Active Re-identification Attacks on Periodically Released Dynamic Social Graphs

Xihui Chen, Ema Këpuska, Sjouke Mauw, and Yunior Ramírez-Cruz

SnT, DCS, University of Luxembourg,
6, av. de la Fonte, L-4364 Esch-sur-Alzette, Luxembourg
{xihui.chen, sjouke.mauw, yunior.ramirez}@uni.lu,
kepuskaema@gmail.com

Abstract. Active re-identification attacks pose a serious threat to privacy-preserving social graph publication. Active attackers create fake accounts to enforce structural patterns that can be used to re-identify legitimate users on published anonymised graphs, even without additional background knowledge. So far, this type of attacks has only been studied in the scenario where the inherently dynamic social graph is published once. In this paper, we present the first active re-identification attack in the more realistic scenario where a dynamic social graph is periodically published. Our new attack leverages tempo-structural patterns, created by a dynamic set of sybil nodes, for strengthening the adversary. We evaluate our new attack through a comprehensive set of experiments on real-life and synthetic dynamic social graphs. We show that our new attack substantially outperforms the most effective static active attack in the literature by increasing success probability by at least two times and efficiency by at least 11 times. Moreover, we show that, unlike the static attack, our new attack remains at the same level of efficiency as the publication process advances. Additionally, we conduct a study on the factors that may thwart our new attack, which can help design dynamic graph anonymisation methods displaying a better balance between privacy and utility.

Keywords: dynamic social graphs · privacy-preserving publication · re-identification attacks · active adversaries

1 Introduction

Social graphs are a valuable source of data for conducting societal studies, market analyses, and other forms of complex data analysis. Analysts profit from social graph data for conducting their studies, whereas data owners find additional business and public service opportunities in making these data available to third parties. However, releasing social network data raises serious privacy concerns, due to the sensitive nature of the information contained in social graphs. Thus, the data needs to be properly sanitised before publication. It has been shown that *pseudonymisation*, i.e. removing users' identities and personally identifying information from the data, is insufficient for protecting sensitive information, as most users can be unambiguously *re-identified* in the pseudonymised graph by means of simple structural patterns [10, 15, 2]. User re-identification subsequently allows a malicious agent, or *adversary*, to infer relations between users,

group affiliations, etc. A method allowing an adversary to re-identify (a subset of) the users in a sanitised social graph is called a *re-identification attack*. Numerous anonymisation methods have been proposed for publishing social graphs that effectively resist re-identification attacks, e.g. [10, 20, 3, 26, 27, 22, 12, 13]. These methods depend on an adversary model, which encodes assumptions about the adversary capabilities. There are two classes of adversaries in social graph publication. On the one hand, *passive* adversaries exploit publicly available information obtainable from online resources, public records, etc., without interacting with the social network before publication. On the other hand, *active* adversaries interact with the network before the sanitised dataset is released. Active adversaries operate by inserting fake accounts in the network, commonly called *sybil nodes*, and creating connection patterns between these fake accounts and a set of legitimate users, the *victims*. After the publication of the sanitised graph, the attacker uses these unique patterns for re-identifying the victims. Active adversaries have been shown to be a serious threat to social graph publication [2, 14], as they remain plausible even if no public background knowledge is available.

Social networks are inherently dynamic. Moreover, analysts require datasets containing dynamic social graphs for conducting tasks such as community evolution analysis [4], link prediction [11] and link persistence analysis [17]. Despite the need for sanitised dynamic social graphs, studies on graph anonymisation have overwhelmingly focused on the scenario where a social graph is released only once. The rather small number of studies on dynamic social graph publication have addressed passive adversary models only. Thus, the manners in which active adversaries can profit from a dynamic publication scenario remain unknown. In this paper, we remedy this situation by formulating active re-identification attacks in the scenario of dynamic social graphs. We consider a scenario where snapshots of the underlying dynamic graph are *periodically* taken, sanitised, and published. We model active adversaries whose knowledge consists in *tempo-structural* patterns, instead of exclusively structural patterns as those used by the original (static) active adversaries. Moreover, in our model the adversary knowledge is incremental and evolves along the publication process. The new dynamic active attack is more effective than the alternative of executing independent static attacks on different snapshots. Furthermore, it is also considerably more efficient than the previous attacks, because it profits from temporal patterns to accelerate several of its components.

Our contributions. The main contributions of this paper are listed in what follows:

- We formulate, for the first time, active re-identification attacks in the scenario of periodically released dynamic social graphs. We describe an instance of the new attack strategy based on tempo-structural patterns for re-identification.
- We conduct a comprehensive set of experiments on real-life and synthetic dynamic social graphs, which demonstrate that the dynamic active attack is at least two times more effective than the alternative of repeatedly executing the strongest active attack reported in the literature for the static scenario [14].
- Our experiments also show that, as the number of snapshots grows, the dynamic active attack runs at least 11 times faster than the static active attack from [14].
- We analyse the factors that affect the effectiveness of our new attack. The conclusions of this study serve as a starting point for the development of anonymisation methods for the periodical publication scenario.

2 Related Work

Re-identification attacks are a relevant threat for privacy-preserving social graph publication methods that preserve a mapping between the real users and a set of pseudonymised nodes in the sanitised release, e.g. [10, 20, 3, 26, 27, 22, 12, 13]. Depending on the manner in which the attacker obtains the knowledge used for re-identification, these attacks can be divided into two classes: passive and active attacks. *Passive* adversaries collect publicly available knowledge, such as public profiles in other social networks, and searches the sanitised graph for vertices with an exact or similar profile. For example, Narayanan and Shmatikov [15] used information from Flickr to re-identify users in a pseudonymised subgraph of Twitter. Subsequently, a considerable number of passive attacks have been proposed, e.g. [15, 23, 18, 16, 7, 6]. On the other hand, *active* adversaries interact with the real network before publication, and force the existence of the structural patterns that allow for re-identification after release. The earliest examples of active attacks are the *walk-based attack* and the *cut-based attack*, introduced by Backstrom *et al.* in [2]. Both attacks insert sybil nodes in the network, and create connection patterns between the sybil nodes that allow their efficient retrieval in the pseudonymised graph. In both attacks, the connection patterns between sybil nodes and victims are used as unique fingerprints allowing re-identification once the sybil subgraph is retrieved. Due to the low resilience of the walk-based and cut-based attacks, a robust active attack was introduced by Mauw *et al.* in [14]. The robust active attack introduces noise-tolerant sybil subgraph retrieval and fingerprint mapping, at the cost of larger computational complexity. The attack proposed in this paper preserves the noise resiliency of the robust active attack, but puts a larger emphasis on temporal consistency constraints for reducing the search space. As a result, for every re-identification attempt, our attack is comparable to the original walk-based attack in terms of efficiency, and to the robust active attack in terms of resilience against modifications in the graph.

Notice that, by itself, the use of connection fingerprints as adversary knowledge does not make an attack active. The key feature of an active attack is the fact that the adversary interacts with the network to enforce the existence of such fingerprints. For example, Zou *et al.* [27] describe an attack that uses the distances of the victims to a set of hubs as fingerprints. This is a passive attack, since hubs exist in the network without intervention of the attacker.

The attacks discussed so far assume a single release scenario. A smaller number of works have discussed re-identification in a dynamic scenario. Some works assume an adversary who can exploit the availability of multiple snapshots, although they only give a coarse overview of the increased adversary capabilities, without giving details on attack strategies. Examples of these works are [21], which models a passive adversary that knows the evolution of the degrees of all vertices; and [27], which models another passive adversary that knows the evolution of a subgraph in the vicinity of the victims. An example of a full dynamic de-anonymization method is given in [5]. Although they do not model an active adversary, the fact that the method relies on the existence of a seed graph makes it potentially extensible with an active first stage for seed re-identification, as done for example in [19]. Our attack differs from the methods above in the fact that it uses an evolving set of sybil nodes that dynamically interact with the network and adapt to its evolution.

3 A Dynamic Active Attack on Periodical Graph Publication

In this section we describe the scenario where the owner of a social network periodically publishes sanitised snapshots of the underlying dynamic social graph, accounting for the presence of active adversaries. We describe this scenario in the form of an attacker-defender game between the data owner and the active adversary. We first introduce the basic notation and terminology, and give an overview of the entire process. Then, we introduce the notions of temporal consistency, which are the backbone of the new attack strategy. Finally, we give a detailed description of the publication process, along with an instantiation of the attack strategy, which exploits tempo-structural patterns and temporal consistency for dynamic re-identification.

