SecGraph: A Uniform and Open-source Evaluation System for Graph Data Anonymization and De-anonymization

Shouling Ji
Georgia Institute of Technology

Weiqing Li
Georgia Institute of Technology

Prateek Mittal
Princeton University

Xin Hu
IBM Thomas J. Watson Research Center

Raheem Beyah
Georgia Institute of Technology

Abstract

In this paper, we analyze and systematize the state-of-the-art graph data privacy and utility techniques. Specifically, we propose and develop *SecGraph* (available at [1]), a uniform and open-source Secure Graph data sharing/publishing system. In SecGraph, we systematically study, implement, and evaluate 11 graph data anonymization algorithms, 19 data utility metrics, and 15 modern Structure-based De-Anonymization (SDA) attacks. To the best of our knowledge, SecGraph is the first such system that enables data owners to anonymize data by state-of-the-art anonymization techniques, measure the data's utility, and evaluate the data's vulnerability against modern De-Anonymization (DA) attacks. In addition, SecGraph enables researchers to conduct fair analysis and evaluation of existing and newly developed anonymization/DA techniques. Leveraging SecGraph, we conduct extensive experiments to systematically evaluate the existing graph data anonymization and DA techniques. The results demonstrate that (*i*) most anonymization schemes can partially or conditionally preserve most graph utilities while losing some application utility; (*ii*) no DA attack is optimum in all scenarios. The DA performance depends on several factors, e.g., similarity between anonymized and auxiliary data, graph density, and DA heuristics; and (*iii*) all the state-of-the-art anonymization schemes are vulnerable to several or all of the modern SDA attacks. The degree of vulnerability of each anonymization scheme depends on how much and which data utility it preserves.

1 Introduction

Many computing systems generate data with graph structure, e.g., social networks, collaboration networks, and email networks [2–4]. Even mobility traces, e.g., WiFi traces, Bluetooth traces, instant message traces, and check-ins, can be modeled by graphs via applying sophisticated techniques [3–5]. Generally, those data are called *graph data*. For research purposes, data and network mining tasks, and commercial applications, these graph data are often transferred, shared, and/or provided to the public, research community, and/or commercial partners. Since graph data carry a lot of sensitive private information of users/systems who generated them [2, 3], it is critical to protect users' privacy during the data transferring, sharing, and/or publishing.

To protect users' privacy, several anonymization techniques have been proposed to anonymize graph data, which can be classified into six categorizes: Naive ID Removal, Edge Editing (EE) based techniques [6], k-anonymity based techniques [7–11], Aggregation/Class/Cluster based techniques [12–14], Differential Privacy (DP) based techniques [15–19], and Random Walk (RW) based techniques [20]. Fundamentally, these techniques try to protect users' privacy by perturbing the original graph's structure while preserving as much data utility as possible.

Following Narayanan and Shmatikov's work [2], many new Structure-based De-Anonymization (SDA, we use DA and SDA interchangeably in this paper) attacks on graph data have been proposed, which can be categorized into two classes: *seed-based attacks*, e.g., Narayanan-Shmatikov's attack [2], and *seed-free attacks*, e.g., Ji et al.'s attack [3]. For both types of attacks, the goal is to de-anonymize anonymized users using their uniquely distinguishable structural characteristics.

Surprisingly, although we already have many sophisticated anonymization techniques (e.g., [6, 7, 12, 15, 20]) and powerful SDA attacks (e.g., [2,3,5,21–24]), whether state-of-the-art anonymization techniques can defend against modern SDA attacks is still an open problem. This is because of the incomplete evaluation of existing anonymization and DA techniques. For anonymization works, they usually only evaluate the data utility performance of their proposed techniques (although some works provide a theoretical security guarantee, these guarantees usually do not hold due to improper

assumptions or incomplete considerations as analyzed in Section 4). For DA works, they usually evaluate their attacks' performance without applying state-of-the-art anonymization techniques (e.g., k-anonymity based schemes, DP based schemes) to their test data.

Contributions. To address the above open problem, we systematically study, implement, and evaluate existing graph data anonymization techniques and DA attacks. Specifically, our main contributions are as follows.

(a) We design and implement a Secure Graph data publishing/sharing (SecGraph) system (available at [1]). SecGraph enables data owners to anonymize their data using state-of-the-art anonymization techniques, measure the anonymized data's graph and application utilities, and comprehensively evaluate their data's actual vulnerability against modern DA attacks. To the best of our knowledge, SecGraph is the first such system publicly available to both academia and industry. More importantly, SecGraph provides the first *uniform platform* that enables researchers to conduct accurate comparative studies of anonymization/DA techniques, and to comprehensively understand the resistance/vulnerability of existing or newly developed anonymization techniques, the effectiveness of existing or newly developed DA attacks, and graph and application utilities of anonymized data.

(b) In SecGraph, we systematically analyze, implement, and evaluate 11 state-of-the-art graph data anonymization schemes and 19 graph and application utility metrics. We also analyze the 11 anonymization schemes with respect to the 19 utility metrics, both analytically and experimentally. The evaluation results demonstrate that most existing anonymization algorithms can partially or conditionally preserve most graph utilities. However, all the anonymization schemes lose one or more application utility.

(c) We summarize and analyze the fundamental properties of existing SDA attacks. Then, we systematically implement and evaluate 15 modern SDA attacks on real-world graph datasets. Our results show that modern SDA attacks are powerful and robust to seed mapping errors. Furthermore, no attack is optimum in all scenarios. The DA performance of an attack depends on the similarity between the anonymized and auxiliary data, graph density, DA heuristics, etc.

(d) We analytically and experimentally evaluate the performance of existing graph data anonymization schemes on defending against modern SDA attacks. We find that existing anonymization techniques are vulnerable to modern SDA attacks. Their degree of vulnerability depends on how much data utility is preserved in the anonymized data.

Abbreviations. For convenient reference, we summarize the used abbreviations in Table 1.

Roadmap. In Section 2, we study existing graph data anonymization schemes and their utility performance. In Section 3, we study modern SDA attacks. In Section 4, the effectiveness of existing anonymization schemes against modern DA attacks is analyzed. We systematically implement and evaluate SecGraph in Section 5. The future research directions are discussed in Section 6. We conclude this paper in Section 7.

Table 1: Abbreviations and acronyms.

Terms	SDA	Structure-based De-anonymization
	DA	De-anonymization
	SF	Seed-Free
Anonymizatoin	EE	Edge Editing
	DP	Differential Privacy
	RW	Random Walk
	k-NA	k-Neighborhood Anonymity
	k-DA	k-Degree Anonymity
	k-auto	k-automorphism
	k-iso	k-isomorphism
Utility metrics	Deg.	Degree
	JD	Joint Degree
	ED	Effective Diameter
	PL	Path Length
	LCC	Local Clustering Coefficient
	GCC	Global Clustering Coefficient
	CC	Closeness Centrality
	BC	Betweenness Centrality
	EV	Eigenvector
	NC	Network Constraint
	NR	Network Resilience
	Infe.	Infectiousness
	RX	Role extraction
	RE	Reliable Email
	IM	Influence Maximization
	MINS	Minimum-sized Influential Node Set
	CD	Community Detection
	SR	Secure Routing
	SD	Sybil Detection
De-anonymization	DV	Distance Vector [5]
	RST	Randomized Spanning Tress [5]
	RSM	Recursive Subgraph Matching [5]
	DeA	De-Anonymization [25]
	ADA	Adaptive De-Anonymization [25]
	BDK	Backstrom et al.'s attacks [26]
	NS	Narayanan-Shmatikov's attack [2]
	NSR	Narayanan et al.'s attack [21]
	NKA	Nilizadeh et al.'s attack [22]
	PFG	Pedarsani et al.'s attack [23]
	YG	Yartseva-Grossglauser's attack [27]
	KL	Korula-Lattanzi's attack [24]
	JLSB	Ji et al.'s attack [3]

2 Graph Anonymization

2.1 Status Quo

Generally, existing graph data anonymization techniques can be classified into six categories. We discuss each category as follows.

Naive ID Removal. To anonymize graph data, a straightforward method is *naive ID removal*. Although this method has been demonstrated to be extremely vulnerable to SDA attacks, it is still widely used because of its simplicity, ease of applicability, and scalability [2, 3, 5, 26, 28].

Edge Editing based Anonymization. To protect graph data's privacy, Ying and Wu proposed spectrum preserved Edge Editing (EE) based schemes *Add/Del* and *Switch* [6]. Under *Add/Del*, k randomly chosen edges will be added followed by the deletion of another k randomly chosen edges. Under *Switch*, k random *edge switches* are conducted.

***k*-anonymity.** k-anonymity has been widely used to anonymize relational data [29, 30]. Similarly, much effort has been spent to extend k-anonymity to graph data [7–11]. To defend against neighborhood attacks, Zhou and Pei proposed *k-Neighborhood Anonymity* (k-NA) for graph data [7]. In another work, Liu and Terzi considered *degree attacks* and proposed *k-Degree Anonymity* (k-DA) for graph data, under which for each user, there exists at least $k-1$ other users with the same degree [8]. In [9], Zou et al. simultaneously considered four types of structural attacks on graph data and proposed *k-automorphism* (k-auto), where each user always has $k-1$ other symmetric users with respect to $k-1$ automorphic functions. Another similar work is [10], where Cheng et al. proposed *k-isomorphism* (k-iso) to defend against structural attacks. Under k-iso, a graph is partitioned and anonymized into k disjoint isomorphic subgraphs. In [11], Yuan et al. considered personalized privacy protection for anonymizing graph data in terms of both semantic and structural information.

Aggregation/Class/Cluster based Anonymization. Another popular idea to protect graph data is to group users into *clusters* (equivalently, *groups, classes*). In [12], Hay et al. proposed an *aggregation based graph anonymization* algorithm, which first partitions users and then describes the graph at the level of partitions. Another work, at the semantics level, is [13], where Bhagat et al. designed a class-based anonymization algorithm. In [14], Thompson and Yao presented two *cluster-based anonymization* schemes for graph data.

