Deanonymizing Mobility Traces: Using Social Network as a Side-Channel

ABSTRACT

Location-based services, which employ data from smartphones, vehicles, etc., are growing in popularity. To reduce the threat that shared location data poses to a user's privacy, some services anonymize or obfuscate this data. In this paper, we show these methods can be effectively defeated: a set of location traces can be deanonymized given an easily obtained social network graph. The key idea of our approach is that a user may be identified by those she meets: a *contact graph* identifying meetings between anonymized users in a set of traces can be structurally correlated with a social network graph, thereby identifying anonymized users. We demonstrate the effectiveness of our approach using three real world datasets: University of St Andrews mobility trace and social network (27 nodes each), SmallBlue contact trace and Facebook social network (125 nodes), and Infocom 2006 bluetooth contact traces and conference attendees' DBLP social network (78 nodes). Our experiments show that 80% of users are identified precisely, while only 8% are identified incorrectly, with the remainder mapped to a small set of users.

1. INTRODUCTION

Applications that employ sensor data (e.g., location, acceleration) generated by everyday mobile devices such as smartphones, tablets, and cars have become quite popular [3, 11, 15, 21, 31]. As examples, CarTel [15] collects location information from GPS sensors in cars to infer traffic conditions, and GreenGPS [11] computes fuel-optimal routes from OBD-II sensors installed in cars. Human mobility patterns have been used for urban planning and traffic forecasting [13]. While useful, these services pose a risk: a user's mobility trace, if revealed, can provide information about habits, interests and activities—or anomalies to them—which in turn may exploited for gain via theft [1], blackmail, or even physical violence.

To reduce the risk of exposing too much information, those who would collect mobility traces promise to *anonymize* them by removing personally identifying information (PII), such as name, address, or birthday. Unfortunately, work in other domains has shown that such data can nevertheless by recovered by an adversary with *auxiliary information* [23]; this recovery process is called *deanonymization*. The key question is: in the case of mobility traces does there exist readily available auxiliary information with which an adversary could effectively perform deanonymization? In this paper, we show that the answer is 'yes.'

Mobility traces can be deanonymized by exploiting the *social network* of the participating users. Such social networks are readily available: friend relationships can be found from public Facebook data, co-authorship relationships from DBLP, and business relationships from LinkedIn. The key insight is that a pattern of meetings between users suggests they have relationship; this relationship may be mirrored in their social network and thus the social network can be used to recover PII removed from a trace.

Our technique involves three main steps. First, we combine the mobility traces of multiple users into a *contact graph*, where nodes are (anonymized) users and edges indicate one or more meetings between them. An edge weight indicates meeting frequency and other timing data. Now we have reduced the problem of deanonymization to that of constructing a mapping between the contact graph and the social network graph, whose nodes are (non-anonymized) individuals and whose edges indicate a relationship between them (friend, colleague, co-author, etc.). However, on large graphs constructing this mapping could be computationally infeasible; it is at least as difficult as the NP-hard graph isomorphism problem.

Hence, in the second step we bootstrap the mapping problem by identifying *landmark* nodes in both the social network and the contact graph using a node centrality measure [24, 7]. We consider all $k!$ mappings in which the top k central nodes in one graph are mapped to the top k in the other (for small k, e.g., $k = 3$). We use a novel measure of centrality for contact graph to reflect the fact that contacts occur over time (unlike a social network whose relationships may be stable over longer periods).

Third, for each possible mapping between landmark nodes of the two graphs, we attempt to map the remaining nodes. We consider three possible techniques. The first is based on distance vectors to landmark nodes—we map nodes with the smallest difference between their distance vectors. The second technique is based on edit distance between randomly chosen spanning trees—we compute n spanning trees of each graph and choose the pair of trees with the smallest distance between them, then use the mapping they induce.

The last technique, and the most effective as it turns out, works by comparing features of sub-graphs surrounding deanonymized nodes, proceeding recursively. This technique is implemented as a constraint satisfaction problem: in essence, we use one graph as ground truth that maximizes the correct mapping of edges with the other graph. To see this, consider the simplified example in Figure 1. Let us consider four users, b, c, d, and e, and their respective anonymized user identities b', c', d' and e'. Suppose that we have deanonymized users $b' = b$ and $c' = c$ (e.g., they are the landmark nodes). Let $f(b)$ denote the set of b's friends in the social network; then $D = f(b) \cap f(c)$ is the set of friends b and c have in common, while $E = f(b) \setminus f(c)$ is the set b's friends not shared by c. Since for the graph in Figure 1 D is a singleton $\{d\}$, there is a unique mapping from d' to d. We say that features D and E are *discriminative* when $D \cap E = \emptyset$, or, in general, when the intersection is

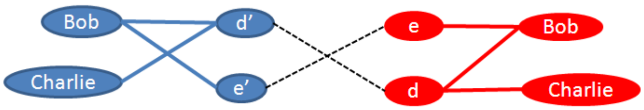

Figure 1: Deanonymization requires Discriminative Features: the Contact Graph is on the left, the Social Network on the right

very small. That is, $D = f(b) \cap f(c)$ and $E = f(b) \setminus f(c)$ act as features over the social network that can be used to discriminate between users d and e. If on the other hand both d' and e' exhibited identical contact patterns with b' and c', we will not be able to discriminate user d from user e. Hence, both the mappings $\{d = d'$ and $e = e'\}$ and $\{d = e'$ and $e = d'\}$ would seem equally likely from the point of view of the deanonymization algorithm.

Mike says: Change the Figure 1 to replace Bob and Charlie with b and c on the right and b' and c' on the left, and draw a directed edge from b' to b and c' to c. Add arrows to the end of the dashed lines in the middle. The idea is to make the directed edges indicate the mapping, with the solid, undirected edges as the friends relationship.

We validate our approach using three real world datasets: University of St Andrews mobility traces and social network (27 nodes each), SmallBlue contact traces and Facebook social network (125 nodes), and Infocom 2006 bluetooth contact traces and conference attendee's DBLP social network (78 nodes). Our initial results show that our approach deanonymizes 80% of the nodes in the traces correctly, 8% incorrectly, and the remaining 8% it is able to correctly map a node to a small set of users (no more than three), one of which is the correct one[1]. Mike says: Should we more particularly distinguish the accuracy of the three different techniques?

We also identify an *automorphism*-based upper bound on the efficacy of any graph de-anonymization algorithm. We argue that in the worst case when both the contact graph and the social network are completely connected ($\frac{1}{2}|V|(|V|-1)$ edges where $|V|$ is the number of vertices) or completely disconnected (zero edges) then graph deanonymization is virtually ineffective. However, we show that in practice most contact graphs and social networks exhibit enough heterogeneity to allow de-anonymization to be not only efficient, but also very effective. Using the derived upper bound we find that our proposed algorithms are within 10% of any optimal deanonymization algorithm for our datasets.

The next section sets up the problem more formally and describes our evaluation datasets. Section 3 presents our approach in detail and evaluates it on these datasets. Section 4 evaluates the impact on deanonymization of adding noise to the data and of using inaccurate or incomplete social network graphs. Section 5 discusses related work, and Section 6 concludes.

2. PROBLEM SETUP

2.1 Notation and Terminology

V_S denotes the set of users with each user having a unique identifier. L denotes the set of location identifiers with each location having a unique identifier. $d(l_1, l_2)$ denotes the distance between two locations l_1 and l_2 in L. We write A to denote a location trace of the form $\langle u, l, t\rangle^*$, where $u \in V_S$, $l \in L$ and t is a timestamp.