3.1 Notation and Terminology

We represent a dynamic social graph as a sequence $\mathcal{G} = (G_1, G_2, \ldots, G_i, \ldots)$, where each G_i is a static graph called the i-th *snapshot* of $\mathcal{G}$. Each snapshot of $\mathcal{G}$ has the form $G_i = (V_i, E_i)$, where V_i is the set of vertices (also called *nodes* indistinctly throughout the paper) and $E_i \subseteq V_i \times V_i$ is the set of edges. We will use the notations V_G and E_G for the vertex and edge sets of a graph G. In this paper, we assume that graphs are simple and undirected. The *neighbourhood* of a vertex v in a graph G is the set $N_G(v) = \{w \in V \mid (v, w) \in E\}$, and its *degree* is $\delta_G(v) = |N_G(v)|$. For the sake of simplicity, in the previous notations we drop the subscript when it is clear from the context and simply write $N(v)$, $\delta(v)$, etc. For a subset of nodes $S \subseteq V_G$, we use $\langle S \rangle_G$ to represent the subgraph of G *induced* by S, i.e. $\langle S \rangle_G = (S, E_G \cap (S \times S))$. Similarly, the subgraph of G *weakly induced* by S is defined as $\langle S \rangle_G^w = (S \cup N_G(S), E_G \cap (S \times (S \cup N_G(S))))$. For every graph G and every $S \subseteq V_G$, $\langle S \rangle_G$ is a subset of $\langle S \rangle_G^w$, as $\langle S \rangle_G^w$ additionally contains the neighbourhood of S and every edge between elements of S and their neighbours. Also notice that $\langle S \rangle_G^w$ does not contain the edges linking pairs of elements of $N_G(S)$. An *isomorphism* between two graphs $G = (V, E)$ and $G' = (V', E')$ is a bijective function $\varphi \colon V \to V'$ such that $\forall_{v,w \in V}\ (v, w) \in E \iff (\varphi(v), \varphi(w)) \in E'$. Additionally, we denote by $\varphi(S)$ the restriction of φ to a vertex subset $S \subseteq V$, that is $\varphi(S) = \{\varphi(v) \mid v \in S\}$.

3.2 Overview

Fig. 1 depicts the process of periodical graph publication in the presence of an active adversary. We model this process as a game between two players, the *data owner* and the *adversary*. The data owner selects a set of time-stamps $T = \{t_1, t_2, \ldots, t_i, \ldots\}$, $t_1 < t_2 < \ldots < t_i < \ldots$, and incrementally publishes the sequence $\mathcal{G}^\star = (G^\star_{t_1}, G^\star_{t_2}, \ldots, G^\star_{t_i}, \ldots)$ of sanitised snapshots of the underlying dynamic social graph. The adversary's goal is to re-identify, in a subset $T' \subseteq T$ of the releases, a (possibly evolving) set of legitimate users referred to as the *victims*. To achieve this goal, the active adversary injects an (also evolving) set of fake accounts, commonly called *sybils*, in the graph. The sybil accounts create connections among themselves, and with the victims. The connection patterns between each victim and some of the sybil nodes is used as a unique *fingerprint* for that victim. The likely unique patterns built by the adversary will

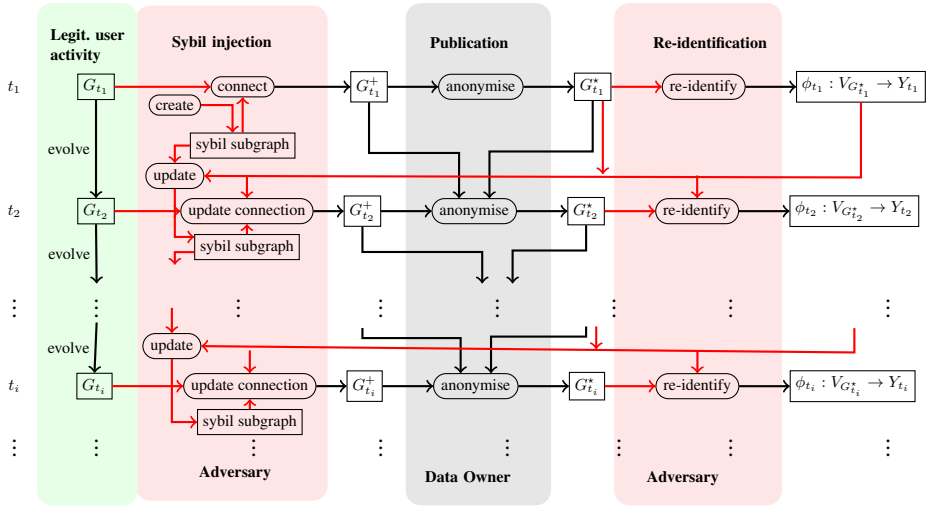

Fig. 1. Overview of periodical graph publication in the presence of active adversaries.

enable her to effectively and efficiently re-identify the victims in the sanitised snapshots. At every re-identification attempt, the adversary first re-identifies the set of sybil nodes, and then uses the fingerprints to re-identify the victims.

The data owner and the adversary have different partial views of the dynamic social graph. On the one hand, the data owner knows the entire set of users, both legitimate users and sybil accounts, but she cannot distinguish them. The data owner also knows all relations. On the other hand, the adversary knows the identity of her victims and the structure of the subgraph weakly induced by the set of sybil nodes, but she does not know the structure of the rest of the network. In this paper we conduct the analysis from the perspective of an external observer who can view all of the information. We will use the sequence $\mathcal{G}^+ = (G^+_{t_1}, G^+_{t_2}, \ldots, G^+_{t_i}, \ldots)$ to denote the view of the network according to the data owner, i.e. the real network containing the nodes representing all users, both legitimate and malicious. Further we use $\mathcal{G} = (G_{t_1}, G_{t_2}, \ldots, G_{t_i}, \ldots)$ to represent the view of the unattacked network, that is the view of the dynamic subgraph induced in $\mathcal{G}^+$ by the nodes representing legitimate users.

In the original formulation of active attacks, a single snapshot of the graph is released, so all actions executed by the sybil nodes are assumed to occur before the publication. This is not the case in the scenario of a periodically released dynamic social graph. Here, the adversary has the opportunity to schedule actions in such a way that the subgraph induced by the sybil nodes evolves, as well as the set of fingerprints. In turn, that allows her to use temporal patterns in addition to structural patterns for re-identification. Additionally, the adversary can target different sets of victims along the publication process and adapt the induced tempo-structural patterns to the evolution of the graph and the additional knowledge acquired in each re-identification attempt. In the new scenario, the actions performed by the adversary and the data owner alternate as follows before, during and after each time-stamp $t_i \in T$.

Before t_i, the adversary may remain inactive, or she can modify the set of sybil nodes, as well as the set of sybil-to-sybil and sybil-to-victim edges. The result of these actions is the graph $G^+_{t_i} = (V_{t_i} \cup S_{t_i}, E_{t_i} \cup E^+_{t_i})$, where V_{t_i} is the current set of legitimate users, S_{t_i} is the current set of sybil nodes, $Y_{t_i} \subseteq V_{t_i}$ is the current set of victims, $E_{t_i} = E_{G_{t_i}} \subseteq V_{t_i} \times V_{t_i}$ is the set of connections between legitimate users, and $E^+_{t_i} \subseteq (S_{t_i} \times S_{t_i}) \cup (S_{t_i} \times Y_{t_i})$ is the set of connections created by the sybil accounts. The subgraph $\langle S_{t_i} \rangle^w_{G^+_{t_i}}$, weakly induced in $G^+_{t_i}$ by the set of sybil nodes, is the *sybil subgraph*. We refer to the set of modifications of the sybil subgraph executed before the adversary has conducted any re-identification attempt as *sybil subgraph creation*. If the adversary has conducted a re-identification attempt on earlier snapshots, we refer to the modifications of the sybil subgraph as *sybil subgraph update*.

During t_i, the data owner applies an anonymisation method to $G^+_{t_i}$ to obtain the sanitised version $G^\star_{t_i}$, which is then released. The anonymisation must preserve the consistency of the pseudonyms. That is, every user must be labelled with the same pseudonym throughout the sequence of snapshots where it appears. Consistent annotation is of paramount importance for a number of analysis tasks such as community evolution analysis [4], link prediction [11], link persistence analysis [17], among others, that require to track users along the sequence of releases. The data owner anonymises every snapshot exactly once.

After t_i, the adversary adds $G^\star_{t_i}$ to her knowledge. At this point, she can remain inactive, or execute a re-identification attempt on $G^\star_{t_i}$. The result of a re-identification attempt is a mapping $\phi_{t_i} \colon V_{G^\star_{t_i}} \to Y_{t_i}$ determining the pseudonyms assigned to the victims by the anonymisation method. The adversary can additionally modify the results of a re-identification attempt executed on some of the preceding releases.