Differential Privacy. Differential Privacy (DP) is an emerging anonymization technique with a strong privacy guarantee [31, 32]. Initially, DP was proposed for statistical databases [31]. Recently, there have been works that seek to enable differentially private graph data release. Aiming at protecting *edge/link privacy*, defined as the privacy of users' relationship in graph data, in [15], Sala et al. introduced *Pygmalion*, a differentially-private graph model. To bypass many difficulties encountered when working with the worst-case sensitivity [15], Proserpio recently presented a general platform, named *wPINQ*, for differentially private data analysis and publishing [16, 17]. Similar to [15], Wang and Wu also employed the dK-graph generation model for enforcing edge DP in graph anonymization [18]. Another recent work for edge DP is [19], where Xiao et al. proposed a Hierarchical Random Graph (HRG) model based scheme to meet edge DP.

Random Walk based Anonymization. In [20], Mittal et al. proposed a *Random Walk (RW) based anonymization* technique for preserving link (edge) privacy. In this technique, an edge in the original graph is replaced by a RW path.

2.2 Anonymization and Utility

Generally, an anonymization scheme can be evaluated from two perspectives: *data utility preservation* and *resistance to DA attacks*. However, most, if not all, existing graph anonymization works have not been significantly evaluated with respect to their utility or resistance to DA attacks. On one hand, most existing graph anonymization works only conducted limited evaluations on their utility preservation, e.g., degree distribution, path length distribution, which are insufficient to understand their value for high-level data mining tasks and applications, e.g., sense-making, search for similar users, user classification, reliable email, influence maximization. On the other hand and more seriously, to the best of our knowledge, no work (including existing DA works) actually evaluated the resistance of state-of-the-art anonymization techniques against modern SDA attacks.

To address these concerns, we comprehensively analyze the utility of existing graph data anonymization algorithms in this subsection and defer the detailed resistance analysis to Section 4. Before performing the analysis, we first present the used utility metrics, which can be classified as *graph utility metrics* or *application utility metrics*.

2.2.1 Graph Utility Metrics

Graph utility captures how the anonymized data preserves fundamental structural properties of the original graph after applying an anonymization technique. Particularly, we examine 12 graph utility metrics of existing anonymization schemes as follows[1].

Degree (Deg.), which refers to the degree distribution; *Joint Degree (JD)*, which refers to the joint degree distribution of a graph; *Effective Diameter (ED)*, which is defined as the minimum number of hops in which 90% of all connected pairs of nodes can reach each other; *Path Length (PL)*, which refers to the distribution of the

[1]Without of causing confusion, we interchangeably use node and user in this paper.

shortest path lengths between all pairs of users; *Local Clustering Coefficient (LCC) and Global Clustering Coefficient (GCC). Clustering coefficient* measures the degree to which users in graph data tend to cluster together. *Closeness Centrality (CC)*, which is defined as the *inverse of the farness* of a user within a graph and measures how long it takes to spread information from a user to all other users sequentially; *Betweenness Centrality (BC)*, which quantifies the number of times a user acts as a bridge along the shortest path between two other users; *EigenVector (EV).* The EV of the adjacency matrix A of a graph G is a non-zero vector $\mathbf{v}$ such that $A\mathbf{v} = \lambda\mathbf{v}$, where λ is some scalar multiplier; *Network Constraint (NC)*, which measures the extent to which a user links to others that are already linked to each other; *Network Resilience (NR)* [33], which measures how robust a graph is and is defined as the number of users in the *largest connected component* when users are removed from the graph in the degree decreasing order; and *Infectiousness (Infe.)* [34], which measures the number of users infected by a disease, given that a randomly chosen user is infected and each infected user transmits this disease to its neighbors with some infection rate.

2.2.2 Application Utility Metrics

In reality, most data is published/shared for data/network mining tasks, high-level applications, etc. Therefore, besides examining data's fundamental structural utility, it is also crucial to ensure that the anonymized data is useful for practical applications. Toward this objective, we evaluate 7 popular application utility metrics for anonymization schemes as follows.

(a) *Role eXtraction (RX) [35].* Based on users' structural behavior, users in a graph can be labeled as having different roles, e.g., *clique members*, *periphery-nodes*. RX is an important operation for graph data that is useful for many network mining tasks such as sense-making. We measure the RX utility of an anonymization scheme using the method in [35].

(b) *Reliable Email (RE) [36].* RE is a whitelisting system leveraging users' neighborhoods to filter and block spam emails. To evaluate the structural utility of an anonymization scheme with respect to RE, we take a similar method as in [15] to compute the number of users who can be spammed by a fixed number of compromised neighbors in a graph.

(c) *Influence Maximization (IM) [37].* The IM problem seeks to find a set of θ users such that these θ users have the maximum influence to the network under some influence propagation model. IM is important for many real world applications, e.g., advertisements.

(d) *Minimum-sized Influential Node Set (MINS) [38].* MINS is another popular and important application utility metric that leverages a graph's structure to identify the minimum-sized set of influential nodes, such that all other nodes in the network could be influenced with a probability above a threshold. MINS can be used in many meaningful applications, e.g., social problems alleviation, new products promotion.

(e) *Community Detection (CD) [39].* CD is a popular application on graph data which enables comprehensive analysis of a network structure and supports other applications, e.g., classification, routing (information propagation). To measure the CD utility of an anonymization scheme, we employ the hierarchical agglomeration algorithm proposed in [39].

(f) *Secure Routing (SR) [40].* The structure of graph data can also be used to improve the performance of secure routing for systems such as P2P systems. For our purpose, we evaluate the SR application utility of an anonymization scheme using the method designed in [40].

(g) *Sybil Detection (SD) [41].* Sybil attacks are a serious threat to both centralized and distributed systems, e.g., recommendation systems, anonymity systems. For our purpose, we evaluate the SD application utility of an anonymization scheme using the method in [41].

2.2.3 Anonymization vs Utility

We are ready to analyze the utility performance of existing graph data anonymization techniques. We summarize the graph and application utilities, and Resistance to SDA attacks (R2SDA) (e.g., [2, 3, 25, 27]) of existing graph anonymization schemes in Table 2. We analyze the results in Table 2 as follows.

For the Naive ID removal scheme, it is straightforward that it preserves all the data utility. However, it is also the most vulnerable scheme to SDA attacks.

Since *Add/Del* randomly adds and deletes edges, which is an global edge edition operation and thus it may change many fundamental structural properties of a graph. It follows that it can conditionally or partially preserve both graph and application utilities. However, utilities like JD, GCC, NC, CD, and MINS would be destroyed if too many existing edges are deleted while new edges are added. For *Switch*, it switches two randomly selected qualified edges, which preserves the degree of each user. Consequently, *Switch* can preserve Deg. and partially preserve most other utilities. Furthermore, compared to *Add/Del*, *Switch* can conditionally preserve the RX and CD utilities which are destroyed in *Add/Del*. This is because that *Add/Del* randomly changes users' degree in the global edge edition process and thus some global structure-sensitive application utility is lost or significantly affected. Furthermore, *Add/Del* and *Switch* cannot defend against modern SDA attacks as shown in [2, 3, 5].

Table 2: Analysis of existing graph anonymization techniques. ✓ = preserving the utility, ◐= partially preserving the utility, ♦ = conditionally preserving the utility depending on parameters and considered data (based on our analysis, it is necessary to distinguish *partially* and *conditionally* preserving a data utility. For instance, k-DA conditionally preserves the Deg. utility depending on k while *Add/Del* can partially preserve the Deg. utility for an arbitrary k), ✗ = not preserving the utility, and **n/a** = evaluation not available in existing works.

	graph utility												application utility							R2SDA
	Deg.	JD	ED	PL	LCC	GCC	CC	BC	EV	NC	NR	Infe.	RX	RE	IM	MINS	CD	SR	SD	
Naive	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗
Add/Del [6]	◐	♦	◐	◐	♦	♦	◐	◐	◐	♦	◐	◐	✗	◐	◐	♦	✗	♦	♦	✗
Switch [6]	✓	♦	♦	◐	♦	♦	◐	◐	◐	◐	♦	◐	♦	◐	◐	♦	♦	◐	◐	✗
k-NA [7]	♦	♦	♦	♦	♦	♦	◐	♦	♦	◐	◐	◐	✗	◐	♦	♦	♦	♦	♦	**n/a**
k-DA [8]	♦	♦	♦	♦	♦	♦	◐	♦	♦	◐	◐	◐	✗	◐	♦	♦	♦	♦	♦	**n/a**
k-auto [9]	♦	♦	♦	♦	♦	♦	◐	♦	♦	◐	◐	◐	✗	◐	♦	♦	♦	♦	♦	**n/a**
k-iso [10]	♦	♦	✗	✗	♦	✗	✗	✗	✗	♦	✗	✗	✗	✗	✗	♦	♦	✗	♦	**n/a**
Aggregation [12]	♦	♦	♦	♦	♦	♦	◐	♦	♦	◐	◐	◐	✗	◐	♦	♦	♦	♦	♦	**n/a**
Cluster [14]	♦	♦	♦	♦	♦	♦	◐	♦	♦	◐	◐	◐	✗	◐	♦	♦	♦	♦	♦	**n/a**
DP [15]	♦	♦	♦	◐	♦	♦	◐	♦	♦	♦	♦	◐	✗	◐	♦	♦	✗	♦	♦	**n/a**
DP [16, 17]	♦	♦	♦	◐	♦	♦	◐	♦	♦	♦	♦	◐	✗	◐	♦	♦	✗	♦	♦	**n/a**
DP [18]	♦	♦	♦	◐	♦	♦	◐	♦	♦	♦	♦	◐	✗	◐	♦	♦	✗	♦	♦	**n/a**
DP [19]	♦	♦	♦	◐	♦	♦	◐	♦	♦	♦	♦	◐	✗	◐	♦	♦	✗	♦	♦	**n/a**
RW [20]	✓	♦	♦	♦	♦	✗	◐	♦	♦	◐	♦	◐	✗	◐	♦	✗	✗	◐	◐	**n/a**

