Faculty of Science, Technology and Communication

Active re-identification attacks on periodically released social graphs

Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master in Information and Computer Sciences

Author:
Ema Këpuska

Student ID:
0170130809

Supervisor:
Prof. Dr. Sjouke Mauw

Reviewer:
Prof. Dr. Martin Theobald

Advisors:
Dr. Yunior Ramirez Cruz
Dr. Xihui Chen

August 2019

Abstract

Online social networks are one of the most popular online services of our time. In consequence, social network owners are now in possession of an enormous volume of information in the form of social graphs, i.e. graphs where the vertices represent users and the edges represent relations. The analysis of social graphs provides useful information for social scientists or other analysts who benefit from this data to extract global statistics and conduct behavioral studies. However, the sensitive nature of the personal information contained in the network (for example, the existence of relations between users, religious beliefs, etc.) raises important privacy concerns that need to be addressed before the information can be released.

In this thesis, we study a family of privacy attacks on published social graphs, which are called *active re-identification attacks*. A re-identification attack allows to map the nodes of the published graph to the real users they represent. That allows the attacker to obtain potentially private information such as the existence of relations, community co-affiliation, etc. In particular, an *active attacker* is one with the capacity to enrol fake users (or *sybils*) in the network. The sybils interact with the targeted victims and create unique connection fingerprints which are later used to re-identify the victims in the published graph.

So far, the threat posed by active adversaries has been studied in the scenario where one static view of the social graph is published. However, analysts need dynamic social graphs to conduct many tasks, e.g. analyzing the speed at which rumors or fake news spread. Consequently, the privacy risks in dynamic graph publication need to be better understood. This thesis is a step in this direction. The core of our study is the extension of the active attack model to the scenario where snapshots of a dynamic social graph are periodically published. Under the extended adversary model, we propose several new attack strategies allowing the adversary to re-identify users along the sequence of snapshots, increase the number of targeted victims over time, and use information from one snapshot to correct re-identification errors in the previous ones. We experimentally show the increased capabilities of active attackers in the new scenario. As an auxiliary tool for these experiments, we also present a library of periodically released dynamic graph simulators, which can be easily extended with new network growth models, attacks, and perturbation mechanisms. The new attacks introduced in this work, as well as the experimental setting developed within it, will serve as an evaluation environment on which new privacy-preserving periodical graph publication methods will be designed and their effectiveness against active attackers will be tested.

Declaration of Honor

I hereby declare on my honor that I am the sole author of the submitted thesis. The conducted work is original and the result of my own investigations.

I only used those resources that are referenced in the work. All formulations and concepts adopted literally or in their essential content from printed, unprinted or Internet sources have been cited according to the rules for academic work and identified by means of footnotes or other precise indications of source. This thesis has not been presented to any other examination authority. The work is submitted in printed and electronic form.

Luxembourg, August 2019.

Ema KËPUSKA

Acknowledgments

I would first like to thank my thesis supervisor, Prof. Dr. Sjouke Mauw, who advised me during these 6 months and gave me valuable feedback. I am also grateful to my advisor, Dr. Yunior Ramirez Cruz, for his patience, motivation and immense knowledge. His guidance helped me in with research and the writing of this thesis. He always was positive and helpful when I struggled and he managed to guide me in the right direction. As well I would like to show my appreciation to my second advisor Dr. Xihui Chen, who gave me useful feedback throughout this whole experience, he also helped me improving the source-code. I really appreciate their efforts and time spent.

Further, the experiments presented in this paper were carried out using the HPC facilities of the University of Luxembourg [29]– see `https://hpc.uni.lu`, it has been very useful to improve our algorithm.

Additionally, during this research and my studies, all my colleagues were very friendly and supportive. I could not forget them, I really appreciate their support and their help so many times. Next, my friends outside the University, many of them very far. All of this was very encouraging. All my gratitude to everyone who followed and supported me during this journey.

Finally, I must express my very profound gratitude to my parents and to my friends for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them. Thank you.

Luxembourg, August 2019.

Ema KËPUSKA

Contents

1 Introduction **1**
- 1.1 Context . . . 1
- 1.2 Motivation . . . 3
- 1.3 Thesis Contribution . . . 5
- 1.4 Outline . . . 6

2 State of the Art **7**
- 2.1 Privacy attacks on social networks . . . 7
- 2.2 Sybil Defense Mechanism . . . 9
- 2.3 Anonymization . . . 10
- 2.4 Evolution of graphs . . . 10

3 Preliminaries **12**
- 3.1 Notation . . . 12
- 3.2 Active attack on single-snapshot graph . . . 13
  - 3.2.1 The original attacker model . . . 13
  - 3.2.2 Description of the adversary model . . . 15
  - 3.2.3 Variants of active attacks . . . 17
  - 3.2.4 Summary of preliminaries . . . 19

4 Our extended adversarial model and new attack strategies **20**
- 4.1 Description of the extended adversary model . . . 20
  - 4.1.1 Extending the Attacker-defender game to periodically released social graph scenario . . . 21
  - 4.1.2 The extended game . . . 22
  - 4.1.3 Strengthening the adversary capabilities . . . 27

5 Experiments **32**
- 5.1 Periodically released dynamic graph simulators . . . 32
  - 5.1.1 Network Growth . . . 32
  - 5.1.2 Simulating attacker actions . . . 33
  - 5.1.3 Simulating defender actions . . . 35
- 5.2 Datasets and experimental settings . . . 35
- 5.3 Results and Discussion . . . 38
  - 5.3.1 Success probability of all the attacks on periodically released graphs of order n . . . 38
  - 5.3.2 Summary of Results . . . 50

6 Conclusion and Perspectives **54**
6.1 Conclusion 54
6.2 Source-code link 56
6.3 Additional experiments on graphs of 100 nodes with 1% noise and 10 % growth of the graph 56

List of Figures

1.1 Social Network Users Worldwide, 2010-2021 2
1.2 Worldwide social networks as of October 2018, ranked by active users in millions 3

3.1 The original attacker-defender game 14

4.1 Attacker-defender perspective on the graph 21
4.2 Dynamic game between the attacker and defender 23
4.3 Detailed process of the attack in 1^{st} snapshot 24
4.4 Detailed process of the attack in 2^{nd} snapshot 26

5.1 The average probability for all attacks without intersection of 1000 runs on graphs with 100 nodes, 4 edges per node, 0.5% noise and 5% evolve of the graph 39
5.2 The average probability for all attacks with intersection of 1000 runs on graphs with 100 nodes, 4 edges per node, 0.5% noise and 5% evolve of the graph 40
5.3 The average probability for all attacks without intersection of 1000 runs on graphs with 100 nodes, 12 edges per node, 0.5% noise and 5% evolve of the graph 42
5.4 The average probability for all attacks with intersection for 1000 runs on graphs with 100 nodes, 12 edges per node, 0.5% noise and 5% evolve of the graph 43
5.5 The average probability for all attacks without intersection of 1000 runs on graphs with 100 nodes, 20 edges per node, 0.5% noise and 5% evolve of the graph 45
5.6 The average probability for all attacks with intersection of 1000 runs on graphs with 100 nodes, 20 edges per node, 0.5% noise and 5% evolve of the graph 46
5.7 The average probability for all attacks without intersection of 1000 runs on graphs with 100 nodes and 4 edges per node with noise of 1% and 5% growth of the graph for each release 48
5.8 The average probability for all attacks with intersection of 1000 runs on graphs with 100 nodes and 4 edges per node with noise of 1% and 5% growth of the graph for each release 49

5.9 The average probability for all attacks without intersection of 1000 runs on graphs with 100 nodes and 12 edges per node with noise of 1% and 5% growth of the graph for each release 50
5.10 The average probability for all attacks with intersection of 1000 runs on graphs with 100 nodes and 12 edges per node with noise of 1% and 5% growth of the graph for each release 51
5.11 The average probability for all attacks without intersection of 1000 runs on graphs with 100 nodes and 20 edges per node with noise of 1% and 5% growth of the graph for each release 52
5.12 The average probability for all attacks with intersection of 1000 runs on graphs with 100 nodes and 20 edges per node with noise of 1% and 5% growth of the graph for each release 53

6.1 The average probability for all attacks without intersection of 1000 runs on graphs with 100 nodes and 4 edges per node with noise of 1% and 10% growth of the graph for each release 57
6.2 The average probability for all attacks with intersection of 1000 runs on graphs with 100 nodes and 4 edges per node with noise of 1% and 10% growth of the graph for each release 58
6.3 The average probability for all attacks without intersection of 1000 runs on graphs with 100 nodes and 12 edges per node with noise of 1% and 10% growth of the graph for each release 59
6.4 The average probability for all attacks with intersection of 1000 runs on graphs with 100 nodes and 12 edges per node with noise of 1% and 10% growth of the graph for each release 60
6.5 The average probability for all attacks without intersection of 1000 runs on graphs with 100 nodes and 20 edges per node with noise of 1% and 10% growth of the graph for each release 61
6.6 The average probability for all attacks with intersection of 1000 runs on graphs with 100 nodes and 20 edges per node with noise of 1% and 10% growth of the graph for each release 62