3.3 Temporal Consistency Constraints

As we discussed in Sect. 3.2, the data owner must assign the same time-persistent pseudonym to each user throughout the subsequence of snapshots where it appears. Since the adversary receives all sanitised snapshots, she is able to determine when a pseudonym was used for the first time, whether it is still in use, and in case it is not, when it was used for the last time. In our attack, the adversary exploits this information in all stages of the re-identification process. For example, consider the following situation. The set of sybil nodes at time-step t_6 is $S_{t_6} = \{s_1, s_2, s_3, s_4\}$. The adversary inserted s_1 and s_2 in the interval preceding the publication of $G^\star_{t_2}$. Additionally, she inserted s_3 before the publication of $G^\star_{t_3}$ and s_4 before the publication of $G^\star_{t_5}$. After the release of $G^\star_{t_6}$, during the sybil subgraph retrieval phase of the first re-identification attempt, the adversary needs to determine whether a set $X \subseteq V_{G^\star_{t_6}}$, say $X = \{v_1, v_2, v_3, v_4\}$, is a valid candidate. Looking at the first snapshot where each of these pseudonyms was used, the adversary observes that v_1 and v_3 were first used in $G^\star_{t_2}$, so they are feasible matches for s_1 and s_2, in some order. Likewise, v_2 was first used in $G^\star_{t_5}$, so it is a feasible match for s_4. However, she observes that v_4 was first used in $G^\star_{t_4}$, unlike any element of S_{t_6}. From this observation, the adversary infers that X is not a valid candidate, regardless of how structurally similar $\langle X \rangle^w_{G^\star_{t_6}}$ and $\langle S_{t_6} \rangle^w_{G^+_{t_6}}$ are.

We now formalise the different types of constraints used in our attack. To that end, we introduce some new notation. The function $\alpha^+ \colon \cup_{t_i \in T} V_{G^+_{t_i}} \to T$ yields, for every vertex $v \in \cup_{t_i \in T} V_{G^+_{t_i}}$, the order of the first snapshot where v exists, that is $\alpha^+(v) = \min\{\{t_i \in T \mid v \in V_{G^+_{t_i}}\}\}$. Analogously, the function $\alpha^\star \colon \cup_{t_i \in T} V_{G^\star_{t_i}} \to T$ yields the order of the first snapshot where each pseudonym is used, that is $\alpha^\star(x) = t_i \iff \exists_{v \in V_{G^+_{t_i}}} \alpha^+(v) = t_i \wedge \varphi_{t_i}(v) = x$. Clearly, the adversary knows the values of the function $\alpha^\star$ for all pseudonyms used by the data owner. Additionally, she knows the values of α^+ for all of her sybil nodes. These functions allow us to define the notion of *first-use-as-sybil* consistency, which is used by the sybil subgraph retrieval method.

Definition 1. *Let $X \subseteq V_{G^\star_{t_i}}$ be a set of pseudonyms such that $|X| = |S_{t_i}|$ and let $\phi \colon S_{t_i} \to V_{G^\star_{t_i}}$ be a mapping from the set of real sybil nodes to the elements of X. We say that X and S_{t_i} satisfy* first-use-as-sybil *consistency according to ϕ, denoted as $X \simeq_\phi S_{t_i}$, if and only if $\forall_{s \in S_{t_i}} \alpha^+(s) = \alpha^\star(\phi(s))$.*

Note that *first-use-as-sybil* consistency depends on the order in which the elements of the candidate set are mapped to the real sybil nodes, which is a requirement of the sybil subgraph retrieval method. We define an analogous notion of first use consistency for victims. In this case, the adversary may or may not know the value of α^+. In our attack, we assume that she does not, and introduce an additional function to represent the temporal information the adversary must necessarily have about victims. The function $\beta^+ \colon \cup_{t_i \in T} Y_{t_i} \to T$ yields, for every $v \in \cup_{t_i \in T} Y_{t_i}$, the order of the snapshot where v was targeted for the first time, that is $\beta^+(v) = \min\{\{t_i \in T \mid v \in Y_{t_i}\}\}$. The new function allows us to define the notion of *first-time-targeted* consistency, which is used in the fingerprint matching method.

Definition 2. *Let $v \in V_{G^\star_{t_i}}$ be a victim candidate and let $y \in Y_{t_i}$ be a real victim. We say that v and y satisfy* first-time-targeted *consistency, denoted as $v \simeq y$, if and only if $\alpha^\star(v) \leq \beta^+(y)$.*

This temporal consistency notion encodes the rationale that the adversary can ignore during fingerprint matching those pseudonyms that the data owner used for the first time after the corresponding victim had been targeted. Next, we define the notion of *sybil-removal-count* consistency, which is used by the re-identification refinement method to encode the rationale that a sybil set candidate X, for which no temporal inconsistencies were found during the t_i-th snapshot, can be removed from $\mathcal{X}_{t_i}$ when the t_{i+1}-th snapshot is released, if the number of sybil nodes removed by the adversary in the interval between these snapshots does not match the number of elements of X that cease to exist in $G^\star_{t_{i+1}}$.

Definition 3. *We say that a set of pseudonyms $X \subseteq V_{G^\star_{t_i}}$ satisfies* sybil-removal-count *consistency with respect to the pair $(S_{t_i}, S_{t_{i+1}})$, which we denote as $X \simeq (S_{t_i}, S_{t_{i+1}})$, if and only if $|X \setminus V_{G^\star_{t_{i+1}}}| = |S_{t_i} \setminus S_{t_{i+1}}|$.*

In certain social networks, the adversary can detect when one victim leaves the network, e.g. by detecting that all connections to the victim from her sybil nodes are

simultaneously lost. From this, the adversary infers that the victim's pseudonym will not be present in the next release of the graph. This rationale can be used to further refine the re-identification of victims in previously released snapshots, by discarding those mappings where the pseudonyms continue to be present in the snapshots released after the corresponding victims terminate their membership of the network. To encode this rationale, the function $\gamma^+\colon \cup_{t_i \in T} V_{G^+_{t_i}} \rightarrow T$ yields the order of the last snapshot where a node is present in the social network. Note that the adversary is certain about $\gamma^+(s)$ for every sybil node s. Analogously, the function $\gamma^\star\colon \cup_{ti \in T} V_{G^\star_{t_i}} \rightarrow T$ yields the order of the last snapshot where a pseudonym appears. By comparing the vertex sets of two consecutive snapshots, the adversary can learn $\gamma^\star(v)$ for any pseudonym v.

Definition 4. *Let $v \in V_{G^\star_{t_i}}$ be a victim candidate and let $y \in Y_{t_i}$ be a real victim. We say that v and y satisfy* last-time-targeted *consistency, denoted as $v \simeq_\ell y$, if and only if $\gamma^\star(v) = \gamma^+(y)$.*

In what follows, we discuss how the temporal consistency constraints introduced in Defs. 1 to 4 are used in our new dynamic re-identification attack.

3.4 Stages of the Attacker-Defender Game

We now discuss the actions performed by the data owner and the adversary at every time-stamp t_i. We first discuss sybil subgraph creation and update, then graph publication, and finally re-identification.

Sybil subgraph creation. The adversary can build the initial sybil subgraph along several releases. This allows the creation of tempo-structural patterns incorporating information about the first snapshot where each sybil node appears, to facilitate the sybil subgraph retrieval stage during re-identification. As in all active attacks, the patterns created must ensure that, with high probability, $\langle S_{t_i} \rangle^w_{G^+_{t_i}}$ is unique. We denote by $F_{t_i}(y)$ the fingerprint of a victim $y \in Y_{t_i}$ in terms of S_{t_i}. Throughout this paper we consider that $F_{t_i}(y)$ is uniquely determined by the neighbourhood of y in S_{t_i}, that is $F_{t_i}(y) = S_{t_i} \cap N_{G^+_y}$. We denote by $\mathcal{F}_{t_i}$ the set of fingerprints of all victims in $G^+_{t_i}$.

Sybil subgraph update. In this step, the adversary can modify the set of sybil nodes, by adding new sybil nodes or replacing existing ones. The adversary can also modify the inter-sybil connections and the fingerprints. Sybil subgraph update is executed after at least one re-identification attempt has been conducted, so the adversary can use information from this attempt, such as the level of uncertainty in the re-identification, to decide the changes to introduce in the sybil subgraph. Finally, if the number of fingerprints that can be constructed using the new set of sybil nodes is larger than the previous number of targeted victims, that is $2^{|S_{t_i}|} - 1 > |Y_{t_{i-1}}|$, the adversary can additionally target new victims, either new users that joined the network in the last inter-release interval, or previously enrolled users that had not been targeted so far. In the latter case, even if these victims had not been targeted before, the consistency of the labelling in the sequence of sanitised snapshots entails that a re-identification in the t_i-th snapshot can be traced back to the previous ones. Additional details on the implementation of sybil subgraph creation and update in the instantiation of the dynamic active attack presented in this paper can be found in App. A.