The k-anonymity based anonymization schemes k-NA [7], k-DA [8], and k-auto [9] can partially/conditionally preserve the graph and most application utilities except for the RX utility. This is because the fundamental idea of k-anonymity based schemes is to make k users/subgraphs structurally similar. Therefore, there is a tradeoff between anonymity and utility. If k is large, more users will be structurally similar while more utility will be lost. On the other hand, if k is chosen to be small, more utility will be preserved at the cost of lower anonymity guarantee. Furthermore, since every user is guaranteed to be structurally similar to at least $k-1$ other users while the RX utility tries to distinguish users based on their structural differences, it turns out k-anonymity based schemes cannot preserve the RX utility. As we discussed before, k-iso achieves structure anonymization by partitioning the original graph into k isomorphic subgraphs. Therefore, several fundamental properties of a graph will be destroyed, e.g., connectivity. It follows that several important graph and application utilities are lost in k-iso, e.g., PL, GCC, NR, Infe., RX, RE, IM, and SR. Finally, compared with other schemes, k-NA, k-auto, and k-iso have higher computational complexities.

Similar to k-anonymity based schemes, the cluster based schemes [12, 14] can conditionally/partially preserve graph and application utilities except for RX. This is because the fundamental idea of cluster based schemes is to make the users within a cluster structurally indistinguishable. Therefore, to what extent these schemes can preserve data utility depends on the cluster size setting. Again, since RX is achieved based on users' structural difference, this utility is not preserved in cluster based schemes.

For DP based schemes (e.g., [15, 19]), their main objective is to protect link privacy by perturbing the edges of a graph. The fundamental idea of these schemes is to make an anonymized graph structurally similar to its neighboring graphs and thus an adversary cannot infer the existence of an edge. Therefore, they can conditionally/partially preserve most graph and application utilities. However, if a high level of privacy is guaranteed, many edges in the graph are changed. Furthermore, similar to *Add/Del*, the edge perturbation in DP also belongs to global edge edition. Therefore, the global structure-sensitive high-level application utilities, e.g., RX, MINS, and CD, are destroyed or significantly reduced in DP based schemes.

In the RW based scheme [20], link privacy is achieved by replacing a random walk path with an edge, and thus this scheme, theoretically, will not change the degree distribution of the original data. It follows that several utilities, e.g., Deg., RX, SD, NR, Infe., can be preserved or partially preserved. However, some other global utilities, e.g. JD, GCC, are lost in the RW based scheme due to the significant change of the overall graph structure.

From Table 2, no existing work evaluates the resistance of state-of-the-art anonymization schemes against modern SDA attacks. Although most of the schemes have nice theoretical privacy guarantees, unfortunately, that privacy analysis cannot guarantee that they can defend against modern SDA attacks due to the improper model of the adversary's auxiliary information, problematic assumptions, etc. Therefore, aiming to address this open problem, we evaluate the effectiveness of existing graph data anonymization schemes against modern SDA attacks in Sections 4 and 5.

3 Graph De-anonymization

3.1 Graph Data DA

3.1.1 Seed-based DA

When de-anonymizing graph data, it is intuitive to identify some users first as seeds. Then, the large scale DA is bootstrapped from these seeds. In [26], Backstrom et al. presented both active attacks and passive attacks to graph data. However, the attacks in [26] have several limitations, e.g., they are not scalable and they leverage sybil users that can be detected by modern sybil defense techniques [41]. To improve the attacks in [26], Narayanan and Shmatikov presented a scalable two-phase DA attack to social networks [2]. In the first phase, some seed users are identified between the anonymized graph and the auxiliary graph. In the second phase, starting from the identified seeds, a self-reinforcing DA propagation process is iteratively conducted based on both graphs' structural characteristics, e.g., node degrees, nodes' eccentricity, edge directionality. Later, Narayanan et al. employed a simplified version of the attack in [2] (using less DA heuristics) for link prediction [21]. In [22], Nilizadeh et al. extended Narayanan and Shmatikov's attack by proposing a community-enhanced DA scheme of social networks. Actually, the community-level DA in [22] can also be applied to enhance other seed-based DA attacks (e.g., [5, 25]).

In [5], Srivatsa and Hicks presented three attacks to de-anonymize mobility traces, which can be modeled as contact graphs by applying multiple preprocessing techniques (e.g., [5]). Similar to Narayanan et al.'s attacks [2, 21], Srivatsa-Hicks' attacks also consist of two phases, where the first phase is for seed identification and the second phase is for mapping (DA) propagation. To achieve mapping propagation, Srivatsa and Hicks proposed three heuristics based on Distance Vector (DV), Randomized Spanning Trees (RST), and Recursive Subgraph Matching (RSM). In [25], Ji et al. proposed two two-phase DA attack frameworks, namely De-Anonymization (DeA) and Adaptive De-Anonymization (ADA), which are workable when the auxiliary data only has partial overlap with the anonymized data.

In [24,27], besides quantifying the de-anonymizability of graph data, the authors also proposed DA attacks. In [27], Yartseva and Grossglauser proposed a simple percolation-based DA algorithm to graph data. Given a seed mapping set, the algorithm incrementally maps every pair of users (from the anonymized and auxiliary graphs respectively) with at least r neighboring mapped pairs, where r is a predefined mapping threshold. Another similar attack was presented by Korula and Lattanzi [24], which also starts from a seed set and iteratively maps a pair of users with the most number of neighboring mapped pairs.

Table 3: Analysis of existing graph DA techniques. SF = seed-free, AGF = auxiliary graph-free, SemF = semantics-free, A/P = active/passive attack, Scal. = scalable, Prac. = practical, Rob. = robust to noise, ✓ = true, ◐= partially true, ⧫ = conditionally true, and ✗ = false.

	SF	AGF	SemF	A/P	Scal.	Prac.	Rob.
BDK [26]	✓	✓	✓	A, P	✗	◐	✗
NS [2]	✗	✗	✓	P	✓	✓	✓
NSR [21]	✗	✗	✓	P	✓	✓	✓
NKA [22]	⧫	✗	✓	P	⧫	⧫	⧫
DV [5]	✗	✗	✓	P	⧫	⧫	✓
RST [5]	✗	✗	✓	P	⧫	⧫	✓
RSM [5]	✗	✗	✓	P	⧫	⧫	✓
PFG [23]	✓	✗	✓	P	✓	⧫	⧫
YG [27]	✗	✗	✓	P	✓	⧫	✓
DeA [25]	✗	✗	✓	P	✓	✓	✓
ADA [25]	✗	✗	✓	P	✓	✓	✓
KL [24]	✗	✗	✓	P	✓	⧫	✓
JLSB [3]	✓	✗	✓	P	✓	✓	✓

3.1.2 Seed-free DA

Taking another approach, some powerful seed-free DA attacks on graph data have been proposed. Using degrees and distances to other nodes as each node's fingerprints, Pedarsani et al. proposed a Bayesian model based seed-free algorithm for graph data DA [23]. Another seed-free DA attack to graph data was presented by Ji et al. [3]. Unlike previous attacks, Ji et al.'s attack is an optimization based single-phase cold start algorithm.

3.2 Graph DA Analysis

In this subsection, we analyze the performance of existing graph data DA algorithms. For convenience, in the rest of this paper, we denote Backstrom et al.'s attacks [26] by BDK (the initials of the authors), Narayanan-Shmatikov's attack [2] by NS, Narayanan et al.'s attack [21] by NSR, Nilizadeh et al.'s attack [22] by NKA, Srivatsa-Hicks' three attacks [5] by DV, RST, and RSM, respectively, Pedarsani et al.'s attack [23] by PFG, Yartseva-Grossglauser's attack [27] by YG, Ji et al.'s two attacks [25] by DeA and ADA, respectively, Korula-Lattanzi's attack [24] by KL, and Ji et al.'s attack [3] by JLSB. We show our analytical results in Table 3 and discuss the result as follows.

Except for BDK, all the existing SDA attacks are passive attacks and require auxiliary graphs to perform the attack, i.e., they employ the structural similarity between the the anonymized graph and the auxiliary graph to break the anonymity. However, when we examine the anonymization schemes in Table 2, we find that none properly consider such auxiliary information in their threat models.

To perform BDK attacks [26], an adversary either has to insert some Sybil users in the dataset before the actual anonymized data release, or has to be an internal user that knows its neighborhoods. In either case, such attacks can only de-anonymize some users but cannot de-anonymize users in large scale. Furthermore, the attacks cannot tolerate any topological change of the original data. Therefore, BDK attacks are not scalable or robust. These attacks require that an adversary successfully launches Sybil users or be an internal user that obtains his neighborhoods.

All the examined DA attacks are semantics-free. This is because the structural information itself is sufficient to perfectly or partially de-anonymize graph users. Furthermore, compared to semantics information, structural information is widely available in large scale, resilient to noise, and easily computable [2, 3, 5]. Following this fact, all the attacks except for BDK are (conditionally) scalable, practical, and robust.