List of Tables

5.1 Properties of graphs in the conducted experiments 36
5.2 Attacks used in experiments and their properties 36
5.3 Attack strategies based on the adaptation strategies 37

Chapter 1

Introduction

The growing popularity of social networks has generated an interesting form of communication nowadays, where human interaction and socialization are all done virtually in the social networks. With a touch of one button a person having an account in one of the famous social networks creates a friendship with someone and along the way makes everything public about their life. By posting in the social networks, people tend to express their emotions and also their opinions about their interests. This has changed the way face-to-face communication works, by posting the locations and the pictures and with a single click of a button people express their feelings toward a person. This also allows the people to share whatever they want online, which now expresses a very big problem. Sharing important and personal data has raised an interest for data analysis problems. Furthermore, it has made the social network a target for attackers who can easily affect the social graph by colluding with other users to get as much information or by creating fake accounts and befriending as much people as they can to harvest data from them. Network owners often share the information for research purpose. Research and social graph analysis, although they are very interesting topics, it also comes with a big threat of jeopardizing an individual's privacy. Hence the reason why the data owner uses involves perturbation or sanitization during the data releases process. This process consists of changing the graph structure in some way to makes it harder for the attacks to be successful. The attackers are given access to the entire graph in an anonymized form. Anonymization can be modeled in different ways, where as the simplest one is consists in removing identifying attributes from the social graph, such as name, email address, and social security number [28]. Dealing with such data sanitization makes it harder to identify the user behind the node in a social graph. This attack is known as the re-identification attack.

1.1 Context

A social network is a social structure having nodes and edges as its components. Nodes correspond to people or other social entities, while edges correspond to the social link between them. In a social network people have the ability to reach

to other people and create new connections, message them, publish images, share location etc.

All around the world people use social networks such as Facebook, Instagram, Linked In, Snapchat etc. An article[1] by Simon Kemp shows that in 2019, 45 % of the world's population are active in social networks. Furthermore, the amount of time that people spend on social networks has increased more in the last year, where the average time spent on these platforms now is 2 hours and 16 minutes per day. According to the article[2] by J.Clement and their forecast which is shown in Figure 1.1, the growth in the number of social network users around the world will increase furthermore, by the year 2021 an approximate of 3.02 billion people will use social networks. This shows how important social networks are becoming in our daily life, basically having an user in any of the mainstream social networks it is a necessity.

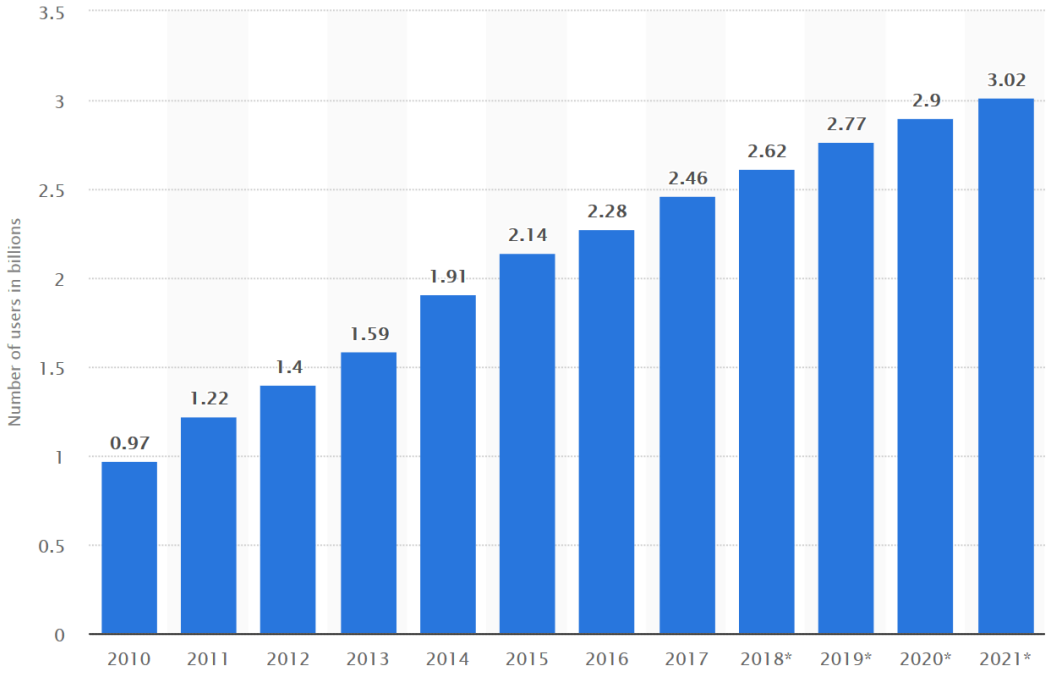


Figure 1.1: Social Network Users Worldwide, 2010-2021

A recent statistic [3] in October 2018, gives insight on the most popular worldwide networks, ranked by the number of users, making Facebook the largest ever social network surpassing 2.2 billion registered accounts and it is shown in Figure 1.2.

[1]https://wearesocial.com/blog/2019/01/digital-2019-global-internet-use-accelerates
[2]https://www.statista.com/statistics/278414/number-of-worldwide-social-network-users/
[3]https://www.smartinsights.com/social-media-marketing/social-media-strategy/new-global-social-media-research/

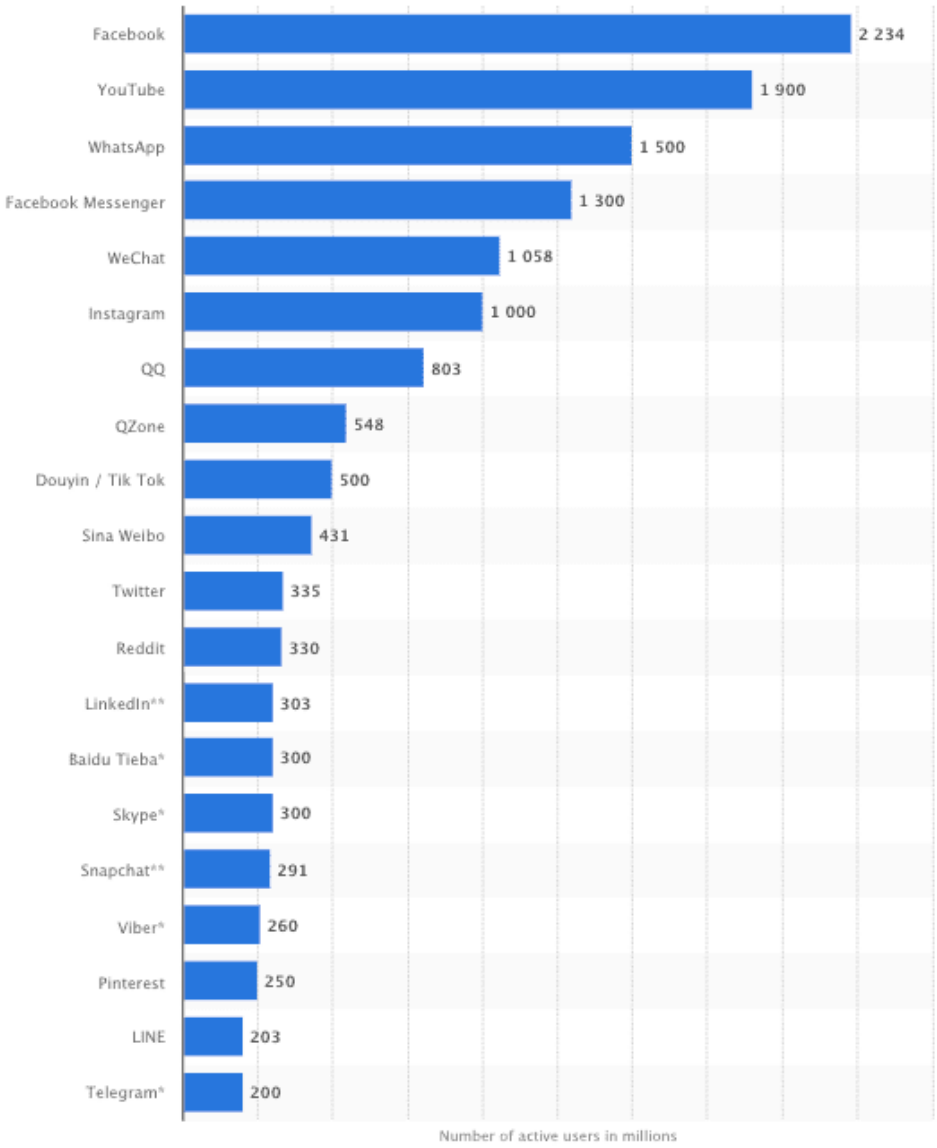


Figure 1.2: Worldwide social networks as of October 2018, ranked by active users in millions