V_C denotes the set of anonymized user identifiers, i.e., the mapping between the set of users V_S and V_C is unknown. L' denotes the set of obfuscated location identifiers. We remark the location

[1]For the remaining 4% nodes we correctly map the node to a set of four users or more

	St Andrews	Smallblue	Infocom06
Comm network type	WiFi	IM	Bluetooth
No comm nodes	27	125	78
Duration (days)	30	30	4
Granularity (secs)	300	300	120
No contacts	18,241	240,665	182,951
Social network type	Facebook	Facebook	DBLP
No social nodes	27	400	616

Figure 2: Datasets

Similarity Measure	St Andrews	Smallblue	Infocom06
Graph edit distance / Num edges	8 / 62	28 / 375	22 / 212
Common sub-graph / Num nodes	18 / 27	72 / 125	42 /78

Figure 3: Graph Similarity Measures

identifiers may be obfuscated using several techniques including the addition of random noise or generalization (i.e., reducing the granularity of location information). For instance, a location l in L may be perturbed to locations l'_1 and l'_2 in L' with probability p and $(1-p)$ respectively for some $0 \leq p \leq 1$ (random noise); or two locations l_1 and l_2 in L may be mapped to the same location l' in L' (generalization). We remark that the efficacy of deanonymization degrades with the extent of location obfuscation.

A' denotes an anonymized location trace of the form $\langle u', l', t\rangle^*$, where $u' \in V_C$, $l' \in L'$ and t is a time stamp. The goal of deanonymization is to infer the mapping between the set of users in V_S and V_C. As discussed earlier, the technique used for deanonymization depends on the nature of available auxiliary information. In this paper we consider auxiliary information that is represented by a social network $S = (V_S, E_S)$, where an edge in set E_S has the form $\langle u_1, u_2, r_{12}, w_{12}\rangle$ such that u_1, $u_2 \in V_S$ (note that users in the social network correspond to non-anonymous user identifiers), $r_{12} \in R$ an directed relationship between users u_1 and u_2 and R is a finite set of relationship types (e.g., manager, friend, colleague, co-author, etc.). In addition to relationship type we consider two types of annotations w_{12} to the social network links: one that corresponds to the strength of a relationship denoted by a weight between [0, 1] (e.g., normalized number of co-authored papers) and second that corresponds to periodicity at which a relationship is active (e.g., *colleague* relationship is active every day from 9am-5pm).

In order to structurally match (and de-anonymize) a mobility trace against a social network we construct a contact graph C. Intuitively a contact graph captures pair-wise meetings between two users. Given two events $\langle u'_1, l'_1, t_1\rangle$ and $\langle u'_2, l'_2, t_2\rangle$ in an anonymized location trace, we say that there is a contact between users u_1 and u_2 if $d(l'_1, l'_2) \leq \tau_1$ and $|t_1 - t_2| \leq \tau_2$, for some thresholds $\tau_1, \tau_2 \geq 0$. Given a sequence of contacts between two users u'_1 and u'_2 we apply Fourier transform to detect periodicity in such contacts. We construct contact graphs at different time granularities – based on the most dominant periodicity of contacts derived by applying the Fourier transform. The contact graph C thus generated is of the form $C = (V_C, E_C)$, where the edges in E_C are of the form $\langle u'_1, u'_2, c_{12}, w_{12}\rangle$ and the class type c_{12} is obtained by manual classifying contacts based on the time-of-the-day and w_{12} denotes the frequency of contacts within the period of interest.

Given a contact graph C constructed from an anonymized mobility trace and a social network S our goal is to determine mappings between nodes in the contact graph and that in the social network. In certain cases this problem reduces to that of graph isomorphism; however, this is not generally the case. In particular our goal is to look for approximate matching between the contact graph and the social network. Our techniques are designed to be robust to imperfections in auxiliary information (e.g., the social network). In our experiments (Section 4) we introduce three types of imperfections: (i) an out-dated or a skewed social network − e.g., in the Infocom06 trace we build five co-authorship networks using DBLP data from years 2004, 2005, $\cdots$, 2008 respectively, (ii) adding/deleting nodes in the social network − e.g., Infocom06 dataset includes a list of all conference attendees, however, not all of them volunteered for the collection of mobility trace; we introducing spurious nodes by adding a random selection of nodes from the attended-but-not-volunteered list to the social network, and (iii) adding/deleting edges in the social network − e.g., by randomly picking a pair of nodes with probability p and introducing an edge between them.

For ease of presentation, in the following sections of this paper we assume that there is only relationship type in the social network and only class type in the contact graph. Hence, both the social network S and the contact graph C may be uniformly represented as graphs of the form (V, E), where V denotes the set of vertices and E denotes edges of the form $\langle v_1, v_2, w_{12}\rangle$ and $v_1, v_2 \in V$ and w_{12} is a real-valued number.

2.2 Datasets

In this section we briefly review the datasets that were used in our study. A description of these datasets [4, 2, 27] is shown in Figure 2. The St Andrews university dataset captures the WiFi hotspot (within the university) to which a user (a student volunteer) is connected at intervals of 300 seconds. We assume that two users are in contact if they are both connected to the same WiFi hotspot for an interval of time that spans 600 seconds. The contact graph includes a weight − the frequency of inter-user contacts over the duration of the trace. The St Andrews dataset also includes a social network (gathered from Facebook) on the same set of student volunteers. The social network has 0/1 link weights based on two users being a *friend* on the Facebook social network. We note that in our datasets the ground truth, that is, the mapping between nodes in the social network and the contact graph is given to us.

The Smallblue dataset captures contacts between users using an instant messenger on an enterprise network. Two users are said to be contact if they exchange chat messages back and forth with each other. A session of such chat messages between a pair of users constitutes one contact. The contact graph is constructed in a similar manner with link weights that correspond to the frequency of inter-user contacts over the duration of the trace. In addition to the mobility trace the dataset also includes a social network (gathered from Facebook) on a superset of users. The social network has 0/1 link weights based on two users being a *friend* on the Facebook social network.

The Infocom06 dataset is a Bluetooth contact trace of Infocom 2006 conference attendees that volunteered to contribute their mobility information. When two users are within Bluetooth connectivity range (typically less than 10 meters), a Bluetooth connection is established between the user devices. If such a connection lasts for 600 seconds we assume that the users are in contact with each other (we ignored contacts during session breaks and lunch breaks). In addition the dataset also includes a list of 616 attendees (names and affiliations) of Infocom 2006. The volunteers (78 of them) are a strict subset of the conference attendees. We use DBLP (co-authorship database [9]) to construct a social network over the conference attendees − the social network has 0-1 link weights based on the normalized number of co-authored papers between two users: if P_{u_1} and P_{u_2} denote the set of papers that includes user u_1 and u_2 on the author list then the weight of the link between the users is given by $\frac{|P_{u_1} \cap P_{u_2}|}{|P_{u_1} \cup P_{u_2}|}$.

2.3 Structural Similarity

The key hypothesis of our approach is that the social network and the contact graph bear structural similarities. We show that this is indeed the case using three well accepted measures of graph similarity: graph edit distance (minimum number of edges that need to be added and/or deleted for an exact match), maximum common sub-graph (number of vertices in the largest common sub-graph) and node degree distribution. We remark that the graph edit distance measure typically applies to graphs that have identical number of vertices, while the latter measures do not impose such a restriction. In order to simplify the measurement of such graph similarity measures we round-off edge weights in the social network and the contact graph to either 0 or 1.

Figure 3 shows graph similarity measures using the graph edit distance and the maximum common sub-graph measures. Figure 4 shows the degree distributions of the contact graph and the social network using the datasets. These figures show that the contact graph and the social network tend to bear a lot of similarity; in our datasets, the average ratio of graph edit distance to the number of edges is about 10.3% (lower the better), the average ratio maximum common sub-graph to the number of nodes is about 60% (higher the better), the average Kullback and Leibler (KL) symmetrised divergence measure [30] between the node degree distributions is about 0.062 bits (lower the better).