Specifically, DV, RST, and RSM [5] are conditionally scalable and practical. This is because they are not computationally feasible when the number of seeds is large. PFG [23] is conditionally practical and robust. This is because it is very sensitive to the graph density of the anonymized data. Generally, this attack is suitable for sparse graphs however it has a significant performance degradation as the graph density increases. YG [27] is conditionally practical because it is designed to de-anonymize users of degree no less than 4 in the anonymized data. In many real world graph datasets, the users with degree less than 4 could dominate or take a significant portion of graph data based on the statistics in [3]. The conditional practicability of KL [24] comes from its improper assumption that $\Theta(\iota \cdot n)$ ($\iota \in (0, 1]$ is a constant and n is the number of nodes in a graph) seeds are available, which is too strong to hold for real world DA attacks. Note that, the community-level DA of NKA [22] is scalable (with complexity of $O(n^2)$). However, the NKA [22] is conditionally scalable, practical, and robust. This is because, if the community-level DA of NKA [22] is employed to enhance DV, RST, RSM, YG, and/or KL, it is conditionally scalable, practical, and/or robust. NS [2], NSR [21], DeA, ADA, and JLSB [3, 25] adaptively perform DA employing several heuristics based on a graph's local and global structural characteristics. It follows that they are scalable, practical, and robust as long as similarity exists between anonymized graphs and auxiliary graphs.

Both seed-based attacks (e.g., NS, DV) and seed-free attacks (e.g., PFG, JLSB) have advantages depending on the application scenarios. On one hand, seed-based attacks are more stable with respect to de-anonymizing arbitrary anonymized graphs. The reason is straightforward since seed knowledge provides more auxiliary information to an adversary. On the other hand, it is possible that in some scenarios seeds are not available, and thus seed-free attacks are more general. Furthermore, if there is some error in the seed seeking phase (which is possible in real world attacks), seed-based attacks will suffer performance de-gradation or will possibly fail.

Table 4: DA attacks vs anonymization techniques. Naive = naive ID removal, EE = EE based schemes [6], k-anony. = k-anonymity based schemes [7]- [10], Cluster = cluster based schemes [12, 14], DP = DP based schemes [15]- [19], RW = the random walk based scheme [20], and ✓, ⧫, and ✗ = the anonymization scheme is vulnerable, conditionally vulnerable, and invulnerable (i.e., resistant) to the DA attack, respectively.

	Naive	EE	k-anony.	Cluster	DP	RW
BDK [26]	✓	✗	✗	✗	✗	✗
NS [2]	✓	✓	⧫	⧫	✓	✓
NSR [21]	✓	✓	⧫	⧫	✓	✓
NKA [22]	✓	⧫	⧫	⧫	✗	✗
DV [5]	✓	✓	⧫	⧫	✓	✓
RST [5]	✓	✓	⧫	⧫	✓	✓
RSM [5]	✓	✓	⧫	⧫	✓	✓
PFG [23]	✓	✓	⧫	⧫	✓	✓
YG [27]	✓	✓	⧫	⧫	✓	✓
DeA [25]	✓	✓	⧫	⧫	✓	✓
ADA [25]	✓	✓	⧫	⧫	✓	✓
KL [24]	✓	✓	⧫	⧫	✓	✓
JLSB [3]	✓	✓	⧫	⧫	✓	✓

4 Anonymization vs DA Analysis

As we analyzed in Tables 2 and 3, understanding the vulnerability/resistance of state-of-the-art graph data anonymization schemes against modern SDA attacks is still an open problem. After carefully analyzing existing anonymization and DA techniques, we summarize the ***vulnerability*** of existing anonymization schemes in Table 4. We further experimentally validate our analysis in Section 5. Below, we analyze and discuss the results in Table 4.

It has been shown in both academia and in practice that the naive ID removal anonymization cannot protect graph data's privacy. Therefore, naive anonymization is vulnerable to all the existing SDA attacks.

As we analyzed before, all other state-of-the-art anonymization schemes (e.g., EE, k-anony., Cluster, DP, and RW) are resistant to BDK attacks. Again, this is because an assumption of BDK attacks is that data is anonymized by the naive ID removal technique.

For EE based anonymization schemes ([6]), they are conditionally vulnerable to NKA [22] and vulnerable to all the other modern SDA attacks [2,3,25,27]. This is because although EE can partially modify the structure of

a graph, to preserve data utility, many structural properties, e.g., neighborhood, degree distribution, closeness/betweenness centrality distribution, and path length distribution, are generally preserved. Therefore, given an auxiliary graph consisting of the same or overlapping group of users with the anonymized graph, powerful DA heuristics can be designed based on these structural properties to break the privacy of EE based anonymization schemes. Furthermore, the availability of seed users make such heuristics more robust to the noise introduced by EE. For instance, NS breaks EE by employing degree and neighborhood similarity [2], DV, RST, and RSM break EE by employing path length and neighborhood similarity [5], DeA and ADA break EE by employing centrality similarity [25], etc. As we analyzed in Table 2, EE based anonymization schemes (e.g., *Add/Del*) may destroy graphs' community utility, and thus they are conditionally vulnerable to NKA [22].

k-anonymity based anonymization schemes ([7]- [10]) are conditionally vulnerable to modern SDA attacks [2, 3, 25, 27]. The reasons are as follows: k-anonymity is initially designed for traditional relational data, which makes a user semantically indistinguishable with $k-1$ other users. Unlike relational data, which are structurally independent of each other, users in graph data have strong structural correlation in addition to semantic similarity. When researchers extended k-anonymity to graph data, they extended the concept of traditional semantics to graph data as different structural properties (e.g., degree, neighborhood, and subgraph), and designed schemes to make k users structurally indistinguishable with respect to some structural semantics, i.e., degree, neighborhood, subgraph, etc. However, even if users in graph data cannot be distinguished with respect to some structural semantics, e.g., degree, neighborhood, subgraph, they can be de-anonymized by other structural semantics, e.g., path length distribution, closeness centrality, betweenness centrality, or the combinations of several structural semantics. Theoretically, the only way to make users indistinguishable with respect to all structural semantics is to make a graph *completely connected* or *disconnected*, which also implies that all the data utility is destroyed. Therefore, as long as some data utility is preserved in the anonymized data, k-anonymity based schemes are vulnerable to modern SDA attacks. The degree of vulnerability depends on how much data utility is preserved.

Cluster based schemes ([12, 14]) are also conditionally vulnerable to modern SDA attacks [2,3,25,27]. The analysis is similar to that of k-anonymity. The fundamental idea of cluster based schemes is to cluster users first and then to make the users within a cluster indistinguishable with respect to neighborhoods. Again, even if users are indistinguishable by neighborhoods, they can be de-anonymized by other structural semantics or the combinations of other semantics, e.g., centralities scores, path length distribution. Consequently, cluster based schemes are vulnerable as long as some data utility, especially graph utilities, are preserved in the anonymized data, and the vulnerability depends on the amount of data utility preserved.

DP and RW based schemes ([15]- [20]) are vulnerable to modern SDA attacks except NKA [22]. The reasons are as follows: First, they are designed with the objective of protecting the link privacy of graph data and no dedicated node privacy protection techniques are considered. Second, to protect link privacy, the edges are perturbed in DP based schemes and random walk paths are replaced by edges in the RW based scheme, both with a nice theoretical privacy guarantee. However, after the edge anonymization process, many data utilities, e.g., degree, path length distribution, are still preserved. This implies that, given an auxiliary graph, users are still de-anonymizable based on several structural semantics under DP and RW based schemes. Furthermore, as shown by Narayanan et al. in [21], link privacy can be breached after de-anonymizing the users in an anonymized graph (we also employ the same approach to break users' link privacy [1]). Again, as we analyzed in Table 2, since DP and RW based schemes cannot preserve data's community utility, they are resistant to NKA.

In summary, based on our analysis, state-of-the-art anonymization schemes are still vulnerable to modern DA attacks. The fundamental reasons are: first, existing anonymization schemes only ensure that graph data users are indistinguishable with respect to some structural semantics (properties). However, other structural semantics, especially global ones, and the combinations of multiple structural semantics can still enable effective DA of users; and second, as one of the main objectives, all the anonymization schemes try to preserve as much data utility as possible. However, data utility from the adversary's perspective is equivalent to structural information, which can be used along with an auxiliary graph for conducting powerful DA attacks.

5 SecGraph

As we found when discussing existing anonymization and DA techniques, they all have limitations when evaluating the techniques' performance. For instance, it is still an open problem to understand the resistance/vulnerability of state-of-the-art anonymization schemes against modern DA attacks. To address this open problem, we implement a Secure Graph data publishing/sharing (SecGraph) system.

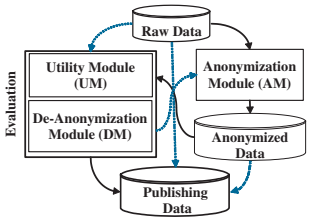

Figure 1: SecGraph: system overview.

5.1 System Overview

The overview of SecGraph is shown in Fig.1. SecGraph consists of three main modules: Anonymization Module (AM), Utility evaluation Module (UM), and DA evaluation Module (DM). The main functions of each module are briefly summarized as follows.

AM: the main function of this module is to anonymize raw graph data and generate anonymized data. In this module, we implement 11 state-of-the-art graph data anonymization schemes, including EE based algorithms [6], k-anonymity based algorithms and its variants [7–11], aggregation/class/cluster based algorithms [12–14], differential privacy based algorithms [15–17,19], and the random walk based algorithm [20].

UM: in this module, we evaluate raw/anonymized data's utility with respect to the 12 graph utility metrics and 7 application utility metrics as defined in Section 2.2. With the UM, we can determine whether the data to be published/shared (e.g., the anonymized data) satisfies required utility requirements. We can also evaluate how an anonymization algorithm preserves data utility.

DM: in this module, we implement 15 SDA algorithms (all the existing SDA algorithms, to the best of our knowledge). By this module, the security of data to be published/shared can be evaluated with real-world SDA attacks. More importantly, the effectiveness of an anonymization algorithm can be examined by this module, i.e., whether the anonymized data of an anonymization algorithm is resistant to modern SDA attacks.

We make further remarks on SecGraph and its modules and functions as follows.