1.2 Motivation

Today data owners can store large volumes of digital human social interactions which has made them rich sources of data that require attention and analysis. Data is important and very useful, researchers rely on the data to build upon their hypotheses, useful analysis and computational models [17]. A massive source of personal information is currently circulating by online social networks, data which also contains personal and potentially sensitive information, such as reflection of people's life on the popular social networks such as Facebook, Instagram and so on. This data is usually published in form of graphs. Nodes are representation of individuals and edges denote relationships or interactions between the nodes. In releasing such data, one needs to set up a protection mechanism for the privacy

of individual users while preserving the social networks properties. This is usually done by a procedure called anonymization, where upon publishing the data, the data owner removes the actual name, phone number, email address and replaces them by a random user ID [16]. A number of re-identification attacks have shown that even if we stripped of all nodes and edges attributes from a graph, the graph is still susceptible of leaking sensitive personal information contained in the network (for example, the existence of relations between users, religious beliefs, etc. The motivation behind this process describes that the sensitive information of a social network cannot be released as such, the researchers in this case are not concerned in which user correspond to each node, but in the properties of the graph, such as the density of edges, its connectivity, node to node distances and so on [13]. The use of such pseudonyms is insufficient to guarantee anonymity, specifically when you have to consider an adversary working with the graph releases. An attacker can cross-reference information from other sources, such as the number of connections, to find out the real user behind a pseudonym is [17]. In social graphs, the adversary's knowledge is regarded to as any subgraph that is isomorphic to a subgraph in the original social graph [1] , taking into consideration that any type of information an attacker may have is known as background knowledge and will benefit him to succeed in its attacks. The extent of attacking depends on what kind of knowledge the adversary has. There are two kinds of attacks, the passive attack and the active attack. The passive attack, an attacker simply observes data as it is presented and carries the attack by trying to re-identify individuals only after the anonymized graph has been released. In [2], Backstrom et al. describes a scenario of few existing passive attackers in the graph, who are able to discover their locations and collude with other users to construct a small coalition [19]. In contrast, an adversary in active attack tries to compromise privacy by strategically creating new user accounts and links before the anonymized network is released, so that these new nodes and edges are present in the anonymized network. So far, the threat posed by active attacks has been demonstrated in the scenario of publishing one snapshot of a social graph. However, the dynamic social graphs provides useful information for social scientists, market among others, who benefit from this data to extract statistics and analyze behaviours of the social graphs, e.g. finding communities of similar or related users, studying the spread of rumours or diseases, etc. Within this thesis, we will focus on the manner in which active adversaries benefit from the inherently dynamic nature of social networks. Consequently, it is necessary to sanitize the released information, to ensure that the privacy of the individual users is protected. Furthermore, the privacy risks in a dynamic graph publication should be better understood.

Therefore, the core of our study is the extension of the active attack model [2], by adapting the model to a periodically published dynamic social graphs and proposing new adaptable active strategies allowing the adversary to re-identify users along the sequence of snapshots. The new attacks proposed in this work, will serve as an environment on which new privacy-preserving techniques can be designed and tested against the active attacker.

1.3 Thesis Contribution

The goal of our research is to design a dynamic framework, presented as a game which considers dynamic interactions and evolution's of the graph and adaption strategies in the actions of the attacker and defender in a multiple graph release scenario. In this study we show that the proposed active attack with the adaption strategies does constitute a serious threat for privacy-preserving publication of social graphs.

The main contributions of this study are:

1. We perform an extensive analysis of all the active attack strategies that have been proposed.

2. We proposed an extension active attack model that is suitable to work in a dynamic framework

3. We proposed new active attack strategies on periodically released dynamic graphs to strengthen the adversary's capabilities against the defender.

4. We performed experiments that demonstrate the increased capacities the active attackers on a periodically released graph and how they constitute a serious threat for the privacy-preserving publication of the social graphs. From the conducted experiments we concluded that adapting the attacker's subgraph for multiple graph releases, helps the adversary keep in track with the growth of the graph and gain resilience with every snapshot.

5. We developed a library of dynamic graph models, that simulates the process of periodically released graphs, the growth of the network. In addition, we simulate the attacker's and defender's actions and allow the computation of statistics of the attacks.

As a summary, the contributions of this thesis are to provide adaptation strategies aimed at transforming the active attacks in dynamic social graphs so that the attacker can strengthen its capabilities with each release of the graph, meaning that it's background knowledge will benefit him/her from the previously released graph and the conducted attack. Furthermore, we practically evaluate and illustrate the importance of using the robust active attack presented in [17] and showing that by applying adaptation strategies to the attacker's infiltrated network, the adversary will have the upper hand against the data owner. We perform experiments in synthetic graphs in order to evaluate the proposed solutions. The results are the success rate of attacking social networks which are transformed by the data owner so the graph is not prone to the attacks. The proposed extended attack model and new attack strategies are a framework suitable to test new anonymization techniques in dynamical releases of the graphs.

1.4 Outline

This thesis is structured as follows. In Chapter 2, we present the related work covering existing approaches of re-identification active attacks and the evolution of graphs. Chapter 3, in the other hand illustrates the background concepts used in this thesis. In Chapter 4, we enunciate our proposed extended adversarial model in the form of an attacker-defender game. We present various attack strategies that make use of the proposed adaptation strategies, where we show how the adversaries capabilities can be strengthened. We continue with the creation of the data set in Chapter 5, by presenting the periodically released dynamic graph simulators. Furthermore we will analyze the design and validation of our experiments and also introduce the results of each experiment. Finally, the thesis is concluded in Chapter 6 with a brief summary and conclusion of the performed experiments and directions for future work.

Chapter 2

State of the Art

In this chapter, we perform an elaborate study of literature review, with emphasis in privacy attacks on single release social graphs, including two attacks according to the manner in which the attacker chooses to apply when attacking a released single snapshot of a social network. Based on the previous work, we show that our study is a new privacy attack on social networks, not only on the existing one released version of the social graph, but in periodically released versions of the graph.

2.1 Privacy attacks on social networks

Social networks are widely used in the modern life. The main reason for threats in social networking is due to its open architecture and its susceptibility to different attacks [18, 2, 22, 23]. Due to this open structure an adversary can compromise user's privacy by using the published social network, which mainly results in the release and collection of sensitive data such as personal data, bank accounts details, e-mails and relationships amongst users[8]. To perform such attacks adversaries, rely on background knowledge, which is usually the structure of the social network, more specifically, they rely on vertex degrees [25] or neighborhoods [36]. Furthermore, as presented in [11], the attacker might have background knowledge on communities, in and out degree distribution, which assumes a pattern of attacks on static graphs. The first to show the impact of attacks on social graphs where Backstrom et al. where they introduce the attack methodology not only for the active attacks but also for the passive attacks [2]. As a passive attacker, the adversary can discover its location in the social graph G and use their knowledge about the structure of the network around them. There exist different type of passive attacks, one of the most known is the coalition of passive attackers that collude to discover their location in the graph. In such a way, the attackers compromise the privacy of some of the connections with neighbors. There exists another category of passive attack which targets de-anonymization of specific individuals by attackers [21]. As presented in [21] by Narayanan et al, in this scenario the attacker has a detailed contextual information about a single individual, which includes