We remark that the existence of such structural similarities is a *necessary but not sufficient* condition for deanonymization. In general structural similarity indicates that there exists at least one good mapping between the contact graph and the social network. However, such a mapping is not guaranteed to be *unique*. Indeed if a node in the contact graph can be mapped to multiple nodes in the social network (or vice-versa) then that effectively reduces the efficacy of deanonymization. Hence, even if the contact graph and the social network may exhibit structural similarity we need *discriminating features* to derive a unique mapping, which is essential for deanonymization (see example in Section 1).

3. GRAPH DE-ANONYMIZATION

3.1 Overview

We explore three approaches de-anonymizing a mobility trace. We assume that for the sake of simplicity we have one contact graph and one social network and our goal is to map nodes in the contact graph to that in the social network. Our overall approach to derive mappings between a contact graph and a social network can be summarized as follows.

First we will detect a set of landmark nodes in both the contact graph and the social network. Landmark nodes serve as *anchor points* (see Section 1), namely, de-anonymized nodes in the contact graph which is used to bootstrap the subsequent steps for deriving all node mappings. In this paper we will explore three solutions to extend landmark node mappings to all node mappings and quantify the efficacy of these solutions on our datasets.

For the first step, we propose to identify such landmark nodes using the node centrality measure [24][7] − a measure of relative importance of a vertex within a graph. Centrality is very well studied metric in both graph theory and social network analysis − to determine how important a person is within a social network or how well-used a road is within an urban network. In this paper we identify outliers, i.e., nodes with high centrality metric. Past work [12] has shown that node centrality in contact graphs and so-

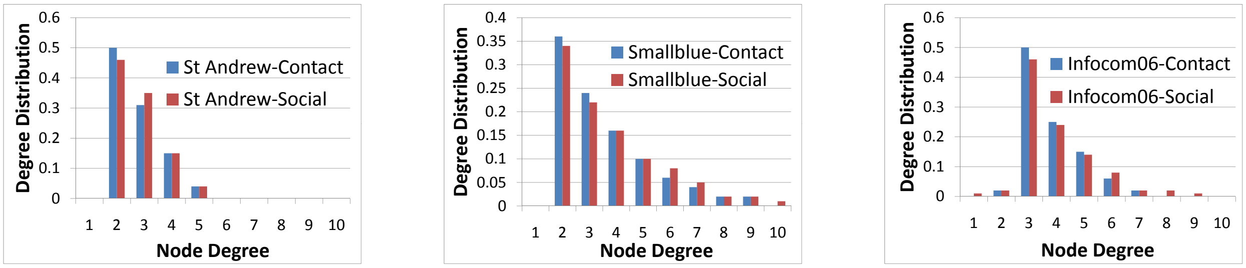

Figure 4: Degree Distribution: St Andrews, Smallblue and Infocom06

cial networks generally follow a heavy tailed distribution; hence, there are a small number of nodes (outliers) with a high centrality score, while a vast majority of the nodes belong to the tail of the distribution. In this paper we propose to identify top-k central nodes in both the contact graph and the social network as landmark nodes (or anchor nodes); typically, $k \ll |V_C|, |V_S|$ – where $|V_C|$ and $|V_S|$ are the number of nodes in the contact graph and the social network respectively. The key intuition behind identifying such highly central nodes is that the set of the top-k central nodes (for some small k) in both the contact graph and the social network are likely to be the same. Further, given k landmark nodes in both the contact graph and the social network there are at most $k!$ mappings between them. Since k is typically small even a brute force enumeration of all possible mappings is feasible.

Given a possible mapping between such landmarks, the subsequent steps use various graph features to deduce mappings for other nodes. In particular we will explore three discriminative graph features: (i) Distance vectors from a node to all the landmark nodes: the key idea here is that a node is characterized by its distance (on both the contact graph and the social network) to the landmark nodes; hence, using a similarity measure between two distance vectors (from a node n in the social network to its landmark nodes and from a node n' in the contact graph to its landmarks) may be used to derive a mapping between nodes n and n'. (ii) Randomized spanning trees: a classical approach to derive mappings between entities in a high dimensional space is to project them to a lower dimensional space wherein it may be easier to identify such mappings; in this approach we will project the contact graph and the social network to spanning trees and use the relative position of a node with respect to landmark nodes to detect node mappings between such spanning trees (note that while graph isomorphism is a NP-hard problem, there are poly-time algorithms to solve the tree isomorphism and tree-to-tree edit problems). (iii) Recursive subgraph features: in this approach we will derive local (sub-graph) features in the neighborhood of the landmark nodes; the key idea is that such local features help us de-anonymize the nodes in the neighborhood of such landmark nodes. As more nodes in the contact graph are de-anonymized we can use them as anchor nodes to construct sub-graph features and essentially propagate the de-anonymization process throughout the contact graph.

In this paper we also report compute times of our solutions on an Intel i5 quad-core processor operating at 2.4 GHz with 4 GB RAM running RedHat Enterprise Linux 5.4.

3.2 Blind Landmarks Selection

In this section we present an approach to identify landmark nodes in both the contact graph and the social network. For the social network we use a node betweenness centrality metric and mark the top-k central nodes as the landmark [24]. Betweenness centrality measure for node n is a normalized measure over the number of uv shortest paths in a graph that includes node n. In general, if a large number of uv shortest paths pass through node n, then node n has a higher betweenness centrality measure.

When deriving a centrality metric for a contact graph, we realize that in a contact graph a 'path' exists over time. A path between two nodes A and B in the contact graph is via a sequence of contacts $N_1, N_2, \cdots, N_{r-1}$. Hence, we adopt the following definition of paths in contact graph and path weight from [12]:

DEFINITION 1. **Opportunistic path** [12]
A r-hop opportunistic path P_{AB} =(V_P, E_P) between nodes A and B consists of a node set $V_P = \{A, N_1, N_2, \cdots, N_{r-1}, B\} \subset V$ and an edge set $E_P = \{e_1, e_2, \cdots, e_r\} \subset E$ with edge weights $\{w_1, w_2, \cdots, w_r\}$. The path weight is the probability $p_{AB}(T)$ that A may reach B along P_{AB} within time T.

We model the inter-contact time X_k between nodes N_k and N_{k+1}, as a random variable, follows a probability density function (PDF) $p_{X_k}(x)$. In our datasets we observed that $p_{X_k}(x)$ was exponentially distributed: $p_{X_k}(x) = w_k e^{-w_k x}$. However, we remark that the approach is applicable to any arbitrary distribution. Assuming that $p_{X_k}(x)$ is exponentially distributed, $Y = \sum_{k=1}^{r} X_k$ following a hypoexponential distribution [26], such that

$$p_Y(x) = \sum_{k=1}^{r} G_k^{(r)} p_{X_k}(x), \quad (1)$$

where the coefficients $G_k^{(r)} = \prod_{s=1, s\neq k}^{r} \frac{w_s}{w_s - w_k}$.

From Eq. (1), the path weight is written as

$$p_{AB}(T) = \int_0^T p_Y(x)dx = \sum_{k=1}^{r} G_k^{(r)} \cdot (1 - e^{-w_k T}), \quad (2)$$

The centrality metric C_i for a node i is then defined as follows:

$$C_i = \frac{1}{N-1} \cdot \sum_{j=1, j\neq i}^{N} p_{ij}(T), \quad (3)$$

where N is the total number of nodes in the network. Intuitively, this metric is a measure of distance from a randomly chosen in the network to node i. Due to the heterogeneity of the pairwise contact frequency in different traces, different values of T were adaptively chosen; T is set as 1 hour for the Infocom traces, 6 hours for the St Andrews and Smallblue traces.