(a) From Fig.1, raw data can be published/shared in multiple forms depending on the data owners' requirements on the security/privacy and utility of the data to be published. Each path in Fig.1 represents a data publishing scenario. For instance, the path *raw data* $\rightarrow$ *publishing data* means to publish the raw data directly. The path *raw data* $\rightarrow$ *AM* $\rightarrow$ *anonymized data* $\rightarrow$ *evaluation* $\rightarrow$ *publishing data* means that the raw data is anonymized first. Then, the anonymized data will be evaluated with respect to utility and/or practical de-anonymizability before actual publishing. The anonymization and evaluation process may be repeated several times until certain security and utility requirements are met.

(b) To the best of our knowledge, SecGraph is the first implemented uniform secure graph data publishing system, which systematically and comprehensively integrates state-of-the-art anonymization schemes, DA schemes, and graph/application utility measurements. The significance of SecGraph to the graph data anonymization and DA area lies in the following aspects. First, SecGraph enables data owners to conveniently and freely choose any modern anonymization algorithm to anonymize their data. They can also employ different evaluation modules to examine whether the anonymized data meets their security/privacy and utility requirements. Second, SecGraph is a uniform platform for testing and comparing different anonymization and DA algorithms. Previously, due to the lack of a uniform system, existing anonymization/DA algorithms are often proposed and implemented on separate platforms and different environments/settings. Consequently, a number of implementation and evaluation differences (e.g., particular assumptions, models, evaluation datasets, programming, testing environments, parameter settings) limit researchers' understanding of the performance of existing anonymization and DA algorithms in different scenarios. However, as a uniform platform, SecGraph can reduce the evaluation bias caused by implementation and testing differences as much as possible. Therefore, SecGraph allows data owners to choose and compare the actual performance of different data anonymization algorithms on their data and thus to make the best decision. Additionally, SecGraph allows data anonymization researchers to compare their anonymization schemes to existing solutions as well as to examine their schemes' resistance against modern DA attacks. SecGraph also allows data DA researchers to evaluate the performance of new DA attacks by de-anonymizing the anonymized data of state-of-the-art anonymization schemes. Therefore, SecGraph is helpful to both data owners and researchers in conveniently applying existing schemes, comprehensively understanding existing algorithms, and effectively developing new anonymization/DA techniques.

(c) Besides providing a uniform platform, SecGraph is an easily portable and extendable system. First, the algorithms in SecGraph are implemented in Java and thus it is system independent. Second, all the modules of SecGraph are independent of each other, which means that each module can work individually. Additionally, as shown in Fig.1, multiple modules can also work together to perform data anonymization, utility evaluation, and de-anonymizability evaluation. Third, all the schemes/measurements within each module are independent, which means that they can be implemented, evaluated, and employed independently. Furthermore, newly

developed anonymization/DA schemes and utility metrics can be easily integrated into SecGraph.

5.2 System Implementation

The implementation of SecGraph is as follows.

In the AM, we implement 11 algorithms, which cover all the categories of state-of-the-art anonymization techniques. Specifically, the implemented anonyzation algorithms are naive ID removal, two EE based algorithms *Add/Del* [6] and *Switch* [6], two k-anonymity based algorithms k-DA [8] and k-iso [10], two cluster based algorithms bounded t-means clustering [14] and union-split clustering [14], three DP based algorithms Sala et al.'s scheme [15], Proserpio et al.'s scheme [16, 17], and Xiao et al.'s scheme [19], and one RW based algorithm [20]. Note that, we do not implement all the algorithms discussed in Section 2.1 even though we cover all the categories. The implementation criteria includes representativeness, scalability, and practicality, which led us to implement the latest, scalable, and practical schemes.

In the UM, we implemented the 12 graph utility metrics and 7 application utility metrics as discussed in Section 2.2.

In the DM, we implement all the 15 SDA attacks discussed in Section 3.1. To the best of our knowledge, these are all of the existing SDA attacks.

5.3 SecGraph-based Analysis

5.3.1 Primary Datasets

The employed datasets for evaluation are *Enron*, an email network consisting of 36.7K users and .2M edges, and *Facebook*, a Facebook friendship network in the New Orleans area consisting of 63.7K users and .82M edges [3, 4].

5.3.2 Anonymization vs Utility

In this subsection, we evaluate the utility performance of anonymization algorithms. Due to the space limitation, we do not show the evaluation results of all the implemented algorithms. Particularly, we demonstrate the results of *Switch* [6], k-DA [8], *union-split clustering* [14], the improved version of Sala et al.'s DP scheme [15–17], and RW [20] which represent all the categories of anonymization algorithms. The evaluation methodology is that we first anonymize the original graph by an algorithm, and then measure how each data utility is preserved in the anonymized graph compared to the original graph. Specifically, when measuring utilities Deg., JD, PL, LCC, CC, BC, NC, NR, Infe., RX, and RE, we measure the *cosine similarity* between their distributions in the anonymized and original graphs; when measuring ED, GCC, and EV, we measure their *ratios* between the anonymized and original graphs; and when measuring MINS and CD, we measure their *Jaccard similarity* in the anonymized and original graphs.

We demonstrate the results in Table 5. (more results are available in [1]). The criteria for anonymization parameters settings are: (*i*) we follow the same/similar settings as in the original works of these anonymization schemes; and (*ii*) many data utilities can be preserved after anonymization. For the three graph utilities IM, SR, and SD, we only test them on small graphs, and put the results in [1]. We analyze the results in Table 5 as follows.

Generally, the evaluation results in Table 5 are consistent with our analysis in Table 2. Most anonymization algorithms can partially or conditionally preserve most graph and application utilities. Therefore, most of the anonymized data can be employed for graph analytics, data mining tasks, and graph applications.

Among all the graph utilities, JD and GCC are the most sensitive utilities to a graph's structure change, and thus they are the easiest ones to be destroyed by the anonymization algorithms. This is because these two utilities are very sensitive to edge changes. Even if the degree distribution of the anonymized data remains the same as the original data, the JD distribution and GCC may change significantly.

Compared to application utility, existing anonymization algorithms are better at preserving graph utility. For instance, most algorithms lost the RX utility and CD utility. This is because most application utilities depend on several graph utilities, e.g., the role of a user in RX depends on that user's degree, CC, BC, community attributes, and other structural characteristics. Therefore, application utilities are more easily affected than graph utilities, i.e., application utilities are more sensitive to graph's structural changes.

No anonymization scheme is optimal in preserving every data utility. For instance, *Switch* is better than k-DA on preserving Deg. and JD while it is worse than k-DA on preserving GCC and MINS, and DP is better than RW on preserving LCC and GCC while it is worse than RW on preserving Deg. Therefore, when choosing an anonymization algorithm, it is better to take into account the specific application. Furthermore, RW has the most utility loss, e.g., GCC, RX, MINS, and CD, which is also consistent with our analysis in Table 2. This is because that the graph's global structure is significantly changed in RW by replacing random walk paths with edges.

5.3.3 DA Evaluation

In this subsection, we evaluate the performance of modern DA attacks. As we analyzed before, BDK [26],

Table 5: Utility analysis of anonymization techniques. k is the number of modified edges for *Switch*, and the anonymization parameter for k-DA and Cluster, ε is the anonymization parameter for DP, t is the random walk step for RW, m is the number of edges in the original graph, and $\mathbb{D}$ is the diameter of the original graph ($\mathbb{D} = 11$ for Enron and $\mathbb{D} = 6$ for Facebook).

	Enron										**Facebook**									
Utility	*Switch* (vs. k)		k-DA (vs. k)		Cluster (vs. k)		DP (vs. ε)		RW (vs. t)		*Switch* (vs. k)		k-DA (vs. k)		Cluster (vs. k)		DP (vs. ε)		RW (vs. t)	
	$.05m$	$.1m$	5	50	5	50	300	50	2	$\mathbb{D}$	$.05m$	$.1m$	5	50	5	50	300	50	2	$\mathbb{D}$
Deg.	1	1	.9988	.9166	.9990	.9934	.9617	.8616	.9871	.9964	1	1	.9990	.9595	.9998	.9981	.9932	.9716	.9958	.9959
JD	.8725	.8338	.8928	.4183	.8216	.7055	.8496	.7363	.6972	.6438	.9941	.9804	.9947	.7328	.9872	.9024	.9755	.8263	.9678	.9362
ED	.9881	.9617	1.080	.9561	1.04	1.02	1.03	.9627	1.02	.9025	.9161	.8328	.9350	1.015	.9957	.9956	.9414	.9313	.9285	.8376
PL	.9954	.9887	.9891	.8934	.9994	.9905	.9565	.9839	.9963	.9657	.9618	.9159	.9999	.9946	.9999	1	.9960	.9653	.9706	.8965
LCC	.9830	.9631	.9972	.9809	.9966	.9797	.9528	.8328	.6785	.5985	.9204	.8303	.9998	.9983	.9968	.9947	.9793	.9437	.6239	.5543
GCC	.8967	.8013	.9921	.9283	.9774	.9097	.7755	.4609	.3107	.5383	.5180	.2241	.9847	.9986	.9766	.9937	.9522	.8702	.2552	.0334
CC	.9986	.9965	.9985	.9955	.9999	.9947	.9759	.9666	.9885	.9994	1	.9999	1	1	1	1	1	.9998	1	.9998
BC	.9859	.9812	.9691	.9019	.9936	.9733	.8360	.7406	.9613	.9246	.9787	.9494	.9790	.9515	.9983	.9897	.9779	.9518	.9935	.9669
EV	.9991	.9977	.9910	.8998	.9947	.9720	.9232	.8653	.9717	.9204	.9881	.9556	.9981	.9626	.9999	.9996	.9977	.9911	.9891	.9480
NC	.9984	.9962	.9999	.9991	.9996	.9956	.9977	.9596	.9042	.9028	.9995	.9986	1	1	1	1	.9987	.9934	.9928	.9942
NR	.9968	.9917	.9988	.9599	.9998	.9962	.9782	.8591	.9313	.8695	.9990	.9990	.9990	.9990	.9990	.9990	.9990	.9990	.9990	.9990
Infe.	.9627	.9597	.9604	.9411	.9427	.9413	.9662	.9593	.9664	.9446	.9748	.9704	.9758	.9695	.9730	.9719	.9730	.9699	.9788	.9778
PR	.9980	.9962	.9848	.8934	.9997	.9974	.9801	.9000	.8925	.9942	.9866	.9825	.9878	.9610	.9900	.9907	.9875	.9691	.9869	.9810
HS	.9991	.9977	.9910	.8998	.9947	.9720	.9232	.8653	.9717	.9204	.9326	.8780	.9711	.9789	.9648	.9625	.9626	.9322	.9283	.8655
AS	.9991	.9977	.9910	.8998	.9947	.9720	.9232	.8653	.9717	.9204	.9920	.9656	.9946	.9498	.9978	.9986	.9970	.9965	.9943	.9594
RX	.6575	.6009	.4561	.3173	.4512	.3685	.4196	.4116	.2955	.2680	.3494	.2608	.2974	.3139	.3902	.4652	.3483	.3134	.3250	.2772
RE	.9997	.9997	.9999	.9954	.9999	.9996	.9994	.9985	.9994	.9990	.9999	.9997	1	.9999	1	1	1	.9996	.9999	.9997
MINS	.7578	.6486	.9639	.9026	.9898	.9297	.7292	.3272	.1815	.1645	.6085	.4419	.9426	.9251	.9240	.9184	.8483	.7768	.2480	.1893
CD	.6251	.5411	.8454	.5339	.6794	.6692	.5095	.1028	.2531	.0569	.3536	.1986	.5043	.5887	.8558	.8523	.5027	.3213	.2860	.1205