Graph publication. At time step t_i, the data owner anonymises $G^+_{t_i}$ and publishes the sanitised version $G^\star_{t_i}$. We formally view anonymisation as a two-step process. The first step is *pseudonymisation*, which consists in building an isomorphism $\varphi_{t_i}\colon V_{G^+_{t_i}} \rightarrow V'^\star_{t_i}$, with $V'^\star_{t_i} \cap V_{G^+_{t_i}} = \emptyset$, that replaces every real identity in $G^+_{t_i}$ for a pseudonym. The pseudonymised graph is denoted as $\varphi_{t_i}G^+_{t_i}$. If $i = 1$, all pseudonyms are freshly generated. In the remaining cases, the pseudonyms for previously existing vertices are kept, and fresh pseudonyms are assigned to new vertices. The second step of the anonymisation process consists in applying a *perturbation* method $\Phi_{t_i}\colon \varphi_{t_i}G^+_{t_i} \rightarrow (V^\star_{t_i}, V^\star_{t_i} \times V^\star_{t_i})$ to the pseudonymised graph. Perturbation consists in editing the vertex and/or edge sets of the pseudonymised graph. Finally, the data owner releases the graph $G^\star_{t_i}$ obtained as the result of applying pseudonymisation on $G^+_{t_i}$ and perturbation on $\varphi_{t_i}G^+_{t_i}$.

First re-identification attempt. The first re-identification attempt is composed of two steps, *sybil subgraph retrieval* and *fingerprint matching*. Sybil subgraph retrieval consists in the following substeps:

1. Find in $G^\star_{t_i}$ a set $\mathcal{X}_{t_i} = \{X_1, X_2, \ldots, X_p\}$, $X_j \subseteq V_{G^\star_{t_i}}$, of candidate sybil sets. For every $X \in \mathcal{X}_{t_i}$, the graph $\langle X\rangle^w_{G^\star_{t_i}}$ is a candidate sybil subgraph. In the instantiation of the dynamic active attack presented in this paper, we apply two filtering criteria:
   (i) Every element X of $\mathcal{X}_{t_i}$ must satisfy $\Delta(\langle X\rangle^w_{G^\star_{t_i}}, \langle S_{t_i}\rangle^w_{G^+_{t_i}}) \leq \theta_{t_i}$, where Δ is the structural dissimilarity function defined in [14] and θ_{t_i} is a tolerance threshold. The value of θ_{t_i} may be fixed (as in [14]), or it may be increased as new snapshots are released in order to adapt to the accumulation of modifications in successive instances of the graph publication step.
   (ii) Every element of $\mathcal{X}_{t_i}$ must satisfy the *first-use-as-sybil* consistency constraint with respect to $\langle S_{t_1}\rangle^w_{G^+_{t_1}}, \langle S_{t_2}\rangle^w_{G^+_{t_2}}, \ldots, \langle S_{t_{i-1}}\rangle^w_{G^+_{t_{i-1}}}$.
2. If $\mathcal{X}_{t_i} = \emptyset$, the attack fails. Otherwise, proceed to fingerprint matching (step 2).

For its part, fingerprint matching consists in the following substeps:

1. Select one element $X \in \mathcal{X}_{t_i}$ with probability $\frac{1}{|\mathcal{X}_{t_i}|}$.
2. Using X and $\mathcal{F}_{t_i}$, find a set of candidate mappings $\mathcal{Y}_X = \{\phi_1, \phi_2, \ldots, \phi_q\}$, where every ϕ_j ($1 \leq j \leq q$) has the form $\phi_j : V_{G^\star_{t_i}} \setminus S_{t_i} \rightarrow Y_{t_i}$. Every element of $\mathcal{Y}_{X_{t_i}}$ represents a possible re-identification of the victims in $G^\star_{t_i}$. In the instantiation of the dynamic active attack presented in this paper, $\mathcal{Y}_{X_{t_i}}$ is composed of the elements simultaneously satisfying two criteria:
   (i) Maximise the noise-tolerant fingerprint similarity function defined in [14], provided that the similarity is above a threshold η.
   (ii) Satisfy the *first-time-targeted* and *last-time-targeted* consistency constraints with respect to $\mathcal{F}_{t_1}, \mathcal{F}_{t_2}, \ldots, \mathcal{F}_{t_{i-1}}$.
3. If $\mathcal{Y}_{\mathcal{X}'_{t_i}} = \emptyset$, the attack fails. Otherwise, select one element of $\mathcal{Y}_{\mathcal{X}'_{t_i}}$ and give it as the result of the re-identification. As in the previous steps, every specific attack defines how the selection is made.

The combination of structural similarity and temporal consistency in steps 1.a and 2.b considerably speed-up the overall re-identification process, and increase its effectiveness, as will be empirically demonstrated in Sect. 4. Additional details on the implementation of steps 1.a and 2.b are given in App. B.

Re-identification refinement. As we discussed above, the first re-identification attempt on $G^{\star}_{t_i}$ can be executed immediately after the snapshot is published. Then, after the publication of $G^{\star}_{t_j}$, $j > i$, the re-identification refinement step allows the adversary to improve her certainty on the previous re-identification, by filtering out elements of $\mathcal{X}_{t_i}$ that fail to satisfy the *sybil-removal-count* consistency constraint with respect to $\langle S_{t_i} \rangle^{w}_{G^{+}_{t_i}}$ and $\langle S_{t_j} \rangle^{w}_{G^{+}_{t_j}}$, and then repeating the fingerprint matching step, excluding the candidate nodes that do not comply with the *first-time-targeted* or the *last-time-targeted* consistency constraints.

4 Experiments

In this section, we empirically evaluate our new dynamic active attack. Our evaluation has three goals. First, we show that our dynamic attack outperforms the alternative of repeatedly executing Mauw *et al.*'s static robust active attack [14] in terms of both effectiveness and efficiency. Secondly, we determine the factors that affect the performance of our new attack, and evaluate their impact. From this analysis, we derive a number of recommendations allowing data owners to balance privacy preservation and utility in random perturbation methods for periodical social graph publication. Due to the scarcity of real-life temporally labelled social graphs, and the complete non-existence of datasets of this type where the phenomenon of users abandoning the social network is observed, we conducted these experiments on synthetic dynamic social graphs. To that end, we developed a flexible synthesiser which generates synthetic dynamic graphs with several parameter settings. Finally, we replicate the second experiment on two real-life datasets, to show that some of the findings obtained on synthetic data remain valid in practical scenarios. For simplicity, throughout this section we use the acronym D-AA for our new dynamic attack and S-RAA for the static robust active attack.

4.1 Experimental Setting

We implemented an evaluation tool based on the attacker-defender game described in Sect. 3. A dynamic social graph simulator loads a real-life dataset, or uses the synthesiser, to generate the sequence $\mathcal{G} = (G_{t_1}, G_{t_2}, \ldots, G_{t_i}, \ldots)$ containing only legitimate users. Each snapshot is then processed by a second module that simulates sybil subgraph creation or update. The output, which is the data owner's view of the social graph, is processed by a graph perturbation module, where we implement a simple perturbation method based on cumulative noise addition. Finally, a fourth module simulates the re-identification on the perturbed graph and computes the success probability of the attack. Sybil subgraph creation and update, as well as re-identification, have been discussed in Sect. 3.4 (with extensive details given in Apps. A and B, respectively). We describe in what follows the implementation of the remaining modules.

Dynamic social graph simulator. Our simulator allows us to conduct experiments on temporally annotated real-life datasets, as well as synthetic datasets. In the first case, the simulator extracts the graph snapshots from each dataset using a specific handler. The simulator is parameterised with a sequence of time-stamps indicating when each snapshot should be taken. A snapshot is built by taking all vertices and edges created at a moment earlier or identical to the corresponding time-stamp and still not eliminated.

In the second case, our simulator synthesises a sequence of snapshots according to the Barabási-Albert (BA) generative graph model [1]. We use BA because it preserves the properties of real social graphs, namely power-law degree distribution, shrinking diameter, and preferential attachment. The BA model has two parameters: the number of nodes n_0 of a (small) seed graph, and the initial degree M_e ($M_e \leq n_0$) of every newly added node. The initial seed graph can be any graph. In our case we use a complete graph K_{n_0}. Every time a new node v is added to the current version G of the BA graph, M_e edges are added between v and randomly selected vertices in V_G. The probability of selecting a vertex $w \in V_G$ for creating the new edge (v, w) is $\frac{\delta_G(w)}{\sum_{x \in V_G} \delta_G(x)}$, as prescribed by preferential attachment. We simulate the phenomenon of users abandoning the social network by removing z randomly selected nodes before creating a new snapshot. The value of z is randomly selected in the interval $[0, z^*]$, where z^* is 10% of the current number of nodes. The synthesiser takes four parameters as input: the parameter n_0 of the BA model, the parameter M_e of the BA model, the number n_v of vertices of the first snapshot, and the growth rate r_Δ, which is defined as the proportion of new edges with respect to the previous number. The parameters n_v and r_Δ determine when snapshots are taken. The first snapshot is taken when the number of vertices of the graph generated by the BA model reaches n_v, and every other snapshot is taken when the ratio between the number of new edges and that of the previous snapshot reaches r_Δ.