Figures 5 and 6 shows the centrality scores of nodes in the contact graph and the social network respectively. We observe due to the inherent heterogeneity of these graphs, only a few select nodes

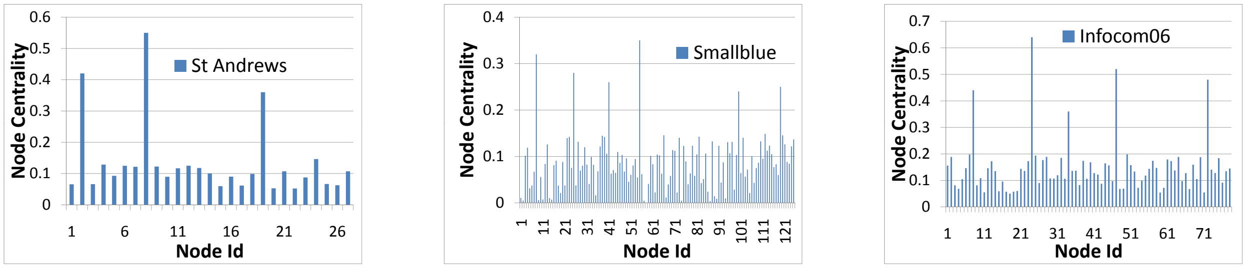

Figure 5: Node Centrality in Contact Graph: St Andrews, Smallblue and Infocom06

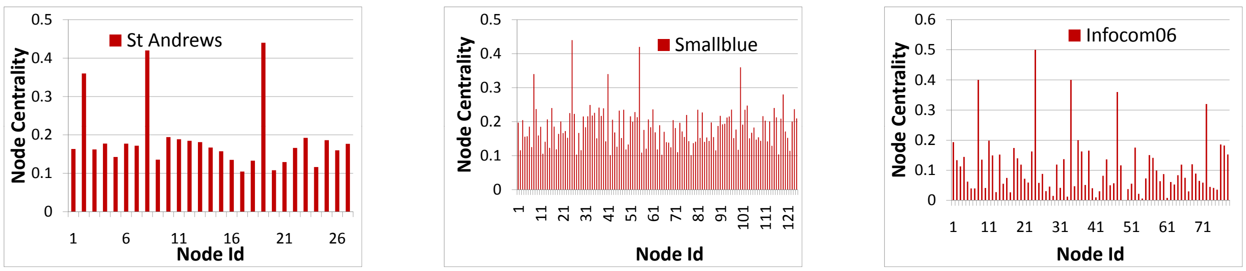

Figure 6: Node Centrality in Social Network: St Andrews, Smallblue and Infocom06. Notice that the peaks (high centrality nodes) in the contact graph (Figure 5) and the social network (Figure 6) match.

have high centrality scores. More interestingly, we observe that the same of set of nodes that have high centrality score in the contact graph also has a high centrality score in the social network. Indeed we note that the set of top-k nodes (ordered by centrality score) is identical for both the contact graph and the social network. For e.g., node identifiers 1, 7 and 18 in the St Andrews dataset are the top 3 central nodes in both the contact graph and the social network; node identifiers 8, 25, 41, 55, 100 and 119 are the top 6 central nodes in the both the contact graph and the social network in the Smallblue dataset; node identifiers 8, 24, 34, 47 and 72 are the top 5 central nodes in the both the contact graph and the social network in the Infocom06 dataset.

Note that at this stage we still do not have a mapping between the landmark nodes; however, given k landmark nodes there are at most $k!$ mappings. In the subsequent sections we propose techniques to start with a mapping of landmark nodes and de-anonymize the rest of the contact graph. We repeat this exercise for all such $k!$ mappings between the landmark nodes; using a goodness-of-fit test on the thus derived mappings we select the most likely mapping between the nodes in the contact graph and the social network.

3.3 Node Feature Vector

In this section we assume that a set of k landmark nodes from both the contact graph and the social network is given to us; we will use $L_C = \{c_1, \cdots, c_k\}$ and $L_S = \{s_1, \cdots, s_k\}$ denote the landmark nodes in the contact graph and the social network respectively with the following mapping: $c_i = s_i$ for all $1 \leq i \leq k$. For each non-landmark node in the contact graph and the social network we derive its feature vector as distances to landmark nodes. Hence, for a node n, its feature vector is given by $\{d_{n1}, \cdots, d_{nk}\}$, where d_{ni} denotes the distance from node n to landmark i (i.e., c_i in the contact graph and s_i in the social network).

Given two nodes c in the contact graph and s in the social net-

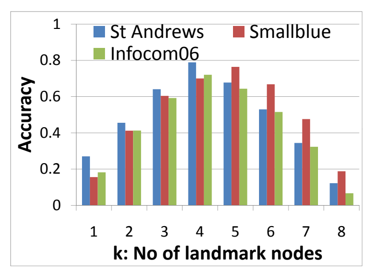

Figure 7: Node Feature Vector

k	St Andrews	Smallblue	Infocom06
1	1.9	195	47
2	2	196	47.4
4	2.2	198	48
8	2.4	200	49

Figure 8: Node Feature Vector: Computation cost (seconds)

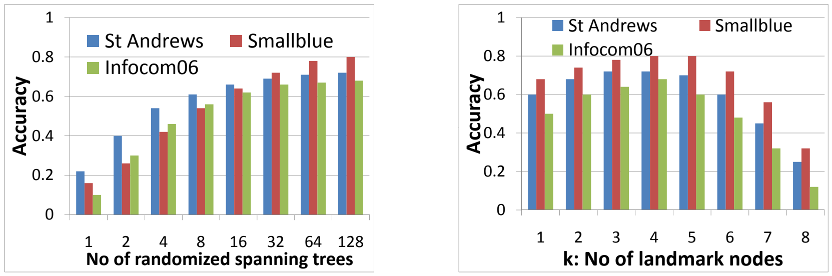

Figure 9: Randomized Spanning Trees

work we quantify a mapping score as $w_{cs} = -\sqrt{\sum_{i=1}^{k}(d_{ci} - d_{si})^2}$. Now we construct a bipartite graph with vertex set $(V_C \setminus L_C) \cup (V_S \setminus L_S)$, where V_C and V_S are the set of vertices in the contact graph and the social network respectively; for every node $c \in V_C \setminus L_C$ and $s \in V_S \setminus L_S$ we add an edge between node c and node s with weight w_{cs}. Now the node mapping problem reduces to a *maximum weighted bipartite graph matching problem* that determines matching pairs of vertices in $V_C \setminus L_C$ and $V_S \setminus L_S$ respectively. We use the Hungarian algorithm that solve the weighted graph matching problem in bipartite graphs in $O(|V|^3)$, where V is the set of vertices and E is the set of edges in the bipartite graph [6]. The result of this algorithm is pairs of matched nodes $s' \in V_C \setminus L_C$ and $s \in V_S \setminus L_S$. We denote the overall matching score as $\sum_{s \in V_S \setminus L_S} w_{s's}$. We repeat this procedure for all $k!$ mappings between landmark nodes and finally select the mapping that results in the highest matching score.

Figure 7 shows the accuracy of de-anonymization as we vary k. We observe that initially as k increases the accuracy of de-anonymization process improves $-$ since larger number of landmarks improves the precision of node feature vectors. However, as k increases further the set of landmark nodes in the contact graph and the social network are no longer identical; hence, the overall efficacy of node mapping decreases.