some of the attributes, a few of the users relationships and so on. This particular attacks goal is to use this information in order to recognize the victim's node in the anonymized network and to learn sensitive information about the victim. Passive attacks to social networks can be combined with active attacks in a way that the adversary apart from having background knowledge on the structure of the social graph it also controls a subset of attacker nodes in the original graph G [17]. Active attacks rely on the structure of the social graph and the ability to alter it by introducing structural properties that are known for the adversary and allow them to re-identify the victims after the publication of the social graph. The most widely studied attack is the one proposed by Backstrom et al. in [2] based on the background knowledge of the adversary that is known as the walk based attack. This attack same as all active attacks provides the attacker with the property of infiltrating several malicious nodes in the network known as sybil accounts. The sybil accounts create relationship between themselves by enforcing a long path connecting the sybil nodes in a defined order and the other relationship edges amongst themselves remain randomized between the sybil nodes with a 50 % chance of connection. Furthermore, the sybil nodes create relationship with victims as well by constructing unique fingerprints to define sybil to victim edges which helps in creating re-identifiable patterns for the victims.
Therefore, when the social graph is published, the adversary retrieves its infiltrated sybil subgraph and re-identifies the victims which preserve the fingerprints. The second proposed active attack was the cut-based attack, where a subset of the sybil nodes is guaranteed to be the only cut vertices linking the sybil graph and the rest of the graph. The difference of both attacks is the manner of receiving the published graph in order to re-identify the victims. As was mentioned before, Backstrom et al.[2] also studied the passive version of these attacks, where fingerprints are also used as an identifying property of the sybil network but in difference to the active attack they do not consider constructing and inserting sybil nodes in the graph. Instead, they model the situation where the users start colluding with each other to share their neighborhood relationships and information in order to retrieve their sybil subgraph and re-identify as much as they can of their remaining neighbors. Sybil nodes guarantee that an active attack is successful by providing unique fingerprints and also a reliable and re-identifiable subgraph. Such attacks can be detectable by sybil detection techniques [9, 34, 33]. Two of the main reasons for an attacker to be discovered and fail are inserting many sybils in the way that a sybil detection technique can remove them, and the other reason is that the attack suffers from low resilience, in the sense that the attacker's chance of recovering its sybil and re-identifying its victims is a relatively small percentage. In their paper, Backstorm et al. in [2] argue that inserting a sufficient number of nodes would prove a successful attack, if the insertion would be in the order of log n for graphs of order n. These nodes would be sufficient to attack any legitimate node. Considering the lack of resilience of the active attack proposed by Backstrom et al. in the noisy releases of the social graph, pushed Mauw et al. in [17] to construct a more robust attack, where in their study they show that robustness lead to considerably more successful attacks. In their paper it is presented that active attacks do constitute a serious threat for privacy-preserving releases of social graphs. The proposed robust attack has new key properties. Firstly, it

can effectively re-identify the users using a small number of sybil nodes. Secondly, it is resilient in the sense that it resists the noise added to the graph with the algorithms of finding the approximate fingerprint and not exactly the original one from the original walk based attack. Furthermore, their attack is resilient in the application of various anonymization algorithms constructed specifically to counteract active attacks [18]. This attack is based on notions of robustness for the sybil subgraph and the set of fingerprints which we will describe in more detail in Chapter 3. Apart from active attacks, Peng et al. in [22, 23] introduced hybrid attacks which are a combination of active and passive techniques. These attacks rely on a small set of sybil nodes which are called seeds. The introduced approach is called Seed-and-Grow attack, they propose an attack where they combine the creation of a uniquely small set of sybil with a progressive self-reinforcing strategy to acquire re-identification of some of the victims connections by using the initial fingerprint and extending to other nodes in the neighborhood and gather as much information as it can. This attack is a iteratively algorithm which uses the set of victims that were re-identified in the previous step.

All the mentioned attacks were constructed to be single-snapshot attacks. In a dynamic framework, where we have multiple releases of a graph, every release is considered as a snapshot of the graph at the specific period, either in time or regarding the growth of the graph. None of the attacks above-mentioned capture the ability to perform on a periodically released graph with multiple snapshots, whereas they were constructed to be effective only in static releases of the graph. The one which was the most promising is the robust active attack mentioned in [18] and which will be the main point of this work and most certainly will lead to other anonymization methods to counteract against active attacks in periodically released graphs.

2.2 Sybil Defense Mechanism

So far the active attacks have had the ability to compromise the whole network, where the advisory claims multiple identities to compromise the whole network. Knowing that there can be malicious node in the social network, each social graph owner takes some countermeasures in case of an attack. One of the mentioned defence strategy known as Sybil guard in [34] is used by social graph owners to detect and remove sybil nodes. Another defence mechanism called Sybil limit [33] assumes that O(log n) is the bound of accepting malicious nodes, where n is the total number of nodes in a social graph. Based on this anti-sybil detection scheme, Sybil defender in [31] uses community detection, where it starts by finding the sybil node and continues finding the region surrounding the sybil node which is assumed to be the sybil network. The assumption is built in the fact that the number of attack edges is limited. Once a sybil node is detected, the whole sybil network is found based on the intuition that the sybil nodes generally tend to connect with each other.

2.3 Anonymization

A large body of existing work in data anonymisation put an emphasis on the notion of k-anonymity [27]. One of the proposed notions of k-anonymity came from, Liu and Terzi [13] where they consider an adversary who knows the degree of the victim node. A graph is set to be k-degree anonymous if for every node of the graph there exists *k-1* other nodes with the same degree [16]. In this work, they devise a simple and efficient algorithm to transform the graph into k-degree anonymous graph. Regarding this notion, further improvements of this notion were proposed in [6, 5, 7, 14, 15, 24, 26], where the goal was increasing the utility levels. Other examples of structural privacy properties that are presented in [36] defined as k-neighbourhood anonymity and k-automorphism anonymity in [37]. In [36], the authors present the privacy notion to be stronger than k-degree anonymity, this property requires that for every node n in the graph there exists at least k-1 other nodes such that the subgraph induced by the neighbours of n is isomorphic to the subgraph induced by all the other nodes neighbours [19]. A strictly stronger privacy notion than k-neighbourhood anonymity is the k-automorphism anonymity, where two nodes u and v are considered to be equivalent if there is an automorphism of the graph that maps u to v [37]. As the authors discuss, in real-life social graphs it is really unlikely for them to satisfy the k-automorphism anonymity. Privacy-preserving methods in a released social graph are very hard to achieve, most of the anonymization method that exists [32, 13, 37, 5, 6], can not effectively counteract active attacks. The first work to be able to counteract against active attacks was recently proposed by Trujillo-Rasua and Yero, where they propose that the adversary's background knowledge is a distance vector of the vertex with respect to the adversary's subgraph[18]. In most cases, the data owner while in the process of publishing a new version of the social graph adds perturbation. Perturbation as described in [17], is the process of flipping edges. Each flip consists in randomly selecting a pair of vertices, if there is an existing edge between it gets removed, otherwise they add an edge.

2.4 Evolution of graphs

The aforementioned literature review only considers "one-time" release of network data. However, knowing that this is not sufficient for evolutionary networks and dynamic social network analysis, we consider that it is a necessity to re-publish network data periodically [3, 10]. With each publication of the graph and with privacy measurements in tact, the adversary can still identify the target with a high probability having the background knowledge on the previous sequence releases[37]. Furthermore, many studies have discovered pattern of attacks in static graphs, analyzing and identifying properties in a single snapshot of a large network, including the information about communities, in and out degree distribution and so on [12]. The evolution of graphs relies in many properties such as nodes degrees and also the distance between the pair of nodes. The most intensively-studied graph model which can be used for periodically released social graphs is the preferential attach-

ment model of Barabasi and Albert in 1999. This graph presents a network in which each new node attaches to the existing network by a constant number of outlinks using the rich-get-richer rule [4].
The existing work on growth patterns of these networks uses two principles which are essential for a dynamically released network. Two principles that are used are the densification power laws and shrinking diameters. The first one, shows that the networks are becoming denser over time while as the average degree increases, this is a living embodiment of the real life in social networks, an example would be a new user which is a blogger in Instagram and the multiple connections that it would have with multiple users. The second principle shows that even though the network grows the diameter shrinks[32]. Another model which is more sophisticated but enforces both of the principles is the Forest Fire Model [30] which is based on having new nodes to the network by burning through existing edges.
Furthermore, there is another generative graph model that obeys all common network properties and the proposed algorithm is fit to construct effectively the structure of a real network [11].

Chapter 3

Preliminaries

In this section, we provide some definitions and notations which show how active attacks work on a single-snapshot graph release. This is the baseline of our work and we will describe how the original adversarial model works. Furthermore, we will briefly explain two types of active attacks from related work that are well suited to use this adversarial model.

3.1 Notation

- A social network is a graph G which is composed by pairs *(V,E)*, where V is a set of vertices that represents users in social network or else known as nodes and E is a set of edges which comes as a result of V x V and represents the relationship amongst the users.
- The vertices of the graph G are denoted as V_G and each vertex is labeled with letters $v_1, v_2, ..., v_n$ and its edges are denoted as E_G. We model graphs that are undirected, which means that the the pair of vertices having an edge between *(v,u)* is the same as *(u,v)*, without self-loops and multiple edges.
- The **distance**s d_G(v,u) between two vertices v and u in G is the number of edges in the shortest path connecting them.
- **Graph Isomporphism.** Given two graphs G= (V_G, E_G) and G'=$(V_{G'}, E_{G'})$, G is isomorphic to G', if and only if there exists a bijective function $f: V_G \rightarrow V'_G$, such that for any edge (u,v)$\in E_G$, there is an edge (f(u),f(v))$\in E'_G$ [17].
- The **degree** of the vertex is the number of edges connected to it. The set of neighbours of a set of nodes W $\subseteq$ V is defined by $N_G(W) = \{v \in V \backslash W \mid \exists_{v \in W} \cdot (v, w) \in E \vee (w, v) \in E\}$. If W=$\{w\}$ is a singleton set, we will write $N_G(w)$ for $N_G(\{w\})\}$. The degree of a vertex v $\in$ V, denoted as $\delta_G(v) = |N_G(w)|$ [17].