RST [5], and RSM [5] are not scalable/practical; NSR [21] and DeA [25] are simplified versions of NS [2] and ADA [25], respectively; and NKA [22] actually depends on other attacks, e.g., NS. Therefore, here, we focus on evaluating the seven general, practical, and scalable DA attacks: NS [2], DV (we replace its seed identification phase with a scalable one) [5], PFG [23], YG [27], ADA [25], KL [24], and JLSB [3]. Furthermore, PFG and JLSB are seed-free and the other five attacks are seed-based.

First, employing the same Enron and Facebook datasets as before, we evaluate the DA performance of the seven DA attacks. The evaluation methodology is generally the same as in previous works [2, 3, 5, 22, 23, 25, 27]: we first randomly sample two graphs with probability s from the original data as the anonymized graph and auxiliary graph respectively, and then employ the auxiliary graph to de-anonymize the anonymized graph. Furthermore, for seed-based attacks, e.g., NS, DV, YG, ADA, and KL, we feed them 50 pre-identified seed mappings. The DA performance of the evaluated attacks with respect to different s is shown in Table 6. From Table 6, we have the following observations.

With the increase of s, more users can be successfully de-anonymized under each algorithm. The reason is evident. Since a large s implies that the anonymized graph and the auxiliary graph are more structurally similar, more accurate structural information can be employed by all the SDA algorithms. Hence, better DA performance can be achieved.

Generally, all the algorithms have their advantages in some specific scenarios, and no algorithm is the best in all the cases. For instance, to de-anonymize Enron, KL has the best performance when $s = .6$ while ADA has the best performance when $s = .95$. Multiple reasons are responsible for the results such as the similarity between the anonymized and auxiliary graphs, the density of the anonymized/auxiliary graph, the heuristics employed by an algorithm, etc.

According to the results, NS is more suitable for the scenarios where the anonymized and auxiliary graphs are highly similar while unsuitable when they are not sufficiently similar, e.g., it can successfully de-anonymize 95.27% Facebook users when $s = .95$ while only 0.18% users when $s = .6$. The reason is because NS mainly employs local graph structural properties to adaptively conduct user DA, and thus is sensitive to users' local structural characteristics. When s is small, most users are indistinguishable with respect to their local structures, e.g., degree, followed by poor DA performance.

Compared to NS, the other attacks, especially DV, PFG, ADA, and JLSB, are more stable even with a small s. For instance, when $s = .6$, DV, PFG, ADA, and JLSB can successfully de-anonymize 15.63%, 10.87%, 15.68%, and 14.73% Facebook users, respectively. This is because these attacks mainly employ global graph characteristics (e.g., clossness centrality, the distance vector to seeds) to perform the DA, which are more resilient to noise.

Table 6: Performance of DA attacks. s is the probability of generating the auxiliary and anonyized graphs from the original graph. Each value, e.g., 0.1277, in the table indicates the ratio of successfully de-anonymized users.

s	De-anonymize Enron							De-anonymize Facebook						
	NS	DV	PFG	YG	ADA	KL	JLSB	NS	DV	PFG	YG	ADA	KL	JLSB
.60	.0037	.1277	.0739	.0310	.1305	.1596	.1191	.0018	.1563	.1087	.2832	.1568	.0599	.1473
.65	.0039	.1601	.0937	.0410	.1651	.1814	.1460	.0020	.1998	.1402	.3346	.2005	.0747	.1799
.70	.0054	.1969	.1397	.0725	.2013	.2026	.1723	.0031	.2437	.1523	.4124	.2444	.0841	.2094
.75	.0055	.2244	.1349	.1004	.2307	.2152	.1958	.8712	.3068	.2041	.4554	.3078	.1196	.2574
.80	.0061	.2841	.1837	.1014	.2896	.2519	.2474	.9056	.3802	.2586	.4970	.3805	.1508	.3042
.85	.3420	.3481	.2180	.1531	.3522	.3123	.2971	.9231	.4561	.3073	.5402	.4576	.1817	.3559
.90	.3660	.4004	.2736	.1885	.4043	.3389	.3443	.9414	.5659	.3977	.5737	.5670	.2552	.4289
.95	.3937	.5814	.4370	.2277	.5898	.5209	.5438	.9527	.7407	.5584	.6071	.7422	.3989	.5542

For the seed-free attacks, PFG and JLSB, they can achieve comparable performance as seed-based attacks in most scenarios even without any seed information. For instance, when $s = .95$, PFG and JLSB can de-anonymize 43.7% and 54.38% Enron users, respectively, which are better than several seed-based algorithms and further demonstrate the power of structure-based attacks. The reason for the effectiveness of seed-free attacks is that in most cases, the combination of a user's local and global structural characteristics, e.g., degree, neighborhood degree distribution, closeness/betweenness centrality, is sufficient to distinguish him/her from other users.

5.3.4 Robustness of Modern SDA Attacks

The robustness of modern DA attacks with respect to graph noise (e.g., adding fake edges and deleting true edges) has been extensively evaluated in existing works [2,3,5,25]. However, to the best our knowledge, no existing work has evaluated the robustness of any seed-based de-anonymizaton attack to incorrect seed mappings. Employing Enron and Facebook, we address this open issue by conducting such an evaluation and the results are shown in Table 7. We analyze the results in Table 7 as follows.

Generally, all the DA algorithms are robust with respect to incorrect seed mappings in most scenarios. This is because during the DA process, most algorithms also employ other seed-independent structural properties, e.g., degree, closeness/betweenness centrality, in addition to relying on seed-dependent structural properties. Even for the pure seed-based DA attacks, e.g., YG and KL, they perform DA in the decreasing order of user degrees. Therefore, the negative impacts of incorrect seed mappings can be partially offset, i.e., even with some incorrect seed mappings, many users are still distinguishable with respect to their structural characteristics.

For all algorithms, when incorrect seed mappings increase, fewer users can be correctly de-anonymized. The reason is evident: more incorrect seed mappings imply more incorrect seed-dependent structural information is provided to each algorithm, followed by the degradation of the DA performance of each algorithm.

When de-anonymizing Enron, the performance of NS has a significant drop when the percentage of incorrect seed mappings is increased from 8% to 10%. This is because of the seed transitional phenomena as observed in [2], i.e., when the correct effective seed-dependent structural information is below/above some crucial threshold, NS's performance has a significant transition.

DV is much more stable than other algorithms. This is because it is a pure global structure-based attack and thus incorrect seed mappings have minimum impact on it.

5.3.5 Anonymization vs DA

Now, we evaluate the effectiveness of state-of-the-art anonymization techniques against modern DA attacks employing Enron and Facebook. The methodology is that we first employ different anonymization techniques to anonymize Enron/Facebook. Then, we sample an auxiliary graph from Enron/Facebook with probability s. Finally, we employ different DA algorithms to de-anonymize the anonymized data using the auxiliary graph. We show the results in Table 8 and analyze the results as follows.

All the state-of-the-art graph anonymization algorithms are vulnerable to some or all of the modern SDA attacks, which confirmed our analytical results in Table 4. For instance, when $s = .85$, NS can still successfully de-anonymize more than 80% Facebook users anonymized by *Switch*, k-DA, Cluster, or DP, and DV can successfully de-anonymize 15.3% Facebook users anonymized by RW ($t = 2$). Similarly, when $s = .85$, NS can successfully de-anonymize more than 35% Enron users anonymized by k-DA ($k = 5$), Cluster ($k = 5, 50$), YG can successfully de-anonymize 13.73% and 15.49% Enron users anonymized by *Switch* ($k = .05m$) and DP ($\varepsilon = 300$) respectively, and DV can successfully de-anonymize 19.23%/24.12% Enron users anonymized by RW with $t = 2/11$. Based on the results, we conclude that modern SDA attacks are very powerful. As

Table 7: DA robustness with respect to seed errors. Each algorithm is provided with 50 seed mappings, and Λ_e/Λ indicates the percentages of incorrect seed mappings. Each value in the table indicates the ratio of successfully de-anonymized users.

