to *genomics*.
  - From a few thousand base pairs to 3 billion base pairs (the entire genome of a person)
- Nobel prize winner James Watson has his genome recorded on two DVDs. The sequencing effort took only two months and just under one million dollars.
- With the easy access to human genomes, a lot of privacy concerns emerge.

Things that might one day happen

1. A private investigator collects some of the biological samples you leave daily anywhere you go (hairs, skin flakes etc) and obtains a full copy of your genome for his client—who then learns, among many other things, your susceptibility to Alzheimer's disease, cancer, food allergies, intolerance to chemicals and so forth, and whether you really are the father of your daughter.

Things that might happen one day

2. Governments hold complete genomic databases of all their citizens, for combined reasons of national health and national security. This is considered sensitive personal information. However, tens of thousands of clerks have access to the databases. Every now and then, one of them loses a laptop with all the data, or a set of disks with an unencrypted copy of the database goes missing in the post. (They've already shown they're good at that [20, 21, 22, 42].)

Things that might happen one day

3. Full genomic data of every individual is publicly available for medical research, but is claimed to have been anonymized. However, advances in computing one day allow anyone to run a simulation showing what any given genome would turn into, when growing as a full human body. There's even a slider for the age: see the person at 5 years old, 10, 20, 50, 80... Stalkers de-anonymize the genome of celebrities by recognizing them from this reconstruction. The secret police does the same with dissidents.

Things that might happen one day

4. Insurers demand genetic prescreening before offering medical insurance. (In the US, GINA aims to prevent that; see section 5.4.) Even a national health service might require prescreening because emergency (and then chronic) treatment for a disease that could have been prevented is an unnecessary cost to society.

Things that might happen one day

5. Your genome, like everyone else's, ends up being public knowledge. You are not able to get a date with the girl of your dreams because your DNA indicates you are not a good enough prospect.

6. Everyone, at birth, to ensure better medical assistance throughout their lifetime, has their genome sequenced and stored in a central database. A central refrigerated facility also stores some of their stem cells, to be used for regeneration of damaged organs. Given the history of security violations that have plagued any such large centralized facilities, it is unclear how unauthorized access to the database and stem cell bank will be prevented.

Things that might happen one day

7. China's one-child policy and India's dowry traditions have already caused countless instances of female infanticide due to the preference of the parents for a baby boy heir. Once genome sequencing at birth is commonplace, the same mind-set naturally leads to eugenetic infanticide: the killing at birth of any babies rated "not good enough"—for instance, those genetically more likely to develop certain diseases in later life.

Things that might happen one day

8. KIN PRIVACY. The availability of personal genomes may provide a huge number of forensic markers. In theory each nucleotide in a genome can be used as a molecular marker. Close and distant relationships can be detected (see for instance the Romanov [4] and the Ashkenazi Jews [49] cases). Two aspects should be mentioned. First, compared to today's methods, using many more markers would extend the kinship that can provide valuable information in a database matching procedure. Secondly, reading the different parts of the available genomes would provide valuable hypotheses on the medical and physical characteristics of the individual ("it's probably a diabetic male with red hair"). These two aspects are not independent and may act synergistically, strengthening each other.