Graph perturbation via cumulative noise addition. To the best of our knowledge, all existing anonymisation methods against active attacks based on formal privacy properties [12, 13] assume a single release scenario, and are thus insufficient for handling multiple releases. Proposing formal privacy properties that take into account the specificities of the multiple release scenario is part of the future work. In our experiments, we adapted the other known family of perturbation methods, random noise addition, to the multiple release scenario. To account for the incrementality of the publication process, the noise is added in a cumulative manner. That is, when releasing $G^\star_{t_i}$, the noise incrementally added on $G^\star_{t_1}, G^\star_{t_2}, \ldots, G^\star_{t_{i-1}}$ is re-applied on the pseudonymised graph $\varphi_{t_i} G^+_{t_i}$ to obtain an intermediate noisy graph $\tilde{G}^\star_{t_i}$, and then fresh noise is added on $\tilde{G}^\star_{t_i}$ to obtain the graph $G^\star_{t_i}$ that is released. In re-applying the old noise, all noisy edges incident in a vertex $v \in V_{G^\star_{t_{i-1}}} \setminus V_{\varphi_{t_i} G^+_{t_i}}$, removed after the release of $G^\star_{t_{i-1}}$, are forgotten. The fresh noise addition consists in randomly flipping a number of edges of $\tilde{G}^\star_{t_i}$. For every flip, a pair $(v, w) \in V_{\tilde{G}^\star_{t_i}} \times V_{\tilde{G}^\star_{t_i}}$ is uniformly selected and, if $(v, w) \in E_{\tilde{G}^\star_{t_i}}$, the edge is removed, otherwise it is added. The cumulative noise addition method has one parameter: the amount of fresh noise to add in each snapshot, called *noise ratio* and denoted Ω_{noise}. It is computed with respect to $|E_{\tilde{G}^\star_{t_i}}|$, the number of edges of the pseudonymised graph after restoring the accumulated noise.

Success probability. As in previous works on active attacks for the single release scenario [12–14], we evaluate the adversary's success in terms of the probability that she correctly re-identifies all victims, which in our scenario is computed by the following formula for the t_i-th snapshot:

$$Pr_{succ}^{(t_i)} = \begin{cases} \frac{\sum_{X \in \mathcal{X}_{t_i}} p_X^{(t_i)}}{|\mathcal{X}_{t_i}|} & \text{if } \mathcal{X}_{t_i} \neq \emptyset \\ 0 & \text{otherwise} \end{cases}, \text{ with } p_X^{(t_i)} = \begin{cases} \frac{1}{|\mathcal{Y}_X|} & \text{if } \exists_{\phi \in \mathcal{Y}_X} \ \phi^{-1} = \varphi_{t_i}|_{Y_{t_i}} \\ 0 & \text{otherwise} \end{cases}$$

and, as discussed in Sect. 3.4, φ_{t_i} is the isomorphism applied on $G_{t_i}^+$ to obtain the pseudonymised graph $\varphi_{t_i} G_{t_i}^+$. For every snapshot $G_{t_i}^\star$, we compute success probability after the re-identification refinement is executed.

4.2 Results and Discussion

We begin our discussion with the comparison of D-AA and S-RAA. Then, we proceed to study the factors that affect the effectiveness of our attack, and characterise their influence. Finally, we illustrate the effectiveness of our attack in practice using the real-life datasets Petster [9] and BitcoinOTC [8]. For the first two sets of results, we use synthetic dynamic graphs generated by our synthesiser. Table 1 summarises the different configurations used for the generation. For each parameter combination, we generated 100 synthetic dynamic graphs, and the results shown are the averages over each subcollection. Every synthetic dynamic graph is grown up to the 20-th snaphsot. In all cases, the number of new victims targeted in each new release is randomly chosen in the interval $[1, 5]$.

	$\mathbf{n_0}$	$\mathbf{M_e}$	$\mathbf{n_v}$	$\mathbf{r_\Delta}$	$\mathbf{\Omega}_{noise}(\%)$
Comparison of D-AA and S-RAA	30	5	200, 400, 800	5%	0.5
Detailed analysis of our D-AA attack	30	5, 10	8000, 10000, 15000	5%	0.5, 1.0, 1.5, 2.0

Table 1. Parameter combinations for the graph synthesiser.

As can be observed in the table, we used considerable smaller graphs for comparing D-AA and S-RAA than the ones used for the detailed analysis of the factors influencing the effectiveness of our new attack. The reason for this difference lies in the considerably poorer performance, in terms of execution time, of the static attack. Since these limits only apply to the static attack, the detailed analysis of our dynamic attack is performed on considerably larger graphs. For example, for $M_e = 5$ and $n_v = 15000$, the graphs generated at the 20-th snapshots have around 80000 nodes.

Comparing D-AA and S-RAA. The goal of this comparison is to show that our dynamic active attack outperforms the repeated execution of the original attack in both effectiveness and efficiency. We use three parameter settings for the dynamic graph synthesiser. We fix $M_e = 5$ and set the initial number of vertices n_v at 200, 400 and 800. In all our experiments, sybil subgraph creation spans the first and second snapshots, and the re-identification is executed for the first time on the second snapshot. Both attacks

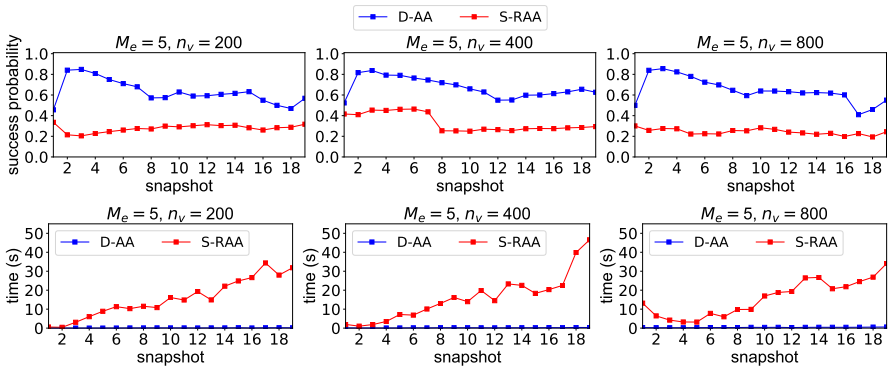

Fig. 2. Comparison of S-RAA and D-AA in terms of effectiveness (top) and efficiency (bottom).

are executed independently. That is, for every run we create two identical copies of each synthetic graph to ensure that both attacks are compared in the same scenario, yet the actions performed by D-AA have no impact on S-RAA, and *vice versa*. S-RAA is allowed to create a fresh sybil subgraph for every snapshot and to increase its number of sybils if D-AA increases hers. In D-AA, we use the variable tolerance threshold $\theta_{t_i} = \min\{1500, 16 + 250 \times (i-2)^2\}$ for the i-th snapshot. Since S-RAA becomes prohibitively slow with arbitrarily large values of θ, we run it with the fixed tolerance threshold $\theta = 16$, for which the attack runs in reasonable time for the largest graphs used in this comparison. These settings guarantee that, to the largest possible extent, D-AA is compared to the most effective feasible instantiation of S-RAA.

In Fig. 2 (top) we show the success probabilities of the two attacks on graphs with different initial sizes. From these results, we can see that, as we had intuitively foreseen, D-AA significantly outperforms S-RAA in terms of success probability. Except for a few cases, D-AA outperforms S-RAA by at least a factor of 2 and by up to 4 in some cases. Moreover, the average success probability of our attack remains above 0.5 in almost all cases, whereas that of S-RAA never reaches this value. Fig. 2 (bottom) shows the average run times of S-RAA and D-AA in different scenarios. We can see that D-AA runs in almost constant time on all snapshots, whereas S-RAA becomes considerably slower as the graphs grow. In fact, D-AA runs at least 11 times faster than S-RAA in all cases, especially in late snapshots, where it runs up to 350 times faster in some cases. This clearly shows that the use of temporal information in dynamic social graphs helps D-AA to effectively avoid the computation overhead. Indeed, as the released snapshots become larger, the number of equally similar matches (in terms of structure alone) grows considerably, which dramatically increases the search space for S-RAA. In this scenario, temporal consistency constraints allow D-AA to discard most of the false positives and thus skip large areas of the search space.