Figure 8 shows the computation cost of deriving node mappings given a mapping for landmark nodes. We note that given a landmark mapping the computation cost does not increase significantly with k $-$ the number of landmark nodes. As k increases we need to compute distances from more landmark nodes to all the other nodes in both the contact graph and the social network. This operation costs $O(k|E|)$. Once such distances are computed the cost of computing similarity is $O(k|V|^2)$. However, the cost of the weighted bipartite graph matching $O(|V|^3)$ which dominates the computation cost is independent of k. Note that Figure 8 shows the cost for a given landmark node mapping $-$ hence, the overall computation cost has to be scaled with $k!$ (for every possible landmark node mapping). We observe that on the 125 node Smallblue dataset with k = 5 nodes the total time to de-anonymize is 5!*200 seconds or about 6.7 hours.

3.4 Randomized Spanning Trees

In this section we adopt a classical graph isomorphism approach to solve the node mapping problem. However, in order to keep the approach computationally tractable over large contact graphs and socials networks we propose to significantly lower the complexity of the graph mapping by reducing it to randomized tree mapping problem. We note that while in general graph isomorphism is a NP-hard problem, there are poly-time algorithms to solve the tree isomorphism problem [28].

k	St Andrews	Smallblue	Infocom06
1	0.12	2.98	1.08
2	0.13	3.29	1.17
4	0.14	3.51	1.26
8	0.16	4.02	1.44

Figure 10: Randomized Spanning Trees: Computation cost (seconds)

In this solution we project both the contact graph C and the social network S into randomized spanning trees. We note given any graph G the number of spanning trees of graph G is given by Kirchhoff's theorem [14]: $\frac{1}{|V|} * \Pi_{i=1}^{|V|-1} \lambda_i$, where λ_i are the non-zero eigenvalues of G's Laplacian matrix. The Laplacian matrix Q of a graph G is defined as $Q = D - A$, where D is a diagonal matrix with the i^{th} diagonal element set to the degree of node i and A is the adjacency matrix of graph G (i.e., the ij^{th} element in A is 1 if there exists an edge between node i and j; 0 otherwise). Kirchhoff's theorem also allows us to explicitly enumerate all spanning trees of a graph and thus enables us to select a random set of such spanning trees.

Given two such spanning trees C_T and S_T we seed the mappings between landmark nodes and then apply a classical tree-to-tree editing algorithm [28] to derive node mappings between other nodes in the tree. The runtime complexity of the tree-to-tree editing algorithm is $O(|V_C||V_S|)$. We quantify the efficacy of mapping using the graph edit distance between the labeled contact graph and the social network. Over multiple such randomized spanning trees and all $k!$ landmark node mappings we select that mapping that has the least the edit distance between the labeled contact graph and the social network.

Figure 9 shows the accuracy of de-anonymization as we increase the number of randomized spanning trees (with the number of landmark nodes k set to 4) constructed from the contact graph and the social network. The figure also shows the accuracy of de-anonymization as we increase the number of landmark nodes (with the number of randomized spanning trees set to 128). The figure indicates that the accuracy of de-anonymization can be as large as 72%, 80% and 68% respectively for St Andrews, Smallblue and Infocom06 datasets.

Figure 10 shows the tree mapping for a given pair of spanning trees C_T and S_T. We observe that on the 125 node Smallblue dataset with k = 5 nodes and 128 randomized spanning trees, the total time to de-anonymize is 5!*128*4.02 seconds or about 6.2 hours.

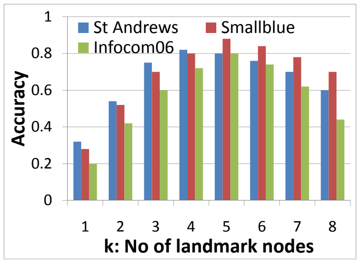

Figure 11: Recursive Sub-Graph Matching

k	St Andrews	Smallblue	Infocom06
1	0.70	12.60	5.25
2	0.72	12.96	5.42
4	0.76	13.68	5.75
8	0.84	15.12	6.36

Figure 12: Recursive Sub-Graph Matching: Computation cost (seconds)

3.5 Recursive Sub-Graph Matching

In this section we propose to use sub-graph features to derive mappings between the contact graph and the social network. We start with the seed set of landmark nodes and recursively expand such mappings to other nodes in the contact graph. The key idea is to model the node mapping as a constraint satisfaction problem (CSP) [22]. We leverage the social network to derive constraints on node mappings as follows. For each node a in the social network we create a variable x_a in the CSP. We constrain all variables x_a to take values from the set V_C, where V_C denotes vertices in the contact graph. If there is a link between users a and b in the social network we introduce conjunctive constraints of the form: $x_a \in c(x_b)$ and $x_b \in c(x_a)$, where $c(x_a)$ denotes the set the of links incident on node x_a in the contact graph. Indeed since the contact graph and the social network may not have an exact mapping the CSP may have no feasible solutions. Hence, we solve for MAX-CSP that minimizes the number of constraint violations (i.e., minimizes the number of unsatisfied constraints).

In general solving such a MAX-CSP is a NP-hard problem. Instead we solve a *dynamic* CSP problem that only introduces constraints of the form $x_a \in c(x_b)$ such that value of x_b has already been determined $-$ the dynamic CSP is bootstrapped using landmark node mappings, i.e., the value of x_{l_i} is assumed to be known for every landmark node l_i. We use a CSP solver in ILOG [16] to recursively expand the set of node mappings from the landmark nodes to span all the nodes in the contact graph. The CSP solver exploits a combination of backtracking, constraint propagation and local search to optimally solve the dynamic MAX-CSP problem. Further, the CSP solver exploits the absence of cyclical dependences in our constraints (note that we introduce a constraints $x_a \in c(x_b)$ only when the value of x_b is known) to scalably solve the node mapping problem.

Figure 11 shows the accuracy of de-anonymization as we increase the number of landmark nodes. The figure indicates that the accuracy of de-anonymization can be as large as 82%, 88% and 80% respectively for St Andrews, Smallblue and Infocom06 datasets. Figure 10 shows the time taken to solve the dynamic MAX-CSP problem for a given landmark node mapping. We observe that on the 125 node Smallblue dataset with k = 5 nodes, the total time to de-anonymize is 5!*15.12 seconds or about 0.5 hours. We note that the computational complexity of recursive sub-graph matching using the dynamic CSP approach (without backtracking) is $O(|V| * max\_node\_degree)$. With limited backtracking (say, at most b local backtrackings) the complexity is still pseudo-polynomial in $|V|$: $O(|V| * max\_node\_degree^b)$.

Figure 13: Graph Automorphism: Completely Connected (left) $\Rightarrow$ cannot infer node mappings and Star Connected (right) $\Rightarrow$ only central node mapping can be inferred

dataset	$\log_2 \lvert aut(C)\rvert$	$\log_2 \lvert aut(S)\rvert$	$\log_2 \lvert V\rvert!$
St Andrews	7.7	7.9	93
Smallblue	18.2	18.4	696
Infocom06	16.5	18.6	382

Figure 14: Number of Automorphisms

3.6 Optimality: Graph Automorphism Bound

In this section we present solutions to analyze the efficacy of graph de-anonymization. First, we make a simple observation that efficacy of node mapping is fundamentally limited by number of graph *automorphisms*. Automorphism of a graph is a form of symmetry in which a graph is mapped onto itself while preserving edge$-$vertex connectivity. Formally, an automorphism of a graph Gă= (V, E) is a permutation σ of the vertex set V, such that the pair of vertices (u, v) form an edge if and only if the pair $(\sigma(u), \sigma(v))$ also form an edge. It is easy to see that no graph de-anonymization algorithm will be able to distinguish between such pairs of vertices (u, v). It may possible to conclude that $(u, v) \in \{u', v'\}$ (i.e., both the mappings $\{u = u'$ and $v = v'\}$ and $\{u = v'$ and $v = u'\}$ are equally likely) but the exact mapping between $\{u, v\}$ and $\{u', v'\}$ may be indeterminate.