$\frac{\Lambda_e}{\Lambda}$	**De-anonymize Enron**					**De-anonymize Facebook**				
	NS	DV	YG	ADA	KL	NS	DV	YG	ADA	KL
4%	.341	.342	.148	.336	.302	.922	.456	.537	.442	.183
6%	.341	.342	.133	.329	.303	.917	.456	.528	.440	.183
8%	.338	.348	.135	.329	.310	.918	.456	.542	.428	.184
10%	.007	.348	.147	.323	.310	.918	.456	.536	.420	.182
12%	.007	.348	.142	.313	.311	.915	.456	.529	.414	.185
14%	.006	.348	.112	.306	.307	.916	.456	.526	.403	.186
16%	.006	.348	.129	.297	.303	.916	.456	.525	.394	.184
18%	.006	.348	.099	.293	.308	.913	.456	.533	.380	.183
20%	.006	.348	.126	.285	.306	.913	.456	.518	.356	.179
22%	.005	.348	.125	.280	.303	.912	.456	.531	.347	.182
24%	.005	.348	.116	.268	.304	.910	.456	.521	.332	.180
26%	.005	.348	.118	.255	.303	.889	.456	.528	.319	.179
28%	.004	.348	.112	.253	.300	.886	.456	.520	.309	.182
30%	.004	.348	.120	.247	.307	.884	.456	.522	.283	.180
32%	.004	.348	.106	.235	.305	.888	.456	.521	.270	.178
34%	.004	.348	.081	.230	.304	.887	.456	.521	.259	.178
36%	.004	.348	.084	.216	.300	.889	.456	.505	.245	.182
38%	.004	.347	.096	.199	.301	.888	.456	.493	.230	.178
40%	.004	.347	.065	.186	.302	.886	.456	.505	.214	.179
42%	.003	.347	.071	.182	.302	.882	.456	.516	.195	.181
44%	.003	.347	.106	.169	.303	.881	.456	.495	.185	.180
46%	.003	.347	.050	.160	.299	.881	.456	.480	.173	.177
48%	.003	.347	.059	.153	.297	.881	.456	.497	.161	.180
50%	.002	.347	.063	.146	.298	.874	.456	.475	.148	.176

Table 8: Anonymization vs DA. The seed-based algorithms are provided with 50 seeds and the anonymization parameters are chosen according to the same criteria as in Table 5.

	s	**Enron**										**Facebook**									
		Switch (k)		k-DA (k)		Cluster (k)		DP (ε)		RW (t)		*Switch* (k)		k-DA (k)		Cluster (k)		DP (ε)		RW (t)	
		5	10	5	50	5	50	300	50	2	$\mathbb{D}$	5	10	5	50	5	50	300	50	2	$\mathbb{D}$
NS	.85	.0072	.0052	.3702	.0088	.3722	.3707	.0091	.0055	.0015	.0015	.8973	.8247	.9454	.9402	.9456	.9442	.9317	.8914	.0008	.0006
	.90	.0077	.0054	.3822	.0105	.3900	.3839	.0095	.0060	.0015	.0015	.9063	.8427	.9520	.9495	.9519	.9508	.9393	.8944	.0008	.0007
	.95	.3577	.0064	.4033	.0418	.4049	.4064	.3946	.0064	.0015	.0016	.9162	.8583	.9570	.9559	.9569	.9558	.9453	.9130	.0000	.0007
DV	.85	.1261	.0813	.1433	.0437	.2120	.1408	.1160	.0701	.1923	.2412	.1716	.0926	.2411	.0588	.3340	.3368	.2324	.0736	.1530	.1271
	.90	.1546	.0956	.1765	.0517	.2564	.1637	.1394	.0733	.2129	.2169	.2124	.1147	.2999	.0758	.4113	.4090	.3623	.0802	.1604	.1322
	.95	.2121	.1366	.2548	.0753	.3745	.2215	.1821	.0858	.2072	.2190	.3006	.1586	.4210	.1161	.5767	.5656	.4087	.1016	.1591	.1332
PFG	.85	.0667	.0422	.0692	.0214	.1116	.0683	.0489	.0365	.1578	.2131	.0706	.0395	.0703	.0154	.1191	.1155	.0891	.0206	.1349	.1190
	.90	.0805	.0478	.0810	.0263	.1317	.0789	.0571	.0390	.1711	.2012	.0978	.0497	.0946	.0213	.1480	.1595	.1870	.0223	.1382	.1217
	.95	.1193	.0695	.1123	.0353	.1978	.0952	.0755	.0479	.1714	.2074	.1378	.0725	.1317	.0332	.2034	.2330	.1756	.0295	.1397	.1216
YG	.85	.1373	.0969	.1646	.0289	.1576	.1570	.1549	.0664	.0394	.0323	.5437	.5056	.5816	.5086	.5897	.5805	.5404	.4347	.0356	.0210
	.90	.1716	.1037	.1612	.0253	.1868	.1710	.1577	.0736	.0404	.0342	.5681	.5182	.6089	.5129	.6036	.5980	.5702	.4818	.0372	.0222
	.95	.1730	.1197	.2155	.3785	.1971	.2064	.1884	.0838	.0418	.0348	.5821	.5439	.6208	.5504	.6223	.6190	.5716	.4538	.0346	.0231
ADA	.85	.1262	.0820	.1468	.0445	.2130	.1418	.1160	.0701	.0771	.0731	.1724	.0926	.2425	.0603	.3358	.3379	.2337	.0749	.0985	.0725
	.90	.1543	.0964	.1795	.0534	.2588	.1652	.1394	.0729	.0855	.0704	.2129	.1146	.3026	.0776	.4124	.4103	.3639	.0823	.1008	.0764
	.95	.2139	.1381	.2605	.0768	.3777	.2230	.1823	.0855	.0872	.0733	.3019	.1589	.4245	.1186	.5780	.5667	.4105	.1038	.1041	.0784
KL	.85	.0904	.0811	.0997	.0357	.0965	.0689	.0745	.0331	.0900	.0729	.0799	.0764	.0819	.0683	.0788	.0762	.0769	.0313	.1099	.0737
	.90	.1077	.0970	.1202	.0549	.1134	.0918	.0874	.0319	.0939	.0744	.0979	.0939	.1013	.0848	.0960	.0863	.1249	.0317	.1099	.0715
	.95	.1381	.1150	.1936	.0978	.2052	.1686	.1719	.0376	.0994	.0776	.1350	.1331	.1418	.1265	.1294	.1206	.1450	.0600	.1171	.0754
JLSB	.85	.0692	.0440	.0798	.0234	.1248	.0854	.0886	.0656	.0709	.0720	.1453	.0786	.2025	.0595	.2618	.2673	.1958	.0768	.0901	.0681
	.90	.0886	.0536	.1046	.0296	.1618	.1135	.1070	.0664	.0767	.0728	.1673	.0911	.2335	.0708	.3001	.3094	.3050	.0777	.0911	.0699
	.95	.1846	.1189	.2381	.0746	.3317	.2319	.1449	.0814	.0838	.0740	.2180	.1174	.3111	.1096	.3983	.3924	.3142	.0950	.0940	.0734

we analyzed in Table 4, two fundamental reasons make state-of-the-art graph anonymization algorithms vulnerable. First, in existing graph anonymization schemes, graph users are only indistinguishable with respect to some structural properties/semantics. However, several other structural properties or the combinations of them can still enable effective graph user DA. Furthermore, the design philosophy of existing anonymization schemes is to preserve as much data utility as possible. However, data utility can be used to conduct powerful SDA attacks. Therefore, it is still an open problem to design effective graph data anonymization algorithms which can defend against modern SDA attacks.

Generally, when s is large and the anonymization level (e.g., k for *Switch* and k-DA) is low, more users can be correctly de-anonymized. The reason is straightforward. A large s implies more structural information of the original graph can be preserved in the auxiliary graph and thus more accurate structural characteristics can be employed for DA. Meanwhile, a low anonymization level implies less perturbation applied to the original graph's structure followed by the anonymized graph is more structurally similar to the original graph and thus is easier to be de-anonymized.

Among all the DA attacks, NS, YG, and ADA perform better than other attacks in most scenarios. This is because they mainly employ the combinations of several local structural characteristics to conduct the DA. According to our utility analysis in Table 2 and evaluation results in Table 5, most existing anonymization algorithms can preserve most graph utilities, especially the local graph utilities, e.g., Deg., LCC. It turns out that the graph utility preserved by anonymization algorithms can be used by DA attacks to conduct effective DA. Therefore, in the scenarios where an anonymization algorithm preserves more data utility, the corresponding dataset is more vulnerable to modern SDA attacks.

Among all the anonymization techniques, RW has better performance than others in most of the cases. The reason is that, a random walk path of length t is replaced by an edge in RW. It follows that the original graph structure is significantly changed. Therefore, a RW-anonymized graph is more resistant to DA attacks. However, RW achieves such DA resistance at the cost of sacrificing more data utility compared with other anonymization techniques, which is consistent with our utility analysis and evaluation results in Tables 2 and 5. Furthermore, we can also find that in most scenarios, existing anonymization techniques can degrade the performance of SDA attacks. Again, as shown in Tables 2 and 5, some data utilities are also degraded/lost.

6 Future Research and Challenges

In this section, we discuss the future research directions and challenges of graph data anonymization and DA.