Factors influencing our attack. This analysis aims to serve as a guide for customising the settings of privacy-preserving publication methods for dynamic social graphs, in particular for determining the amount of perturbation needed to balance the privacy requirements and the utility of published graphs. We evaluate the impact of three factors

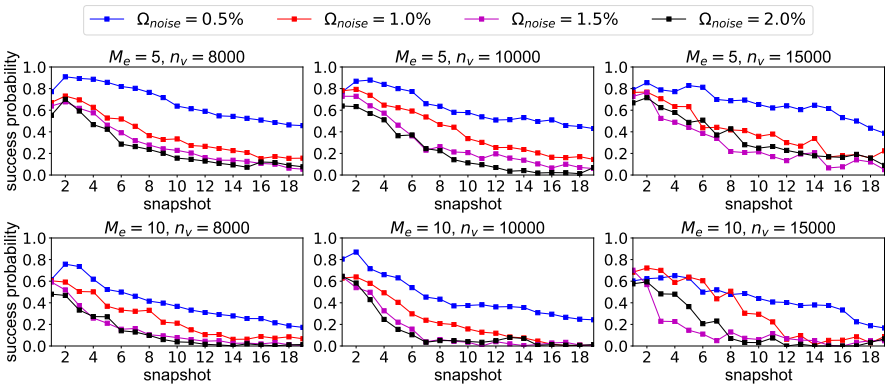

Fig. 3. Factors influencing success probability of D-AA.

on the effectiveness of D-AA on dynamic graphs: the size of graphs, the speed of growth between releases, and the amount of added noise. To that end, we analyse three parameters which determine these factors in our simulator: n_v, M_e, and Ω_{noise}. The number n_v of vertices in the initial snapshots determines the scale of the released graphs, while the parameter M_e of the BA model controls the number of new nodes and edges added before the next release. Finally, Ω_{noise} determines the amount of added noise.

Fig. 3 shows the success probability of our attack when different noise ratios are applied on dynamic graphs with different initial sizes and growth speeds. Our first observation is that the success probability decreases when more noise is applied. This is a natural behaviour, as more perturbation makes it more difficult for the attacker to find the correct sybil subgraph, either because it has been excessively perturbed to be found as a candidate, or because edge perturbation makes other subgraphs appear more similar to the original sybil subgraph. When $M_e = 5$ and the noise ratio is set to 0.5%, success probability always remains above 0.5. For this value of M_e, even with Ω_{noise} at 2.0%, the attack still displays success probability above 0.5 in the first three snapshots. Our second observation is that increases in noise ratio do not translate into proportionally large decreases in success probability. Indeed, the largest drop in success probability occurs when we increase Ω_{noise} from 0.5 to 1.0. This suggests that arbitrarily increasing the amount of perturbation may not necessarily guarantee a better privacy protection, but just damage the utility of the released graphs. Our third observation is that success probability values show a weak dependence on the initial size of the graphs, with other parameters fixed. Finally, we observe that success probability decreases faster, and is around 10% lower, snapshot-by-snapshot, when dynamic graphs grow faster (in this case, when M_e grows). Summing up, we observe that re-identification risk decreases when more perturbation is applied, or when the graphs grow faster, whereas the initial size of the graphs has a relatively small impact on the attacker's success probability.

We evaluate the utility of released graphs in terms of three measures: the percentage of edge editions, the variation of the average local clustering coefficient, and the Kullback-Leibler divergence of degree distributions. As all three measures present very similar patterns for different values of n_v, we only show the results for $n_v = 15000$. We

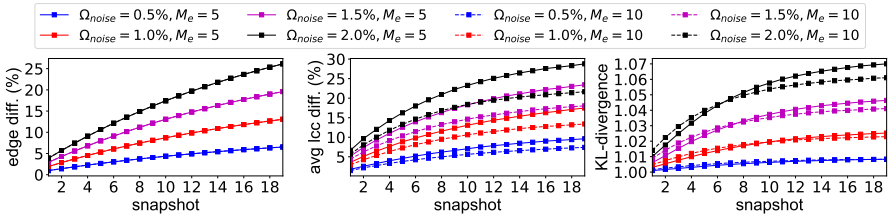

Fig. 4. Factors influencing the utility of released graphs.

have two major observations. First, as expected, the values of all three measures increase as the noise accumulates, indicating that the utility of released graphs deteriorates. Even with Ω_{noise} set to just 1.0%, at the 10-th snapshot we can have up to 10% of edges flipped and changes in edge density around 15%. Second, when the dynamic graph grows faster, the impact of noise becomes smaller, as a larger number of legitimate edges offsets the impact of noisy edges. Combining these results on utility with the finding that larger growth speed results in smaller success probability for the attacker, we can enunciate the following global recommendation for the design of publication strategies and anonymisation methods for dynamic social graphs: the data owner should publish dynamic social graphs that grow fast among releases, as they feature the best balance between re-identification risk and utility.

Results on real-life dynamic social graphs. We use two publicly available datasets to validate to what extent the results reported on synthetic data remain valid in a more realistic domain. The first one was collected from Petster, a website for pet owners to communicate [9]. The Petster dataset is an undirected graph whose vertices represent pet owners, and are labelled by their joining date. The graph contains 1898 vertices and 16750 edges. We take a snapshot every six months. The second dataset was collected from the platform BitcoinOTC, where users can trade with bitcoins. This platform allows members to rate others. In the resulting social graph, nodes represent members and an edge betwen two nodes indicates that one of them rated the other. Every edge is tagged with the date of the first rating between the corresponding pair of users. The joining date of a member is set as the date when his first rating is posted. The graph contains 5881 nodes and 21455 edges. We take the first snapshot at the 9th month, and every other snapshot every 3 months, totalling 20 snapshots. Both datasets are incremental, that is nodes are added but never removed.

We present in Fig. 5 (top) the success probabilities of our D-AA attack on the two datasets when the noise ratio is set to 0.5%, 1.0% and 1.5%. Compared to the success probabilities discussed above on synthetic graphs, the curves have different shapes and more fluctuations. This is because, instead of a fixed growth speed (determined by r_Δ and M_e in our synthesiser), real-life graphs grow at different speeds in different periods, as shown in Fig. 5 (bottom). For example, consider the evolution of Petster. After the first few years of steady growth, it gradually lost its popularity, especially in the last three years, where few new users joined. By cross-checking the attack's behaviour on Petster with the network's evolution, we can see that the success probability changes with the amount of growth before the corresponding release. It first increases

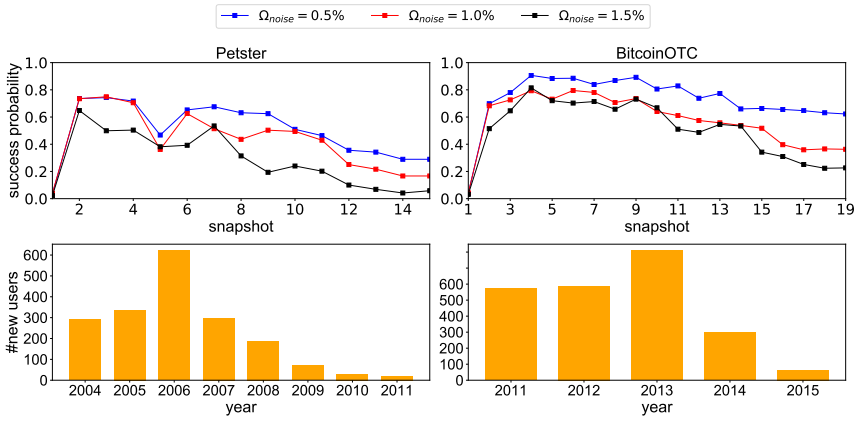

Fig. 5. Evaluation on real-life datasets.

steadily, due to the steady growth of the graph, until the fifth snapshot, which shows an abrupt growth in the number of new vertices, along with a drop of success probability. Then, when the growth slows down, the success probability also recovers; and when the growth stops (from the 12-th snapshot), it starts increasing again, even though the noise continues to accumulate. These observations validate our finding on synthetic graphs that the speed of growth among releases is the dominating factor that affects the success probability of our dynamic attack.

5 Conclusions

We have presented the first dynamic active re-identification attack on periodically released social graphs. Unlike preceding attacks, our new attack exploits the inherent dynamic nature of social networks by leveraging tempo-structural patterns, enforced by a dynamic set of sybil nodes. Compared to the best static active attack, our new attack significantly improves success probability, by at least two times, and efficiency, by at least 11 times. Moreover, unlike the static attack, our new attack remains at the same level of efficiency as the publication process advances. Through comprehensive experiments on synthetic data, we determined the factors that influence the success probability of our new attack against a data owner using cumulative noise addition for graph perturbation, namely the speed of growth and the amount of noise injected. These findings can subsequently be used to develop dynamic graph anonymisation methods that better balance privacy protection and the utility of the released graphs. Additionally, we evaluated our attack on two real-life datasets, which allowed us to ascertain that these findings obtained on synthetic data remain valid in practical scenarios.

Acknowledgements. This work received funding from Luxembourg's Fonds National de la Recherche (FNR), via grant C17/IS/11685812 (PrivDA).