For example, if the graph G is completely connected then the number of automorphisms is $|V|!$, i.e., any permutation of vertex labels results in an isomorphic graph (see Figure 13). Similarly, a graph that has no edges also has $|V|!$ automorphisms. Hence, such graphs are completely resilient to graph de-anonymization. Let us consider another example wherein the graph G has a star topology, wherein one vertex v_0 has degree n and all the other vertices $\{v_1, \cdots v_n\}$ have exactly one edge to v_0. In such a graph one can determine the mapping for node v_0, however, any permutations on

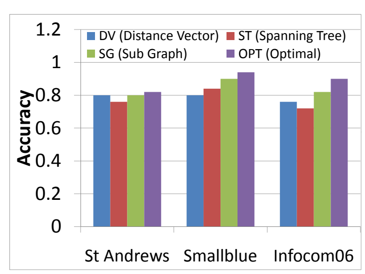

Figure 15: Optimality

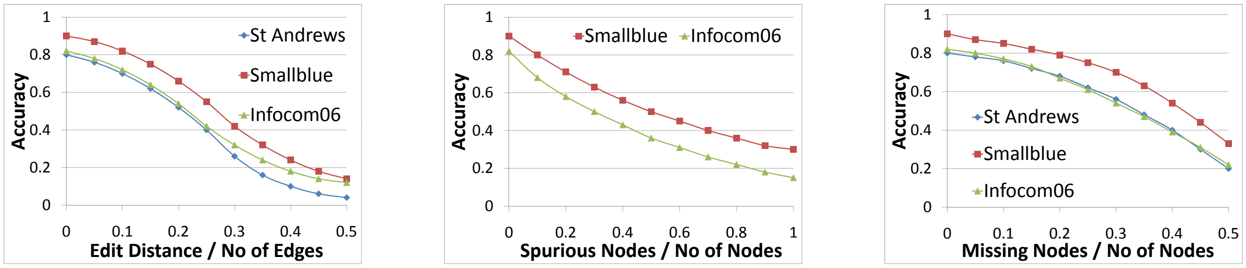

Figure 16: Accuracy with Noisy Social Network

vertex labels $\{v_1, \cdots, v_n\}$ results in an isomorphic graph. Indeed a graph with a star topology has $(|V| - 1)!$ automorphisms.

While graph automorphisms (and its counting version, i.e., determining the number of automorphisms) is a NP-complete problem there are various tools that are very effective in estimating the number of automorphisms for a given graph. For example, Saucy2 [8] has been effective in computing the number of automorphisms in graphs of size ranging from 3K to 5M and number of automorphsisms ranging from $4-10^{8000}$ in under 30 minutes. Figure 14 shows the number of graph automorphisms on both the contact graphs C and the social networks S in our datasets. We note that the maximum possible information gain achievable by any graph denonymization algorithm is given by: $\log_2 |V|! - \log_2 max(|aut(C)|, |aut(S)|)$, where $|aut(G)|$ denotes the number of automorphisms of graph G.

We derive an upper bound on accuracy that may be achieved by any de-anonymization algorithm using the number of graph automorphism. In particular, if the contact graph C has $|aut(C)|$ automorphisms then the accuracy of any de-anonymization algorithm cannot exceed $1 - \frac{\eta}{|V_C|}$, where η is the smallest natural number such that $\eta! \geq |aut(C)|$.

Figure 15 shows a comparison of our algorithms: DV (landmark based distance vectors), ST (randomized spanning trees), SG (recursive sub-graph matching) and OPT (automorphism bound). Our results show that the recursive sub-graph matching approach consistently outperforms the other approaches. The figure also shows that the nodes in the contact graphs exhibit significant heterogeneity: the automorphism bound on St Andrews, Smallblue and Infocom06 datasets is 82%, 94% and 90% respectively. We note that our approach (SG: recursive sub-graph matching) can achieve up to 97.6%, 95.7% and 91.1% of optimality respectively on St Andrews, Smallblue and Infocom06 datasets.

4. EVALUATION: TOLERANCE TO NOISE AND OBFUSCATION

In the following portions of this section we will evaluate the efficacy of our algorithms on various datasets. In particular we will examine the efficacy of various algorithms when: (i) the social network or the contact graph is obfuscated, (ii) a small subset of nodes mappings are known a prior (e.g., insiders reveal their identities or some node mappings are inadvertently leaked), and (iii) examine the efficacy of de-anonymization when stale social networks are used.

4.1 Noisy Social Network

Figure 16 shows the accuracy of node de-anonymization when the social network is obfuscated. We examined three different types of noise that could be added to the social network. First we add/delete spurious edges in the social network – for example, when *edit distance* : *No of edges* is 0.1 then 5% of randomly selected edges are deleted from the graph, followed by introducing edges between 5% of the randomly chosen pairs of nodes. Note that in this case no new nodes are introduced into the social network.

Second we introduce spurious nodes into the social network, that is, users that did not participate in the mobility trace. Note that for both the Smallblue and Infocom06 dataset we have the social network over a superset of users that participated in the mobility trace; hence, we randomly add nodes from the social network (including respective edges to nodes that were already part of the social network). For example, when *Spurious Nodes* : *No of Nodes* equals 0.1 then 10% of randomly chosen nodes is added to the social network as follows: let V_S denote the set of vertices in the current social network; let $ngh(V_S)$ denote the set of neighbors of vertices in V_S; we pick one vertex at random from the set $ngh(V_S) \setminus V_S$ and add it to the social network; the process is repeated until the required number of nodes are added to the social network. In this case, the size of the resulting social network has 10% more nodes than the contact graph.

Third we randomly remove nodes (and all incident edges) from the social network. For example, when *Missing Nodes* : *No of Nodes* equals 0.1 then 10% of randomly chosen nodes (and respective edges) is removed from the social network. In this case, the resulting social network is 10% fewer nodes than the contact graph.

In all of Figure 16 we used our *recursive sub-graph matching* algorithm since it performed the best. The x-axis in the figures quantifies the extent of obfuscation and the y-axis shows the accuracy of our algorithm. We note that in general the efficacy of our algorithm degrades gracefully with the extent of obfuscation. Also we note that adding/deleting edges and deleting nodes has lesser impact on the efficacy of de-anonymization; on the other hand, adding spurious nodes into the social network is more effective in reducing the efficacy of our algorithm. This is because adding spurious nodes significantly increases the number of possible node mappings, thereby, significantly impacting the efficacy of our algorithms.