- Let G=(V,E) be a graph and let S $\subseteq$ V. The ***weakly-induced subgraph*** of S in F, denoted by $\langle S \rangle_G^w$, is the subgraph of G with vertices S $\cup N_G(S)$ and edges $\{(v, v') \in E \mid v \in S \vee v' \in S\}$[17].

3.2 Active attack on single-snapshot graph

We assume that the social graph is a graph with n- nodes, where nodes interact with each other. The interaction of the nodes is represented by edges, and since the node correspond to user attack, an edge between two nodes will indicate that there is a connection between them. Either a communication or a newly created friendship. Existing papers on attacks of social graphs in one single-snapshot, show that active attacks can be used to reveal true identities of targeted users [2, 17]. The structure of the active attacks, consists of creating k new user accounts and linking them together to create as such a small subgraph inside the original social graph. It then uses these accounts to create more connections to nodes that we target as victims. These nodes are existing users in the social graph represented as $\{w_1, ..., w_n\}$. Now when we have an anonymized release of the graph G, the attacker's subgraph denoted as H along with the connections with the set of victims will be present on the anonymized graph. The attack consist of the attacker identifying its subgraph, by re-identifying the true location of the victims in G and its sybil connections to the set.

3.2.1 The original attacker model

The attacker model is an attacker-defender game with two actors, attacker $\mathcal{A}$ and defender $\mathcal{D}$. The game starts with a single snapshot of a social network which is a graph G =(V,E) as depicted in Figure 3.1 (a). Before the graph is released, the attacker infiltrates sybil nodes in the social network and establishes connections with victims, as depicted in 3.1(b). The sybil accounts are visualized with dark red color and the edges known to the adversary are solid, since he/she manages to enforce the path between the sybil nodes. Seeing that the goal of the attacker is to re-identify the victims and learn as much information as it can from them, the goal of the defender is to anonymize the social graph by removing real identities from the nodes or replacing them with pseudonyms and adding perturbation to the social graph. In figure 3.1(c) we illustrate the process of pseudonymization. Therefore, after the release of the pseudonymised graph, the attacker has the chance to analyze the social network that its now in their hands and start by re-identifying its own sybil nodes and the the victims as shown in figure3.1(d). This allows the attacker to have information that C and D are connected.

From the Figure 3.1, we can imagine a scenario where an adversary in a social network can create as many accounts as he/she wants and send multiple friend request to other people which will be victims. In this case, the attacker has created multiple relationships with victims using all of its accounts. We can consider Jane represented as node 1, Mike as node 2 and Emma as node 3 being attacker

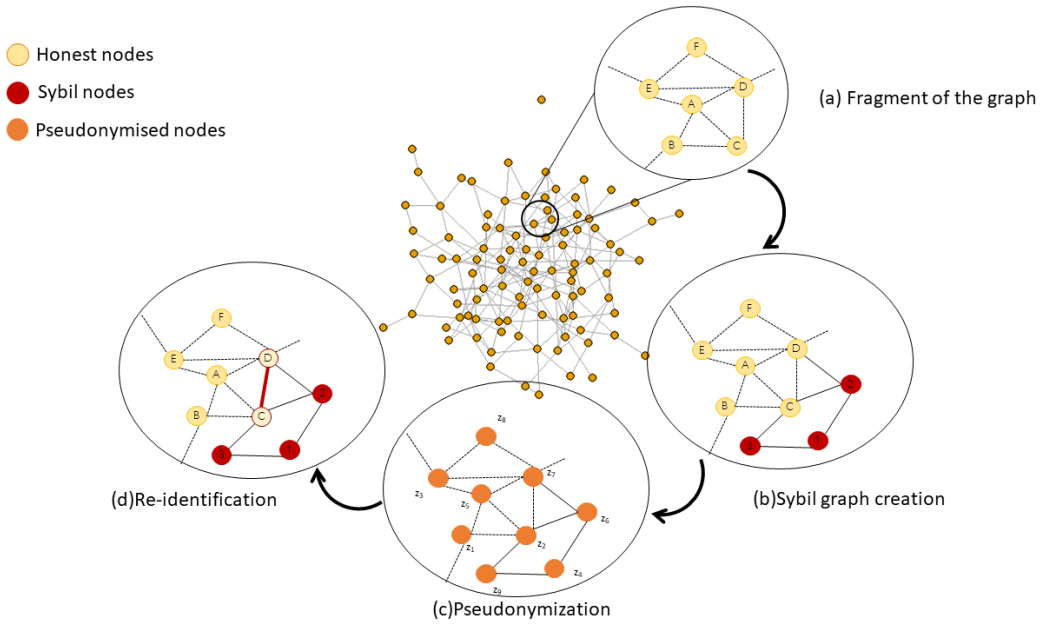


Figure 3.1: The original attacker-defender game

accounts. They have created connections to Dan represented by node D and Carol represented by node C. It is illustrated, that node 2 and 3 both have a connection to C - Carol and 2 has another connection to D - Dan. So the attacker, can benefit from this by re-identifying its sybil nodes and their fingerprints and in such a way understands that Dan and Carol are friends, information that before was hidden due to the pseudonymization.

3.2.2 Description of the adversary model

Let *G=(V,E)* representing a n-node graph in a released anonymized social graph. We describe the model from the attacker's point of view. As discussed, the active attacker will choose a set of k named users, $W = \{w_1, ..., w_k\}$, that we wish to target in the network and the attacker wants to learn the connections that they have between them. So the attacker is aiming to get information about the edges that are in *G*. To find, each element of the set *W*, the adversary uses a well known strategy. It firstly creates a set of k new user accounts, represented as $X = \{x_1, ...x_k\}$, which will be inside of the released graph. The next step, is creating the connections between (x_i, x_j) with probability $1/2$. In such a way, the attacker constructs a random graph H on X [2]. The construction of the attacker's subgraph continues by creating edges between (x_i, w_i) for each i. The connection in a real network might involve x_i sending w_i a friend request, an email or some other communication method that is offered by the real social networks. The basic version of this attack, assumes the fact that the account x_i corresponds to a fake identity and as such will not get any potential message or friend request apart from the one that he already managed to create when connecting with w_i. When the anonymized graph G is released, the attacker starts the search to find its infiltrated subset H, and to correctly re-identify its sybil nodes as $x_i, ...x_k$. Having found these nodes, the next step is to find the targeted users $w_1, ..., w_k$ as the unique nodes in *G-H* that are linked to x_i [2]. In such a manner, we identify the full set of targeted users in G and the attacker now can make use of the information of which users are connected between edges in the graph.

This whole idea looks easy and forthcoming, a number of technical steps need to be taken care of before starting this attack. Firstly, one should assume that certain subgraphs have the same structure as each other. In [2], Backstrom et al. introduces the terminology of the symmetry between subgraphs. Firstly, they start by introducing a set of nodes *S*, and denoting *G[S]* as a subgraph of *G* induced by the nodes *S*. An isomorphism between two sets of nodes *S* and *S'* is presented as the mapping of edges (u,v) in $G[S]$, if and only if there is an existing edge (f(u),f(v)) of *G[S']*. Therefore, *G[S]* and *G[S']* are called isomorphic if the are the same graph when re-identifying them [2]. Whereas an automorphism is an isomorphism from the set *S* to itself, a relabeling of the nodes that preserves the graphs structure [2]. The construction of such an attacker's subgraph succeeds if the attacker fulfills one of the conditions presented below:

- There is no subgraph different from the constructed *H*, such that *G[S]* and *G[S'] = H* are isomorphic.
  If this holds, then any image of the subgraph *H* that we find in *G* is exactly the one that we constructed [2].
- The subgraph *H* can be efficiently found, given the original graph *G*, if this holds, the we can in fact find the copy of *H*.
- The subgraph *H* has no non-trivial automorphisms, if this is the case once we find *H*, we can correctly re-identify our sybil nodes as $x_1, ..., x_k$ and hence find our targeted victims $w_1, ..., w_k$ [2].

In this section we summarize the stages of this attack model.

1. **Creation of the attacker's sybil subgraph**

   The attacker constructs a set of *sybil* nodes which are described as $S = \{x_1, x_2, ..., x_{|S|}\}$. The sybil nodes should be added on the actual set of vertices V, such that $S \cap V = \emptyset$. The set of edges in this network is the combination of relations between the sybil nodes and sybil nodes with vertices in the original graph, $F \subseteq (S\ x\ S) \cup (S\ x\ V) \cup (V\ x\ S)$ since the graph is undirected and edges between vertices (u,v) or (v,u) are the same.
   The attacker does not know the complete graph G^+ but only its weakly-induced subgraph $\langle S \rangle_G^w$ in G^+, which is called the attacker subgraph [2, 17]. The attacker sybil network creation has two phases:

   (a) ***Creation of inter-sybil connections***.
   In this phase, the attacker creates a unique and efficient retrievable connection between the sybil nodes to facilitate the retrieving of sybil nodes in the final phase.