References

1. Albert, R., Barabási, A.L.: Statistical mechanics of complex networks. Review of Modern Physics **74**, 47–97 (2002)
2. Backstrom, L., Dwork, C., Kleinberg, J.M.: Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography. Communications of the ACM **54**(12), 133–141 (2011)
3. Casas-Roma, J., Herrera-Joancomartí, J., Torra, V.: k-degree anonymity and edge selection: improving data utility in large networks. Knowledge and Information Systems **50**(2), 447–474 (2017)
4. Dakiche, N., Tayeb, F.B., Slimani, Y., Benatchba, K.: Tracking community evolution in social networks: A survey. Information Processing & Management **56**(3), 1084–1102 (2019)
5. Ding, X., Zhang, L., Wan, Z., Gu, M.: De-anonymizing dynamic social networks. In: Procs. of GLOBECOM 2011. pp. 1–6 (2011)
6. Ji, S., Li, W., Srivatsa, M., Beyah, R.: Structural data de-anonymization: Quantification, practice, and implications. In: Procs. of CCS 2014. pp. 1040–1053 (2014)
7. Korula, N., Lattanzi, S.: An efficient reconciliation algorithm for social networks. Procs. of the VLDB Endowment **7**(5), 377–388 (2014)
8. Kumar, S., Spezzano, F., Subrahmanian, V.S., Faloutsos, C.: Edge weight prediction in weighted signed networks. In: Procs. of ICDM 2016. pp. 221–230 (2016)
9. Kunegis, J.: KONECT: the Koblenz network collection. In: Procs. of WWW 2013. pp. 1343–1350 (2013)
10. Liu, K., Terzi, E.: Towards identity anonymization on graphs. In: Procs. of SIGMOD 2008. pp. 93–106 (2008)
11. Martínez, V., Berzal, F., Cubero, J.C.: A survey of link prediction in complex networks. ACM Computing Surveys **49**(4), 69 (2017)
12. Mauw, S., Ramírez-Cruz, Y., Trujillo-Rasua, R.: Anonymising social graphs in the presence of active attackers. Transactions on Data Privacy **11**(2), 169–198 (2018)
13. Mauw, S., Ramírez-Cruz, Y., Trujillo-Rasua, R.: Conditional adjacency anonymity in social graphs under active attacks. Knowledge and Information Systems **61**(1), 485–511 (2018)
14. Mauw, S., Ramírez-Cruz, Y., Trujillo-Rasua, R.: Robust active attacks on social graphs. Data Mining and Knowledge Discovery **33**(5), 1357–1392 (2019)
15. Narayanan, A., Shmatikov, V.: De-anonymizing social networks. In: Procs. of S&P 2009. pp. 173–187 (2009)
16. Nilizadeh, S., Kapadia, A., Ahn, Y.: Community-enhanced de-anonymization of online social networks. In: Procs. of CCS 2014. pp. 537–548 (2014)
17. Papadopoulos, F., Kleineberg, K.K.: Link persistence and conditional distances in multiplex networks. Physical Review E **99**(1), 012322 (2019)
18. Pedarsani, P., Figueiredo, D.R., Grossglauser, M.: A bayesian method for matching two similar graphs without seeds. In: Procs. of the 51st Annual Allerton Conf. on Communication, Control, and Computing. pp. 1598–1607 (2013)
19. Peng, W., Li, F., Zou, X., Wu, J.: A two-stage deanonymization attack against anonymized social networks. IEEE Transactions on Computers **63**(2), 290–303 (2014)
20. Rousseau, F., Casas-Roma, J., Vazirgiannis, M.: Community-preserving anonymization of graphs. Knowledge and Information Systems **54**(2), 315–343 (2017)
21. Tai, C.H., Tseng, P.J., Philip, S.Y., Chen, M.S.: Identities anonymization in dynamic social networks. In: Procs. of ICDM 2011. pp. 1224–1229 (2011)
22. Wu, W., Xiao, Y., Wang, W., He, Z., Wang, Z.: K-symmetry model for identity anonymization in social networks. In: Procs. of the 13th Int'l Conf. on Extending Database Technology. pp. 111–122 (2010)

23. Yartseva, L., Grossglauser, M.: On the performance of percolation graph matching. In: Procs. of COSN 2013. pp. 119–130 (2013)
24. Yu, H., Gibbons, P.B., Kaminsky, M., Xiao, F.: Sybillimit: A near-optimal social network defense against sybil attacks. In: Procs. of S&P 2008. pp. 3–17 (2008)
25. Yu, H., Kaminsky, M., Gibbons, P.B., Flaxman, A.: Sybilguard: defending against sybil attacks via social networks. In: Procs. of SIGCOMM 2006. pp. 267–278 (2006)
26. Zhou, B., Pei, J.: Preserving privacy in social networks against neighborhood attacks. In: Procs. of ICDE 2008. pp. 506–515 (2008)
27. Zou, L., Chen, L., Özsu, M.T.: K-automorphism: A general framework for privacy preserving network publication. Procs. of the VLDB Endowment **2**(1), 946–957 (2009)

A Implementation Details of Sybil Subgraph Creation and Update

Sybil subgraph creation. Let $G^{\star}_{t_i}$ be the first snapshot where the adversary conducts a re-identification attempt. Sybil subgraph creation is executed during the entire time window preceding t_i. The adversary initially inserts a small number of sybil nodes, no more than $\left\lfloor \log_2 \left(|V_{G^{+}_{t_i}}| \right) \right\rfloor$. This makes the sybil subgraph very unlikely to be detected by sybil defences [25, 24, 2, 13, 14], while allowing to create unique fingerprints for a reasonably large number of potential initial victims. Spreading sybil injection over several snapshots helps create temporal patterns that reduce the search space during sybil subgraph retrieval. Inter-sybil edges are created in a manner that has been shown in [2] to make the sybil subgraph unique with high probability. First, an arbitrary (but fixed) order is established among the sybil nodes. In our case, we simply take the order in which the sybils are created. Let $s_1 \prec s_2 \prec \ldots \prec s_{|S_{t_i}|}$ represent the order established among the sybils. Then, the edges (s_1, s_2), (s_2, s_3), $\ldots$, $(s_{|S_{t_i}|-1}, s_{|S_{t_i}|})$ are added to force the existence of the path $s_1 s_2 \ldots s_{|S_{t_i}|}$. Additionally, every other edge (s_j, s_k), $|j-k| \geq 2$, is added with probability 0.5. The initial fingerprints of the elements of Y_{t_i} are randomly generated (yet enforcing that all fingerprints are unique) by connecting each victim to each sybil node with probability 0.5.

Sybil subgraph update. Let $G^{\star}_{t_{i-1}}$ and $G^{\star}_{t_i}$ be two consecutive releases occurring after the first snapshot where the adversary conducted a re-identification attempt ($G^{\star}_{t_{i-1}}$ itself may have been this snapshot). In the interval between $G^{\star}_{t_{i-1}}$ and $G^{\star}_{t_i}$, the adversary updates the sybil subgraph by adding and/or removing sybil nodes and inter-sybil edges, updating the fingerprints of (a subset of) the victims, and possibly targeting new victims. We describe each of these modifications in detail in what follows.

Adding and replacing sybil nodes. In our attack, the adversary is conservative regarding the number of sybil nodes, balancing the capacity to target more victims with the need to keep the likelihood of being detected by sybil defences sufficiently low. Thus, the number of sybil nodes is increased as the number of nodes in the graph grows, but keeping $|S_{t_i}| \leq \left\lfloor \log_2 \left(|V_{G^{\star}_{t_{i-1}}}| \right) \right\rfloor$. Additionally, the attacker may select a small random number of existing sybil nodes and replace them for fresh sybil nodes.

Let $S_{t_{i-1}} = \{s_1, s_2, \ldots, s_{|S_{t_{i-1}}|}\}$ be the set of sybil nodes present in $G^{+}_{t_{i-1}}$, and let $s_1 \prec s_2 \prec \ldots \prec s_{|S_{t_{i-1}}|}$ be the order established among them. We first consider the case of sybil node addition. Let $S' = \{s'_1, s'_2, \ldots, s_q\}$ be the set of new sybil nodes that will be added to $G^{+}_{t_i}$, and let $s'_1 \prec s'_2 \prec \ldots \prec s_q$ be the order established on

them. The path $s_1 s_2 \dots s_{|S_{t_{i-1}}|}$ is extended into $s_1 s_2 \dots s_{|S_{t_{i-1}}|} s'_1 s'_2 \dots s_q$ by adding to $G^+_{t_i}$ the edges $(s_{|S_{t_{i-1}}|}, s'_1), (s'_1, s'_2), \dots, (s'_{q-1}, s'_q)$. Additionally, the adversary adds to $G^+_{t_i}$ every node (x, y), $x \in S'$, $y \in (S_{t_{i-1}} \cup S') \setminus N_{G^+_{t_i}}(x)$, with probability 0.5. In order to replace a sybil node $s_j \in S_{t_{i-1}}$ for a new sybil node s ($s \notin S'$), the adversary adds to $G^+_{t_i}$ the edges (s_{j-1}, s) and (s, s_{j+1}), where s_{j-1} and s_{j+1} are the sybil nodes immediately preceding and succeeding s_j according to $\prec$. The order $\prec$ is updated accordingly to make $s_1 \prec s_2 \prec \dots \prec s_{j-1} \prec s \prec s_{j+1} \prec \dots \prec s_{|S_{i-1}|}$. These modifications ensure that the path $s_1 s_2 \dots s_{|S_{t_{i-1}}|}$ guaranteed to exist in $G^+_{t_{i-1}}$ is replaced in $G^+_{t_i}$ for $s_1 s_2 \dots s_{j-1} s s_{j+1} \dots s_{|S_{i-1}|}$. Additionally, the new sybil node s is connected to every other sybil node with probability 0.5. In our attack, every sybil node removal is part of a replacement, so the number of sybil nodes never decreases.