4.2 Noisy Contact Graph

Figure 17 shows the accuracy of node de-anonymization when the mobility trace is obfuscated. In this experiment we use the location information available from the St Andrews trace. We note that in both the Smallblue and Infocom06 dataset we only have the contacts. We examine the efficacy of our recursive sub-graph matching algorithm as we add more noise to the mobility trace – in our experiments we add IID (independent and identical distributed) noise to the location of each user each point in time. As shown in

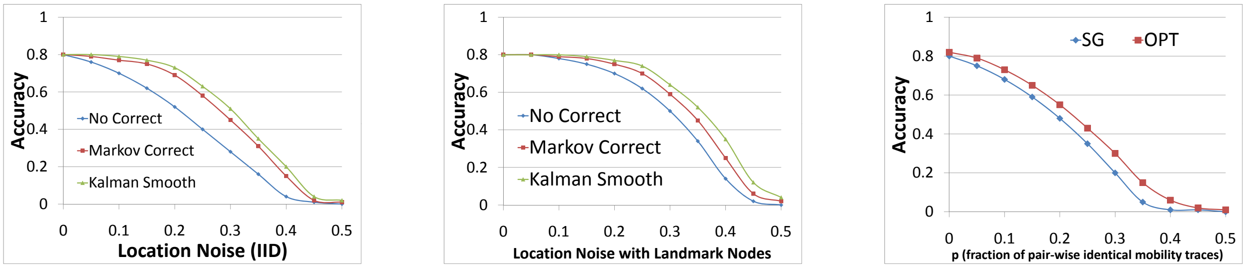

Figure 17: Accuracy with Obfuscated Mobility Traces

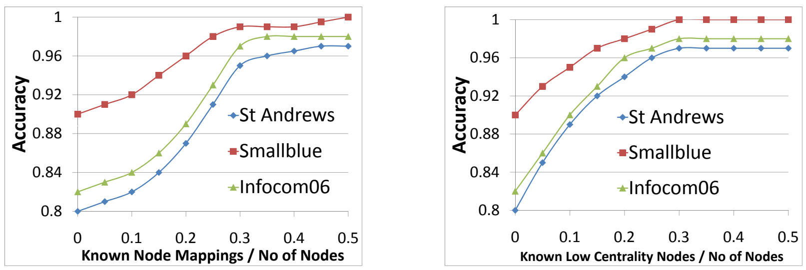

Figure 18: Accuracy with Known Node Mappings

past work, one can use physical limitations on the speed at which a user may move and aggregate *mobility models* to de-obfuscate the mobility trace. In particular, we use the mobility trace to construct a Markovian mobility model built over all users: $Pr(l_{t+1}|l_t)$ (the probability that a user may be location l_{t+1} at time $t+1$ given that the user was at location l_t at time t.

We use the Markovian mobility model (built over all users) to enrich per-user mobility traces. In particular, we used two techniques to refine the obfuscated mobility trace: (i) using Viterbi decoding [30] to deduce the location of user u at time t (this algorithm is quadratic in the number of states = number of location identifiers) and (ii) using Kalman filtering [30] for both backward and forward smoothing to deduce the location of user u at time t (this algorithm is cubic in the number of states = number of location identifiers). We note that these both Viterbi decoding and Kalman smoothing is applied as a pre-processing to enrich the mobility trace $-$ the enriched mobility trace is then used as input by the recursive sub-graph matching algorithm.

Figure 17 shows the effectiveness of adding location noise to: (i) all nodes in the mobility trace and (ii) to only landmark nodes in the mobility trace. We note that using more sophisticated smoothing and state estimation algorithms (e.g., Kalman filter) enhances the efficacy of de-anonymization (at a added computational cost). We also note that adding noise to the mobility traces of landmark nodes is not quite effective; this is because we detect landmark nodes are *outliers* with respect to node centrality score. In general the centrality score of landmark nodes are significantly higher than the remaining nodes; hence, even with added location noise the centrality score of landmark nodes tend be outliers. For example, consider a star topology contact graph, wherein all nodes $\{v_1$, v_2, $\cdots$, $v_n\}$ are in contact with node v_0 and there is no contact between two nodes v_i and v_j such that $1 \leq i < j \leq n$. Even with added location noise we see that it is easy to identify node v_0.

Figure 17 also shows the efficacy of our algorithm when $p\%$ of user mobility traces are identical, that is, we randomly select $p\%$ nodes and for each selected node v we select a random node v' and set the mobility of trace of v to be identical to that of v'. Doing so makes the nodes v and v' automorphic in the contact graph, thereby, decreasing the effectiveness of de-anonymization. We note that this approach is more effective in lowering the efficacy of our algorithm than adding IID location noise to user mobility traces. Note that in this approach to location obfuscation the use of Viterbi decoding or Kalman filter does not enhance the effectiveness of de-anonymization.

4.3 Known Node Mappings

In certain cases one may a priori know the mappings between a small subset of nodes in the social network and the contact graph. This could be due to information that is inadvertently leaked or due to insiders (who participated in the mobility trace) that reveal their mapping information. Figure 18 shows the increase in effectiveness of de-anonymization when more node mappings are revealed. We simply add known node mappings in addition to high centrality node mappings as a seed mapping and uses recursive sub-graph matching algorithm to derive the node mappings for other nodes in the contact graph. For example, when *Known Node Mappings : No of Nodes* is 0.1 then 10% of node mappings are assumed to be known a priori. We observe that given about 30% random node mappings the accuracy of de-anonymization exceeds 95%.

We remark that knowing the node mappings for high centrality nodes is not quite useful for de-anonymization since our algorithm can detect such mappings with high fidelity. On the other hand, the figure shows the effectiveness of de-anonymization when low centrality nodes are revealed (these are nodes that are hard to de-anonymize).

In the next experiment we sort nodes in ascending order of their centrality score. In Figure 18, when *Known Low Centrality Nodes : No of Nodes* equals 0.1 then the mappings of 10% of nodes that

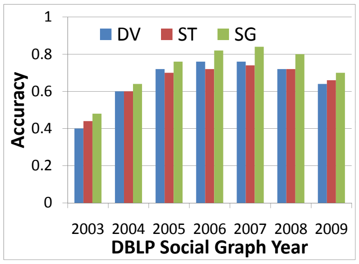

Figure 19: Which Social Network: effectiveness of de-anonymization using DBLP co-authorship graphs from various years

have the least centrality score are assumed to be known a priori. We observe that given 10% random node mappings in the Infocom06 dataset the de-anonymization algorithm achieves an accuracy of 83%, while given the same number of node mappings for low centrality nodes achieves an accuracy of 90% (see Figure 18).

4.4 Which Social Network to Use?

Figure 19 shows the effectiveness of de-anonymization when we use different social networks. In particular, for the Infocom06 dataset we constructed social networks (from DBLP publication database) based on co-authorship information from year 2003, 2004, $\cdots$, 2009 (note that Infocom06 dataset is a mobility trace collected from the Infocom conference conducted in 2006). Figure 19 shows that the effectiveness of de-anonymization initially increases as we approach 2006-07 and then decreases. However, we observed that the de-anonymization is most effective using the 2007 co-authorship social network. We conjecture that this is because authors who met at the 2006 conference collaborated and co-authored papers that were subsequently published in 2007. We could in part verify this conjecture by measuring the number of co-authored papers in 2006 and 2007 respectively, by authors who met frequently during the 2006 conference $-$ indeed our dataset shows a 12% increase in the number of such co-authored papers. Hence, the effectiveness of 2007 social network (over the 2006 social network) can be explained by its stronger causal relationship with the contact graph in 2006. We remark that this observation serves as a measure to quantify the effectiveness of conferences in increasing collaborative research amongst authors.

5. RELATED WORK

Location privacy (as applied to mobile users) requires that it be hard-to-track the location of a user given a mobility trace. In particular several authors have examined the predictability and uniqueness of user location traces using diverse tools; for a survey on past work we refer the readers to [19]. For example, 802.11 user fingerprinting [25] attempts to identify a user using *implicit identifiers* such as IP addresses or the service set identifier (SSID) being actively searched by a user's device; others have suggested RF (Radio Frontend) fingerprinting [5] to uniquely identifier a user's device; others have proposed the use of triangulation (e.g., based on received signal strength from multiple vantage points) to further improve the precision of a user's location [17]; others have suggested that a user be fingerprinted using a historical set of locations visited by that user (e.g., a per-user Markovian mobility model as discussed earlier in Section 1); [20] proposes a framework for recognizing mobile users' activities based on the places they visit and also the temporal patterns of their visit.