   (b) ***Fingerprint creation***.
   In a sybil network, each victim will has its unique fingerprint. A fingerprint of a given victim vertex are the victims neighbours in S,$y \in N_G^+(y_i)(S) \backslash S$.
   Considering the set of victims $Y = \{w_1, w_2, ..., w_k\}$ the attacker makes sure that the fingerprint of each victim is unique, hence $N_G^+(w_i) \cap S \neq N_G^+(w_j) \cap S$ for every $w_i, w_j \in Y, i \neq j$ [2].

2. **Anonymisation.**
   The defender secures G^+ by constructing an isomorphism φ from G^+ to φG^+ [17]. This graph is now a pseudonymised graph. In this phase the defenders goal is to remove all personally identifiable information from the vertices of G. Both $\mathcal{A}$ and $\mathcal{D}$ know that the defender will also apply some modification to the graph, respectively adding or removing vertices or edges. This will result in the final transformed graph $t(\varphi G^+)$[17].

3. **Re-identification.**
   The last stage of the game is the re-identification of the sybil nodes and victims from the attacker's side. In this stage, the attacker obtains the final transformed graph $t(\varphi G^+)$ and executes the re-identification attack in two unique stages.

   (a) Attacker subgraph retrieval. In this stage, the attacker determines the isomorphism φ which is restricted to the domain of sybil nodes S=$\{x_1, x_2, ..., x_{|S|}\}$ [2, 17]. In such a way, the attacker finds its labeled sybil nodes by finding the path that was constructed when infiltrating the sybil graph.

   (b) Fingerprint matching. In this stage, the attacker determines the isomorphism φ which is restricted to the domain of victim node s Y=$\{y_1, y_2, ..., y_m\}$.

The whole re-identification attack depends on the last-step of the attacker-defender game. The attacker is successful if it can determine effectively the isomorphism φ restriction on the sybil network, respectively on the domain of victim nodes. Thus, when the attacker re-identifies all the victims in Y=$\{y_1, y_2, ..., y_m\}$ in the anonymised graph, we consider the attack to be successful [2, 17].

3.2.3 Variants of active attacks

In this section, we will describe how two of the active attacks in single-releases graphs, the original walk based attack[2] and robust attack [17] construct the sybil network and how they re-identify it. The attacker's goal to succeed are divided in three stages. These stages consist of: sybil subgraph connection and fingerprint creation, attacker subgraph retrieval and fingerprint matching.

1. Sybil graph construction and fingerprint creation
   For creating the internal links of sybil subgraph both of the approaches use the same strategy as presented above based on [2]. By creating the set of sybil nodes and the set of victim nodes, the construction of edges between the nodes works in specific manner. Firstly, the sybil nodes have an enforced path from (x_i, x_{i+1}) and we include the edges between them and the edges with other sybil nodes (x_i, x_j) are enabled with a probability of $1/2$.

   1.1. Original approach
   As described in the previous section, the original approach constructs the fingerprints in a randomized manner. Firstly, the attacker creates the set of targeted victims such as $W = \{w_1, ...w_b\}$, for a value $b = O(log^2 n)$ [2]. Therefore, for each targeted victim w_j we randomly choose a set of neighbours represented as N_j which consists of all of the sybil nodes that were created in the previous step, $x_1, ...x_k$. For each victim, the neighbours set is distinct and unique. The length of the created fingerprints is the same as the sybil set size. In addition, when creating the neighbours set, one has create also the edges between the respective nodes(x_i, w_i).

   1.2. Robust approach
   In contrast to the original approach, the robust attack proposes that it is not necessary to re-identify the exact same fingerprints. Instead Mauw et al. proposes to obtain a set of fingerprints that is close enough to the original set of fingerprints [17]. A fingerprint is a set of vertices, more specifically the vertices that are considered as target victims, the robust approach of creating fingerprints offers the cardinality of the symmetric different of two fingerprints to measure the distance between them. As presented in [17], an *independent set* I of a graph G is a subset of nodes from G such that all nodes in I are not linked by edges. If the graph is constructed in such a way for every pair of fingerprint whose distance is less than some value of i, the independent set guarantees a set of finger-

prints with minimum separation of at least *i+1* [17]. The lower bound of distance between the fingerprints is introduced as *minimum separation*. Given a set of victims which a set of fingerprints should be defined, the larger the minimum separation of these fingerprints, the larger the number of perturbation can be tolerated after the re-identifying stage in a transformed graph. The attacker will still be able to re-identify the perturbed fingerprints and map them to the correct victims. The idea of the proposed algorithm in [17] is to arrange all possible fingerprints, in such a way that nodes representing similar fingerprints are linked by an edge, and node representing well-separated fingerprints are not. The fingerprint generation method works iteratively and creates denser fingerprint graphs. The robust approach applies a greedy algorithm for maximizing the minimum separation of fingerprints, the method iterates until it finishes finding the smallest maximal independent set that still contains at least m fingerprints [17]. If the maximal set does not contain exactly m fingerprints then only the successive runs of the attack can randomly construct m fingerprints.

2. Attacker subgraph retrieval

   2.1. Original approach
   When the graph is released, in this approach we start by identifying our infiltrated sybil network denoted earlier as H. The search to identify the subgraph starts by firstly finding the path that was enforced for the sybil nodes $x_1, x_2, ..., x_k$. The proposed search algorithm is a pruning approach based on two tests, a degree test where each possible candidate sybil nodes has the correct degree and the internal structural test, where we search if each of the sybil nodes has the appropriate edges in its subset of neighbours $\{x_1, x_2, ..., x_k\}$. We will describe the algorithm briefly, the search algorithm includes construction of a search tree, where every node apart from the root corresponds to a node in G. The tree is constructed, so the path of the nodes of the graph G has the same degree sequence and same internal edges as the sybil nodes path. Taking each leaf node, which represents each node in the graph G, with the created path, we find all the neighbours of the targeted victims. For each identified victim, we create a new child in the tree [2].

   2.2. Robust approach
   Considering, that the robust attack argues that exactly matched fingerprints are not necessary since we consider that the fingerprints have been perturbed. The authors in [17], define a dissimilarity measure to compare candidate subgraphs to the original attacker subgraph. In order to reduce the space, they consider a perturbation threshold sot that the search procedure is feasible. The search procedure expects the existence of a fixed order of the sybil nodes, such as the path amongst sybil nodes that was described before. The proposed search procedure assumes that the transformation of the graph did not remove the image

of any sybil node from the perturbed release of the original graph G. The method performs BFS, which analyses at the *i-th* level for possible matches of the sybils path $\{x_1, ..., x_k\}$. After the pruning of the tree, the algorithm outputs a set of fingerprints that are used as an input in the fingerprint matching phase, if the algorithms outputs an empty set, then it is safe to say that the attack failed and the re-identification of the fingerprints was not possible. The details of the algorithms are found in [17].

3. Fingerprint matching

   3.1. Original approach
   In the original walk based approach, if there is a unique path of sybil nodes represent in the before mentioned tree search algorithm, one assumes that the path corresponds to the nodes of H. Having retrieved the sybil subgraph, we then find the targeted victim nodes $w_1, ..., w_k$, emphasizing that w_j is the only node with connections precisely to its neighbours set N_j [2].

   3.2. Robust approach
   In contrast, the robust approach introduces the noise-tolerant fingerprint matching process. The proposed algorithm is a depth-first search procedure. Firstly, the algorithm starts by finding for all the existing fingerprints, the set with the most similar candidate fingerprints and it only considers the ones that reach the minimum distance thresholds [17]. There might be the case that the algorithm outputs one or several partial re-identifications, the reason of which is that one candidate fingerprint may be equally similar to the set of original fingerprints. Thus, the algorithm recursively finds the set of best re-identifications for every partial re-identification and combines them to construct the final set of approximately the same fingerprints.

3.2.4 Summary of preliminaries

As described in the previous chapter the original work on re-identification attacks as presented by Backstrom et al. [2], aims at re-identifying the victims in pseudonymised graphs. In this work, they state that the uniqueness of the fingerprint is sufficient to reach a high probability re-identification of the victims, providing that the attacker subgraph is correctly retrieved. The success rate of this attack comes from the fact that it relies on finding the exact matches between the fingerprints created by the attacker at the beginning of the attack and the images of the transformed graph $t(\varphi G^+)$. The attack surely fails after a considerably small amount of perturbation that is introduced in the graph touches the sybil network, since it will be harder to find the exact match of the fingerprint.
In the robust re-identification attack, it was observed that it is not a necessity to obtain the exact same fingerprints in the perturbed graph but rather obtain a set of fingerprints that is close enough to the original set of fingerprints.