Updating fingerprints of existing victims. After replacing a sybil node $s \in S_{t_{i-1}}$ for a new sybil node $s' \in S_{t_i} \setminus S_{t_{i-1}}$, the adversary adds to $G^+_{t_i}$ the edge (s', y) for every $y \in Y_{t_{i-1}} \cap N_{G^+_{t_{i-1}}}(s)$, to guarantee that the replacement of s for s' does not render any pair of fingerprints identical in $G^+_{t_i}$. Additionally, if new sybil nodes were added, the fingerprints of all previously targeted victims in $Y_{t_{i-1}}$ are modified by creating edges linking them to a subset of the new sybil nodes. For each new sybil node $s \in S_{t_i} \setminus S_{t_{i-1}}$ and every victim $y \in Y_{t_{i-1}}$, the edge (s, y) is added with probability 0.5. Finally, if the adversary has conducted a re-identification attempt on $G^\star_{t_{i-1}}$, she makes additional changes in the set $\mathcal{F}_{t_i}$ of fingerprints in $G^+_{t_i}$ based on the outcomes of the re-identification. To that end, she selects a subset $Y'_{t_{i-1}}$ of victims whose fingerprints were the least useful during the re-identification attempt, in the sense that they were the most likely to lead to a larger number of equally likely options after fingerprint mapping. The adversary modifies the fingerprint of every $y \in Y'_{t_{i-1}}$ by randomly flipping one edge of the form (y, s), $s \in S_{t_{i-1}}$, checking that the new fingerprint does not coincide with a previously existing fingerprint. The set $Y'_{t_{i-1}}$ is obtained as follows. For every victim $y_j \in Y_{t_{i-1}}$ and every vertex v mapped to y_j according to some $X \in \mathcal{X}_{t_{i-1}}$ and the corresponding $\mathcal{Y}_X$, let $p_j(v)$ be the probability that v has been mapped to y_j in the previous re-identification attempt according to some sybil subgraph candidate and some of the resulting fingerprint matchings. We make $Y'_{t_{i-1}} = \arg\max_{y_j \in Y_{t_{i-1}}} \{H(p_j)\}$, where $H(p_j)$ is the entropy of the distribution p_j.

B Implementation Details of Dynamic Re-identification

Sybil subgraph retrieval. The sybil subgraph retrieval method is a breadth-first search procedure, which shares the rationale of analogous methods devised for active attacks on static graphs [2, 14], but differs from them in the use of temporal consistency constraints for pruning the search space. To establish the order in which the search space is traversed, our method relies on the existence of an arbitrary (but fixed) total order $\prec$ among the set of sybil nodes, which is enforced by the sybil subgraph creation method and maintained by the sybil subgraph update method.

Let $s_1 \prec s_2 \prec \dots \prec s_{|S_{t_i}|}$ be the order established on the elements of S_{t_i}. The search procedure first builds a set of cardinality-1 partial candidates $\mathcal{X}_{t_i,1} = \{\{v_{j_1}\} \mid$

$v_{j_1} \in V_{G^\star_{t_i}}\}$. Then, it obtains the pruned set of candidates $\mathcal{X}'_{t_i,1}$ by removing from $\mathcal{X}_{t_i,1}$ all elements $\{v_{j_1}\}$ such that $\alpha^\star(v_{j_1}) \neq \alpha^+(s_1)$, or $\left|\delta_{G^\star_{t_i}}(v_{j_1}) - \delta_{G^+_{t_i}}(s_1)\right| > \theta$. The first condition verifies that the *first-use-as-sybil* consistency property $\{v_{j_1}\} \simeq_\phi \{s_1\}$ holds, with $\phi = \{(s_1, v_{j_1})\}$. The second condition excludes from the search tree all candidates X such that $\Delta(\langle X\rangle^w_{G^\star_{t_i}}, \langle S_{t_i}\rangle^w_{G^+_{t_i}}) > \theta$, where Δ is a structural dissimilarity function, defined in [14], and θ is a tolerance threshold. Then, for every $\ell \leq |S_{t_i}|$, the method builds the set of partial candidates

$$\mathcal{X}_{t_i,\ell} = \{\{v_{j_1}, \ldots, v_{j_\ell}\} \mid \{v_{j_1} \ldots, v_{j_{\ell-1}}\} \in \mathcal{X}_{t_i,\ell-1}, v_{j_\ell} \in V_{G^\star_{t_i}} \setminus \{v_{j_1}, \ldots, v_{j_{\ell-1}}\}\}$$

and obtains the pruned candidate set $\mathcal{X}'_{t_i,\ell}$ by removing from $\mathcal{X}_{t_i,\ell}$ all elements $\{v_{j_1}, \ldots, v_{j_\ell}\}$ such that $\{v_{j_1}, \ldots, v_{j_\ell}\} \not\simeq_\phi \{s_1, \ldots, s_\ell\}$, with $\phi = \{(s_1, v_{j_1}), \ldots, (s_\ell, v_{j_\ell})\}$, and $\Delta(\langle\{v_{j_1}, \ldots, v_{j_\ell}\}\rangle^w_{G^\star_{t_i}}, \langle\{s_1, \ldots, s_\ell\}\rangle^w_{G^+_{t_i}}) > \theta$. Finally, the method gives as output the pruned set of cardinality-$|S_{t_i}|$ candidates, that is $\mathcal{X}_{t_i} = \mathcal{X}'_{t_i,|S_{t_i}|}$. Summing up, our method outputs the set of temporally consistent vertex subsets whose weakly induced subgraphs in $G^\star_{t_i}$ are structurally similar to that of the original set of sybil nodes in $G^+_{t_i}$.

Fingerprint matching. The fingerprint matching step is conducted for a sybil subset candidate $X = \{v_{j_1}, v_{j_2}, \ldots v_{j_{|S_{t_i}|}}\}$ randomly selected from $\mathcal{X}_{t_i}$, with probability $\frac{1}{|\mathcal{X}_{t_i}|}$. Let $v_{j_1} \prec v_{j_2} \prec \ldots \prec v_{j_{|S_{t_i}|}}$ be the order established on the elements of X by the sybil subgraph retrieval method. Our fingerprint matching method is a depth-first search procedure, which gives as output a set $\mathcal{Y}_X = \{\phi_1, \phi_2, \ldots, \phi_q\}$, where every $\phi \in \mathcal{Y}_X$ has the form $\phi : Y_{t_i} \to N_{G^\star_{t_i}}(X)$. Every element of $\mathcal{Y}_X$ maximises the pairwise similarities between the original fingerprints of the victims and the fingerprints, with respect to X, of the corresponding pseudonymised vertices. The method first finds all equally best matches between the (real) fingerprint F_j of a victim $y_j \in Y_{t_i}$ and that of a temporally consistent vertex $u \in N_{G^\star_{t_i}}(X)$ with respect to X, that is $F^\star_u = N_{G^\star_{t_i}}(u) \cap X$. Then, for every such match, it recursively applies the search procedure to match the remaining real victims to other temporally consistent candidate victims. For every victim y_j and every candidate match u, the similarity function $\mathrm{sim}(F^\star_u, F_j)$ integrates the verification of the temporal consistency and the structural fingerprint, and is computed as

$$\mathrm{sim}(F^\star_u, F_j) = \begin{cases} \mathrm{sim}_c(F^\star_u, F_j) & \text{if } u \simeq y_j \text{ and } \mathrm{sim}_c(F^\star_u, F_j) \geq \eta \\ 0 & \text{otherwise.} \end{cases}$$

where η is a tolerance threshold allowing to ignore insufficiently similar matches and the function $\mathrm{sim}_c(F^\star_u, F_j)$ is defined as $\mathrm{sim}_c(F^\star_u, F_j) = \sum_{k=1}^{|S_{t_i}|} \mu_k(F^\star_u, F_j)$ with

$$\mu_k(F^\star_u, F_j) = \begin{cases} 1 & \text{if } v_{j_k} \in F^\star_u \text{ and } s_k \in F_j \\ 0 & \text{otherwise.} \end{cases}$$