More recently, in Oakland 2011, [29] proposed a solution to quantify location privacy with the goal of providing a unified framework that can be used to evaluate various location obfuscation mechanisms. Their paper presents solutions to quantify location privacy given an obfuscated mobility trace and the Markovian mobility model. They show that the effectiveness of an anonymized mobility trace in protecting location privacy not only depends upon the extent of obfuscation but also the fidelity of the auxiliary information. However, their location privacy metrics only applies to auxiliary information of the type per-user Markovian mobility model.

Several authors have explored the use of auxiliary data (sometimes referred to as *side channel* information) to break privacy. A generic template for violating privacy using auxiliary information is often represented as follows [23]: *anonymized data + auxiliary information = de-anonymized data*. Side channels in the form of timing analysis [10] and power analysis [18] have been extensively studied in literature. More recently, authors have used side-channel information (e.g., zipcode, age and sex of users) to de-anonymize the Netflix Prize dataset [23].

The closest related work is [**?**] that proposes using graph deanonymization for social networks. Their approach in principle is similar to our recursive sub-graph matching approach. However, this paper presents various key solutions over prior work: (i) using node centrality to identify landmark nodes and bootstrap deanonymization, (ii) reductions to weighted graph matching and tree edit distance problem, (iii) dynamic CSP formulation can be viewed as a alternate formulation of the approach presented in [**?**], and (iv) lower bounds based on automorphism. Finally, to the best of our knowledge, this paper presents the first attempt to leverage social networks as a side-channel to de-anonymize user mobility traces.

6. SUMMARY

This paper we explored an alternate source of auxiliary information $-$ inter-user correlations which can be often inferred from publicly available social networks to de-anonymize mobility traces. A vast majority of past work developed detailed *per-user models* (e.g., a per-user Markovian mobility model) and used such models to identify the most probable user that could have generated an anonymized trace. In contrast, this paper studied the use of *inter-user correlation models* to address this problem. In particular, we exploited structural similarities between two sources of inter-user correlations (the contact graph and the social network) and developed techniques to leverage such structural similarities to deduce mapping between nodes in the contact graph with that in the social network, thereby de-anonymizing the contact graph (and thus the underlying mobility trace). We validated our hypothesis using three real world datasets and showed that the proposed approach achieves over 80% accuracy, while incurring no more than a few minutes of computational cost in de-anonymizing these mobility traces.

7. REFERENCES

[1] Please rob me.

[2] Smallblue. http://domino.research.ibm.com/comm/research_projects.nsf/pages/smallblue.index.html.

[3] A. Biem, E. Bouillet, H. Feng, A. Ranganathan, A. Riabov, O. Verscheure, H. Koutsopoulos, and C. Moran. Ibm infosphere streams for scalable, real-time intelligent transportation services. In *Proceedings of the 2010 international conference on Management of data*, SIGMOD '10, pages 1093–1104, 2010.

[4] G. Bigwood, D. Rehunathan, M. Bateman, T. Henderson, and S. Bhatti. CRAWDAD data set st_andrews/sassy (v. 2011-06-03). Downloaded from http://crawdad.cs.dartmouth.edu/st_andrews/sassy, June 2011.

[5] A. Cohen. Rf fingerprinting pinpoints location. http://www.networkworld.com/news/tech/2004/101104techupdate.html.

[6] W. J. Cook, W. H. Cunningham, W. R. Pulleybank, and A. Schrijver. *Combinatorial Optimization*. Wiley-Interscience, 1997.

[7] P. Crucitti, V. Latora, and S. Porta. Centrality measures in spatial networks of urban streets. In *Physical Review E: Statistical, Non-Linear and Soft Matter Physics*, 2006.

[8] P. T. Darga, H. Katebi, M. Liffiton, I. Markov, and K. Sakallah. Saucy2: Fast symmetry discovery. http://vlsicad.eecs.umich.edu/BK/SAUCY/, 2008.

[9] DBLP. DBLP Bibiography. http://www.informatik.uni-trier.de/ ley/db/.

[10] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second generation onion router. In *13th USENIX Security Symposium*, 2000.

[11] R. K. Ganti, N. Pham, H. Ahmadi, S. Nangia, and T. Abdelzaher. Greengps: A participatory sensing fuel-efficient maps application. In *In Proc. of Conference on Mobile Systems, Applications, and Services, MobiSys*, pages 151–164, 2010.

[12] W. Gao, A. Iyengar, M. Srivatsa, and G. Cao. Supporting cooperative caching in disruption tolerant networks. In *IEEE Intl Conference on Distributed Computing Systems (ICDCS)*, 2011.

[13] M. Gonzalez, C. Hidalgo, and A.-L. Barbasi. Understanding individual human mobility patterns. *Nature*, 453:779–782, 2008.

[14] J. M. Harris, J. L. Hirst, and M. J. Mossinghoff. *Combinatorics and Graph Theory*. Springer Mathemathics, 2008.

[15] B. Hull et al. Cartel: a distributed mobile sensor computing system. In *Proc. of SenSys*, pages 125–138, 2006.

[16] IBM ILOG Solver. Constraint programming. http://www-01.ibm.com/software/integration/optimization/cplex-cp-optimizer/.

[17] T. Jiang, H. J. Wang, and Y.-C. Hu. Preserving location privacy in wireless lans. In *MobiSys*, 2005.

[18] P. Kocher, J. Jaffe, and B. Jun. Differential power analysis. In *Advances in Cryptology (Crypto)*, 1999.

[19] J. Krumm. A survey of computational location privacy. In *Personal Ubiquitous Computing, 13(6): 391-399*, 2009.

[20] L. Liao, D. J. Patterson, D. Fox, and H. Kautz. Learning and inferring transportation routines. In *Artificial Intelligence 171:311Ű331*, 2007.

[21] H. Lu, W. Pan, N. Lane, T. Choudhry, and A. Campbell. Soundsense: Scalable sound sensing for people-centric applications on mobile phones. In *In Proc. of ACM MobiSys*, pages 165–178, 2009.

[22] I. Miguel. *Dynamic Flexible Constraint Satisfaction and its Application to AI Planning*. Springer, 2003.

[23] A. Narayanan and V. Shmatikov. De-anonymizing social networks. In *IEEE Symposium on Security and Privacy*, 2009.

[24] T. Opsahl, F. Agneessens, and J. Skvoretz. Node centrality in weighted networks: Generalizing degree and shortest paths. In *Social Networks*, 2010.

[25] J. Pang, B. Greenstein, R. Gummadi, S. Seshan, and D. Wetherall. 802.11 user fingerprinting. In *ACM MobiCom*, 2007.

[26] S. M. Ross. *Introduction to probability models*. Academic Press, 2006.

[27] J. Scott, R. Gass, J. Crowcroft, P. Hui, C. Diot, and A. Chaintreau. CRAWDAD data set cambridge/haggle (v. 2009-05-29). Downloaded from http://crawdad.cs.dartmouth.edu/cambridge/haggle, May 2009.

[28] S. M. Selkow. The tree-to-tree editing problem. In *Inform. Process Lett., 6(6):184Ű186*, 1977.

[29] R. Shokri, G. Theodorakopoulos, J.-Y. Boudec, and J.-P. Hubaux. Quantifying location privacy. In *IEEE Symposium on Security and Privacy*, 2011.

[30] R. F. Stengel. *Optimal Control and Estimation*. Dover Publications, 1994.

[31] A. Thiagarajan, J. Biagioni, T. Gerlich, and J. Eriksson. Cooperative transit tracking using smart-phones. In *In Proc. of Conference on Embedded Networked Systems, SenSys*, pages 85–98, 2010.