Chapter 4

Our extended adversarial model and new attack strategies

Most academic research have typically focused on a static model with a particular attack or defense on security of the social graphs, without considering dynamic environment and the interactions between the attacker and the defender, whereas in this specific model we are constantly adjusting the attacker's strategies to strengthen his capabilities. In this chapter, we introduce a novel approach for an offensive mechanism using several attack strategies. We also model the interactions between the attacker and defender in the adversary model by introducing new actions.

This chapter illustrates the main motivation of this work. It will be a description of a new attacker-defender game with extensive new attack strategies, due to the new circumstances such as working with a number of releases of the same graph and allowing the attacker to use the information from each snapshot to maximize his success.

4.1 Description of the extended adversary model

In the previous chapter we mentioned the theoretical properties of the adversary model. In this attacker-defender game we will add some new notation.

- A periodically released graph, is a graph that has multiple images of the graph released in a period of time or after a sequential growth of the graph.
- Each graph G_i will have a graph sequence $\mathcal{G} = \{G_{p1}, G_{p2}, ..., G_{pn}\}$ where p represents the graph state at the specific period in the time line.
- A graph G in the dynamic framework is represented by vertices and edges, $G_i = (V_i, E_i)$. Where, V_i is the set of vertices in snapshot i and E_i is the set of edges in snapshot i.
- The players will perform actions periodically on snapshots. The defender

periodically adds perturbation to the graph at the i^{th} release or as he/she sees meaning that he can perturb the graph at a certain point in time. On the other hand, the attacker periodically add sybils and creates fingerprints by using adapting strategies so that the sybil subgraph and the victim size set per each release will be evolved, assuming that the graph evolves over time. In addition, the attacker performs the re-identification attack for each release of the snapshot.

- The published graphs are labeled, in such a way that the nodes are pseudonyms and the labels are consistent from one snapshot to the next. Consistency means that a user that is present in snapshot i should have the same pseudonym in that snapshot and the previous one *i-1*.

4.1.1 Extending the Attacker-defender game to periodically released social graph scenario

We extend the game between the attacker and the defender. The proposed attack has various strategies for both the attacker. In this scenario the attack will not have only three stages, but we will introduce more stages of how the proposed adaptation stages for the attacker can be iteratively run for periodically released graph. Identifying the victim node after pseudoanonymisation and transformation by the defender is still the goal of the attacker in this game.

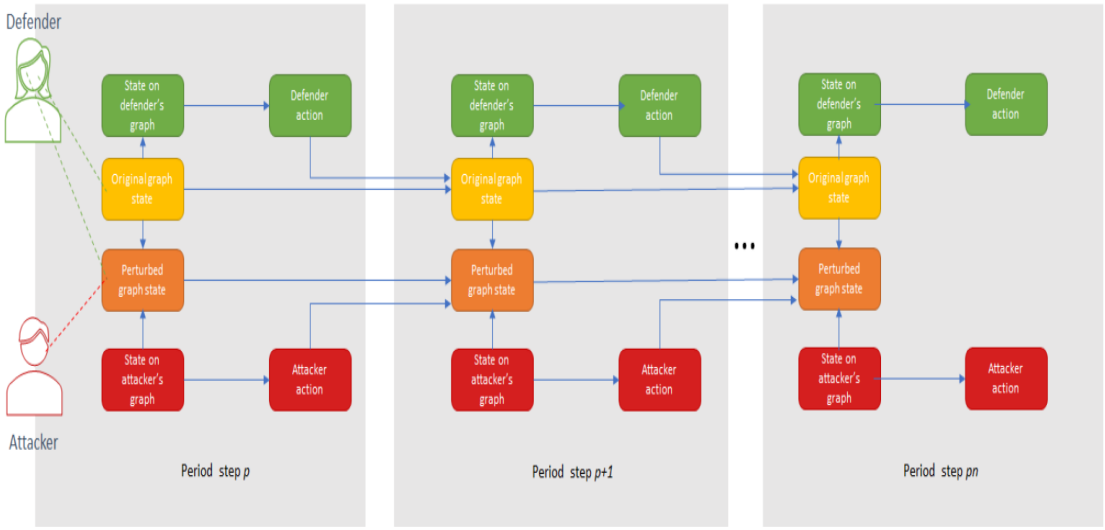


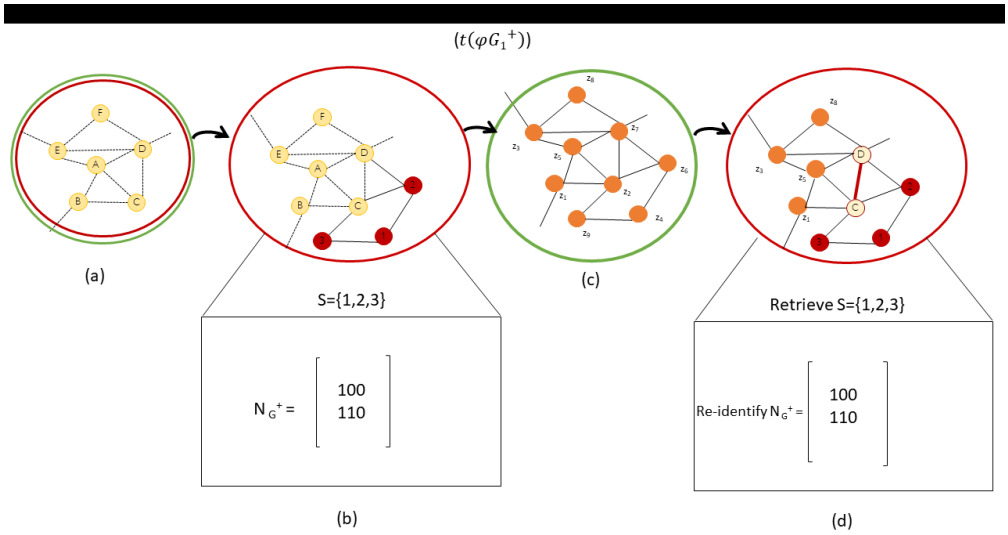
Figure 4.1: Attacker-defender perspective on the graph

This game starts with a fragment of the graph $G_i = (V_i, E_i)$ representing a snapshot of a social network. In Figure 4.1, each snapshot is represented by a period step. The presented attack will have periodically released snapshots of the social graph, simulating in such a way periodically released graphs which are as close as a dynamic graph. In Figure 4.1, we can see that each of the players has different views on the graph. The defender will always know the original graph state and

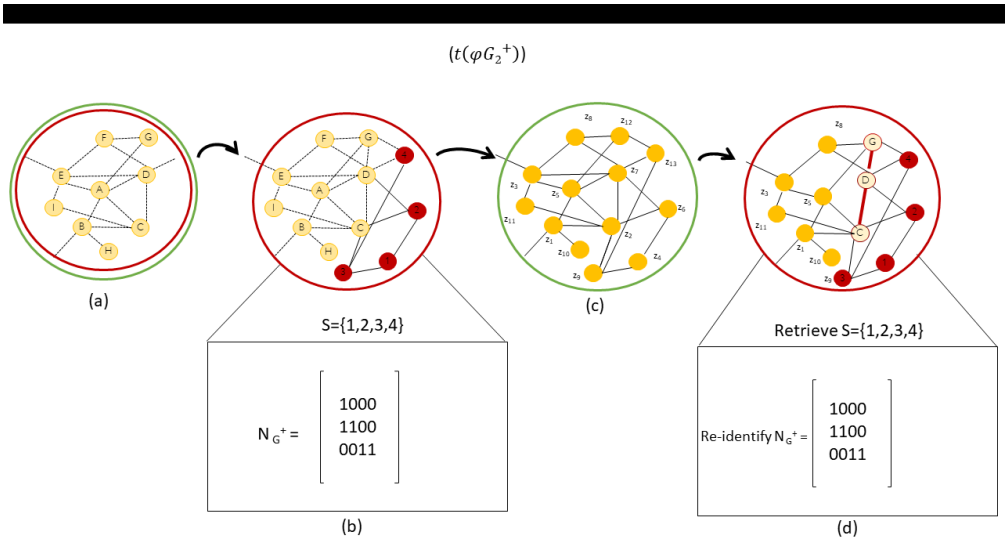
the perturbed graph state, where as the attacker will only have access to the perturbed graph state. The perturbed graph state will be a release of the social network where the attacker can than use the parameterized strategies to attack the graph. The view from the attacker perspective can be analyzed in details in Figure 4.2.