Graph Data Anonymization. According to our analytical results in Table 4 and evaluation results in Table 8, all the state-of-the-art anonymization techniques, e.g., k-anonymity based schemes, DP based schemes, are vulnerable to modern SDA attacks. Their vulnerability depends on how much data utility is preserved in the anonymized data. Therefore, it is very difficult, if not impossible, to develop effective and universal graph data anonymization techniques to defend against modern SDA attacks. The main challenges are two-folds. First, guaranteeing data utility is one of the primary objectives when publishing/sharing graph data. However, as we explained before, the preserved graph and application utilities enable adversaries to conduct large-scale DA attacks. Therefore, it is a big challenge to effectively anonymize graph data with desired data utility preservation and without enabling adversaries to utilize these data utilities. Second, many local and global structural characteristics (or, structural scemantics), e.g., Deg., LCC, CC, BC, are embedded in graph data's structure. Existing anonymization techniques can only make graph users structurally indistinguishable with respect to one or several semantics, e.g., degree and neighborhood. However, as we explained before, in many scenarios, several other structural semantics and their combinations are sufficient to enable a SDA attack to de-anonymize graph users. Therefore, it is also a key challenge to make graph users structurally indistinguishable with respect to most, if not all, structural semantics.

Considering that it is difficult to seek a tradeoff between generic utility and anonymity, a promising research direction could be developing *application-oriented* anonymization techniques. Instead of preserving as much data utility as possible, one only considers some specific application-aware utility when designing the anonymization techniques. For instance, although RW loses more data utility than most existing graph anonymization techniques, it achieves better anonymity and meanwhile supports some application utility, e.g., sybil detection [20].

Graph Data DA. Based on our DA evaluation results, future DA research may follow two directions.

First, it is interesting to study how to combine the advantages of different algorithms and develop new stable and improved DA schemes. To achieve this, the challenge is to decide which structural characteristics should be employed and how to use these characteristics during the DA process. This is because some structural characteristics are local (e.g., Deg.) while others are global (e.g., CC and BC). Therefore, it is better to seek a bal-

ance between the employed local and global structural semantics. Additionally, some structural characteristics may carry similar structural semantics, and thus simultaneously employing such characteristics will not lead to too much improvement.

Second, instead of trying to design a uniformly optimal DA algorithm, it is better to develop some anonymization technique-oriented and application-aware DA schemes. This is because, for some anonymization algorithms, e.g., most k-anonymity based schemes, they mainly achieve anonymity by local graph perturbation. In this scenario, the global graph characteristics based DA algorithms will be more effective. On the other hand, for some anonymization algorithms, e.g., *Add/Del* and RW, they mainly achieve anonymity through global graph perturbation. Therefore, the local graph characteristics based DA schemes will be better at de-anonymizing the data anonymized by these techniques. Furthermore, according to our DA evaluation experience, some DA attacks are more effective to de-anonymize dense graphs, e.g., NS and JLSB, while some other attacks are more effective to de-anonymize sparse graphs, e.g., DV, PFG. Therefore, when developing new DA algorithms, it is helpful to take into account both the attacked anonymization technique and the attacked application.

More Future Work. In this paper, we focus on implementing and evaluating graph data anonymization and DA techniques. It is also interesting to integrate the anonymization and DA techniques for other data types, e.g., relational data. In the future, we propose to develop a uniform and open-source evaluation system supporting multi-type data anonymization and DA.

7 Conclusion

In this paper, we propose, implement, and evaluate SecGraph (available at [1]), an *open-source* secure graph data publishing/sharing system. Within SecGraph, we systematically analyze, implement, and evaluate 11 graph data anonymization algorithms, 19 data utility metrics, and 15 modern SDA attacks. To the best of our knowledge, SecGraph is the first such system that provides a *uniform platform* enabling data owners to anonymize and evaluate the security of their data, and simultaneously enabling researchers to conduct fair studies of existing or newly developed anonymization/DA techniques. Leveraging SecGraph, we conduct extensive experimental evaluations. The results demonstrate that (*i*) most anonymization schemes can partially or conditionally preserve most graph utility but lose some application utility; (*ii*) no DA attack is optimum in all scenarios. The actual DA performance depends on several factors; and (*iii*) all the state-of-the-art anonymization schemes are vulnerable to modern SDA attacks. Based on our findings and analysis, we discuss the future research directions and challenges of graph data anonymization and DA.

8 Acknowledgments

The authors are very grateful to the anonymous reviewers for their time and valuable comments. The authors are also grateful to the following researchers in developing SecGraph: Ada Fu, Michael Hay, Davide Proserpio, Qian Xiao, Shirin Nilizadeh, Jing S. He, Wei Chen, and Stanford SNAP developers.

This work was partly supported by NSF-CAREER-CNS-0545667. Prateek Mittal was supported in part by the NSF under the grant CNS-1409415.

References

[1] S. Ji, W. Li, P. Mittal, X. Hu, and R. Beyah. Secgraph. *http://www.ece.gatech.edu/cap/secgraph/*.

[2] A. Narayanan and V. Shmatikov. De-anonymizing social networks. *S&P*, 2009.

[3] S. Ji, W. Li, M. Srivatsa, and R. Beyah. Structural data de-anonymization: Quantification, practice, and implications. *CCS*, 2014.

[4] S. Ji, W. Li, N. Gong, P. Mittal, and R. Beyah. On your social network de-anonymizablity: Quantification and large scale evaluation with seed knowledge. *NDSS*, 2015.

[5] M. Srivatsa and M. Hicks. Deanonymizing mobility traces: Using social networks as a side-channel. *CCS*, 2012.

[6] X. Ying and X. Wu. Randomizing social networks: a spectrum preserving approach. *SDM*, 2008.

[7] B. Zhou and J. Pei. Preserving privacy in social networks against neighborhood attacks. *ICDE*, 2008.

[8] K. Liu and E. Terzi. Towards identity anonymization on graphs. *SIGMOD*, 2008.

[9] L. Zou, L. Chen, and M. T. Özsu. K-automorphism: A general framework for privacy preserving network publication. *VLDB*, 2009.

[10] J. Cheng, A. Fu, and J. Liu. K-isomorphism: Privacy preserving network publication against structural attacks. *SIGMOD*, 2010.

[11] M. Yuan, L. Chen, and P. Yu. Personalized privacy protection in social networks. *VLDB*, 2010.

[12] M. Hay, G. Miklau, D. Jensen, D. Towsley, and P. Weis. Resisting structural re-identification in anonymized social networks. *VLDB*, 2008.

[13] S. Bhagat, G. Cormode, B. Krishnamurthy, and D. Srivastava. Class-based graph anonymization for social network data. *VLDB*, 2009.

[14] B. Thompson and D. Yao. The union-split algorithm and cluster-based anonymization of social networks. *ASIACCS*, 2009.

[15] A. Sala, X. Zhao, C. Wilson, H. Zheng, and B. Zhao. Sharing graphs using differentially private graph models. *IMC*, 2011.

[16] D. Proserpio, S. Goldberg, and F. McSherry. A workflow for differentially-private graph synthesis. *WOSN*, 2012.

[17] D. Proserpio, S. Goldberg, and F. McSherry. Calibrating data to sensitivity in private data analysis. *VLDB*, 2014.

[18] Y. Wang and X. Wu. Preserving differential privacy in degree-correlation based graph generation. *TDP*, 2013.

[19] Q. Xiao, R. Chen, and K. Tan. Differentially private network data release via structural inference. *KDD*, 2014.

[20] P. Mittal, C. Papamanthou, and D. Song. Preserving link privacy in social network based systems. *NDSS 2013*.

[21] A. Narayanan, E. Shi, and B. Rubinstein. Link prediction by de-anonymization: How we won the kaggle social network challenge. *IJCNN*, 2011.

[22] S. Nilizadeh, A. Kapadia, and Y.-Y. Ahn. Community-enhanced de-anonymization of online social networks. *CCS*, 2014.

[23] P. Pedarsani, D. R. Figueiredo, and M. Grossglauser. A bayesian method for matching two similar graphs without seeds. *Allerton*, 2013.

[24] N. Korula and S. Lattanzi. An efficient reconciliation algorithm for social networks. *VLDB*, 2014.

[25] S. Ji, W. Li, M. Srivatsa, J. He, and R. Beyah. Structure based data de-anonymization of social networks and mobility traces. *ISC*, 2014.

[26] L. Backstrom, C. Dwork, and J. Kleinberg. Wherefore art thou r3579x? anonymized social networks, hidden patterns, and structural steganography. *WWW*, 2007.

[27] L. Yartseva and M. Grossglauser. On the performance of percolation graph matching. *COSN*, 2013.

[28] D. Goodin. http://arstechnica.com/tech-policy/2014/06/poorly-anonymized-logs-reveal-nyc-cab-drivers-detailed-whereabouts/.

[29] P. Samarati. Protecting respondents' identities in microdata release. *TKDE*, 2001.

[30] J. Brickell and V. Shmatikov. The cost of privacy: Destruction of data-mining utility in anonymized data publishing. *KDD*, 2008.

[31] C. Dwork. Differential privacy. *ICALP*, 2006.

[32] R. Chen, G. Acs, and C. Castelluccia. Differentially private sequential data publication via variable-length n-grams. *CCS*, 2012.

[33] R. Albert, H. Jeong, and A.-L. Barabasi. Error and attack tolerance of complex networks. *Nature*, 2000.

[34] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 1998.

[35] K. Henderson et al. Roix: Structural role extraction & mining in large graphs. *KDD*, 2012.

[36] S. Garriss, M. Kaminsky, M. J. Freedman, B. Karp, D. Mazières, and H. Yu. Re: Reliable email. *NSDI*, 2006.

[37] W. Chen, Y. Wang, and ang S. Yang. Efficient influence maximization in social networks. *KDD*, 2009.

[38] J. He, S. Ji, R. Beyah, and Z. Cai. Minimum-sized influential node set selection for social networks under the independent cascade model. *Mobihoc*, 2014.

[39] J Yang and J. Leskovec. Overlapping community detection at scale: A nonnegative matrix factorization. *WSDM*, 2013.

[40] S. Marti, P. Ganesan, and H. Garcia-Molina. Sprout: P2p routing with social networks. *P2P&DB*, 2004.

[41] H. Yu, P. B. Gibbons, M. Kaminsky, and F. Xiao. Sybillimit: A near-optimal social network defense against sybil attacks. *S&P*, 2008.