4.1.2 The extended game

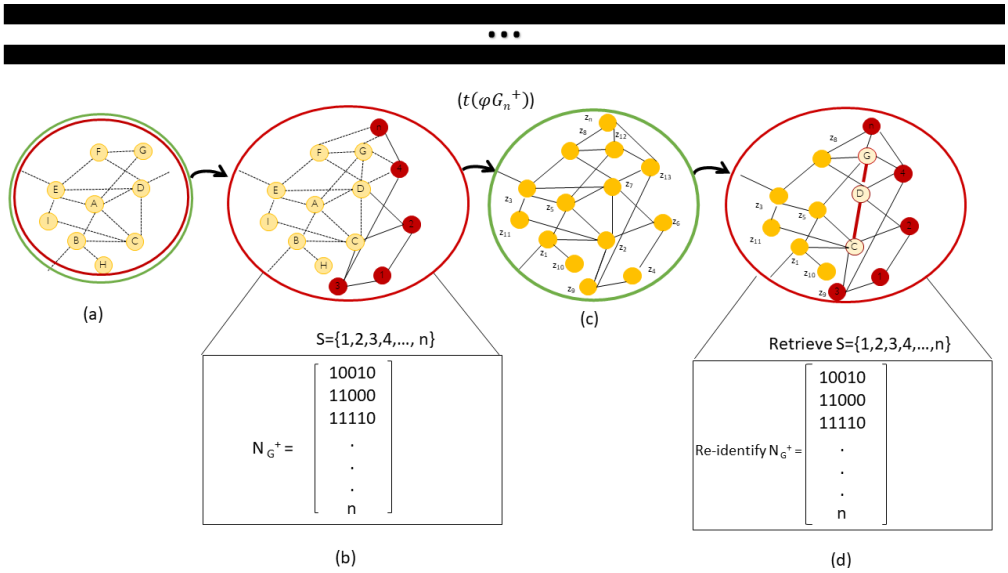
In the proposed game, we have an attacker and a defender. The attacker's objective is the same as in the original game, re-identifying as much victims as he can from the social graphs. We model a scenario where the attacker will use adaptation strategies to be successful in a dynamic graph release framework. The dynamic graph release framework consists of several stages. As depicted in Figure 4.2, we simulate in such a way a multiple release of snapshots of the social graph. In each snapshot, the attacker will apply a set of adaptation strategies. The original graph will evolve between snapshots over time. Figure 4.2, shows the dynamic framework of the graph releases and the player's action along the way. We have to put an emphasis that the green circle represents the action for the defender and the red circle is the action for the attacker, same as was displayed in Figure 4.1, whereas the nodes of the graph will have the same yellow color as the original graph state and then after pseudonymization and perturbation they have the orange color as represented in the Figure 4.1 as a perturbed graph state. As a starting point, we analyze the first snapshot focused in Figure 4.3, where from (a) to (d) it is indicated which player is doing the action by the color of the spherical shape. These stages are in accordance with the first overlook of the game presented in Figure 4.2. In Figure 4.3(a), the attacker knows a subset of user but not the connections between them, in this case the nodes are represented as real identities of the users by capital letters and the dotted line represent the relations that exists between them and the attacker does not know this information. The attacker will start its attack by infiltrating a subset of nodes/users in the fragmented piece of the graph as seen in Figure 4.3(b), where the sybil node are represented by dark-coloured red nodes and the edges known to the adversary are solid lines.



(a) Graph state in the first snapshot- timestamp 1



(b) Graph state in the first snapshot- timestamp 2



(c) Graph state in the n^{th} snapshot

Figure 4.2: The evolution of the game for both players in multiple releases of the graph

In Figure 4.3(b) we see that the adversary has managed to create connection between its sybil nodes and the victim nodes, by creating in such a way randomized unique fingerprints. The two sub stages of the construction of sybil network are presented here, such as the inter-sybil connections where we see all the sybil nodes connected to each other and the fingerprint construction, were the connection between sybil nodes and victims is done randomly. After the adversary finishes its attacker subgraph creation, the defender before publishing the graph perturbs the graph as can be seen in Figure 4.3(c).

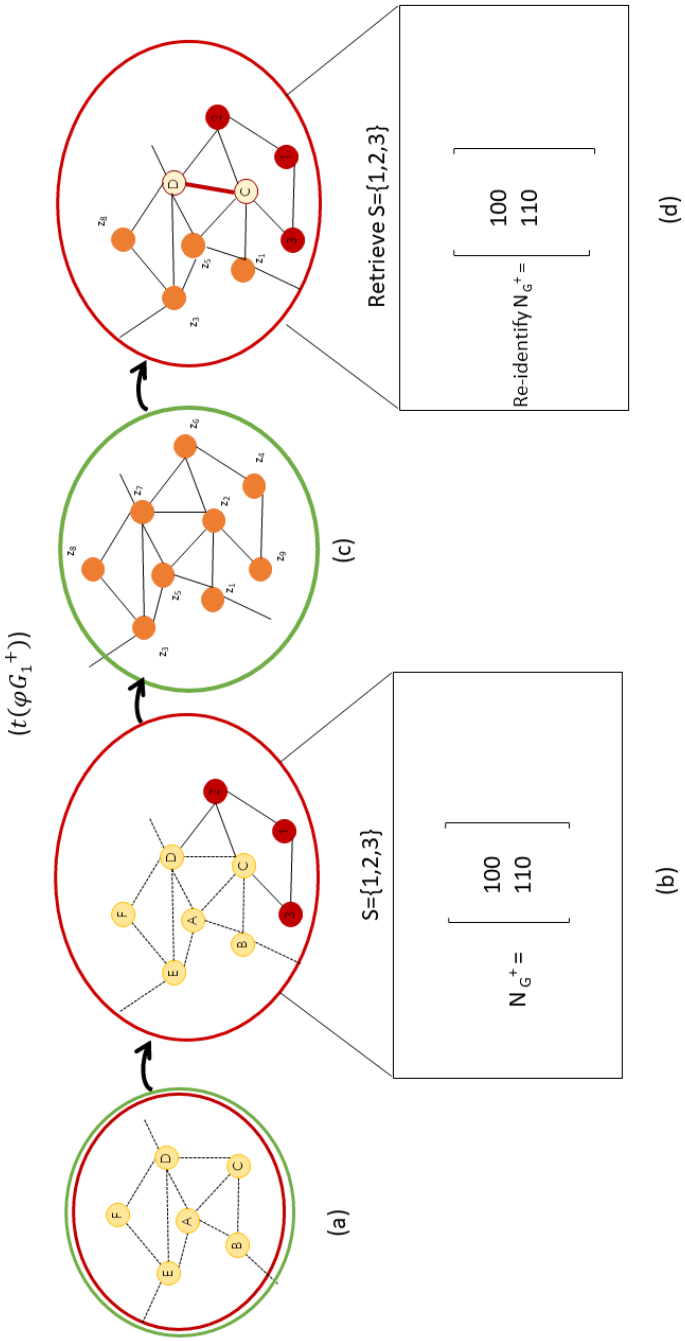


Figure 4.3: Detailed process of the attack in 1^{st} snapshot

This part is illustrated as the transformation of the graph before release, where the noise in the graph might affect the attacker's subgraph. In this step of the game the defender will keep track of the original graph state and the perturbed graph state. The last part of the first cycle of this game is the process of re-identification, which is the main action of the attacker. In the Figure 4.3(d) we display the process of retrieving the sybil nodes and also re-identifying the fingerprints. In order to have success we have to consider that the attacker successfully and effectively determines the real identities of pseudonymized victims [17]. After creating the attacker's subgraph and having unique fingerprints for the victims, the attacker will proceed by attacking the pseudonymized released graph. Therefore, upon retrieving its sybil subgraph and re-identifiyng the victims, the attacker might be able to find out who hides behind the pseudonymized nodes. In this example, he will have the information that Carol and Dan are friends.
Furthermore, the reason of creating this multi-stage adversarial model is the ambition of the attacker to use the adaptation strategies, in order to learn from previous re-identification errors and correct them to retrieve as much information as it can from the social network that was released. Since we simulate a periodically released graph scenario, we describe a model where the adversary can use the adaptation strategies in a repetitive form for as long as there are new snapshot releases. Between releases of snapshots, the attacker adds sybil nodes if necessary in the social network and establishes connections with new targeted victims, as depicted in Figure 4.4(a), where it has been illustrated in details what happens at the specific step for the second release of the graph. The new sybil account is visualized as well with dark red color and the edges known to the adversary are solid, since he/she manages to enforce the path between the sybil nodes. As depicted in Figure 4.4 (b) we can see that the attacker has managed to add more sybil accounts in the original graph and grow its sybil node set, which creates connections with other victims. In this case, the attacker has now an evolved sybil set and also modified fingerprints, in the context that the old fingerprints are updated considering the number of newly added sybils and targeted victims.

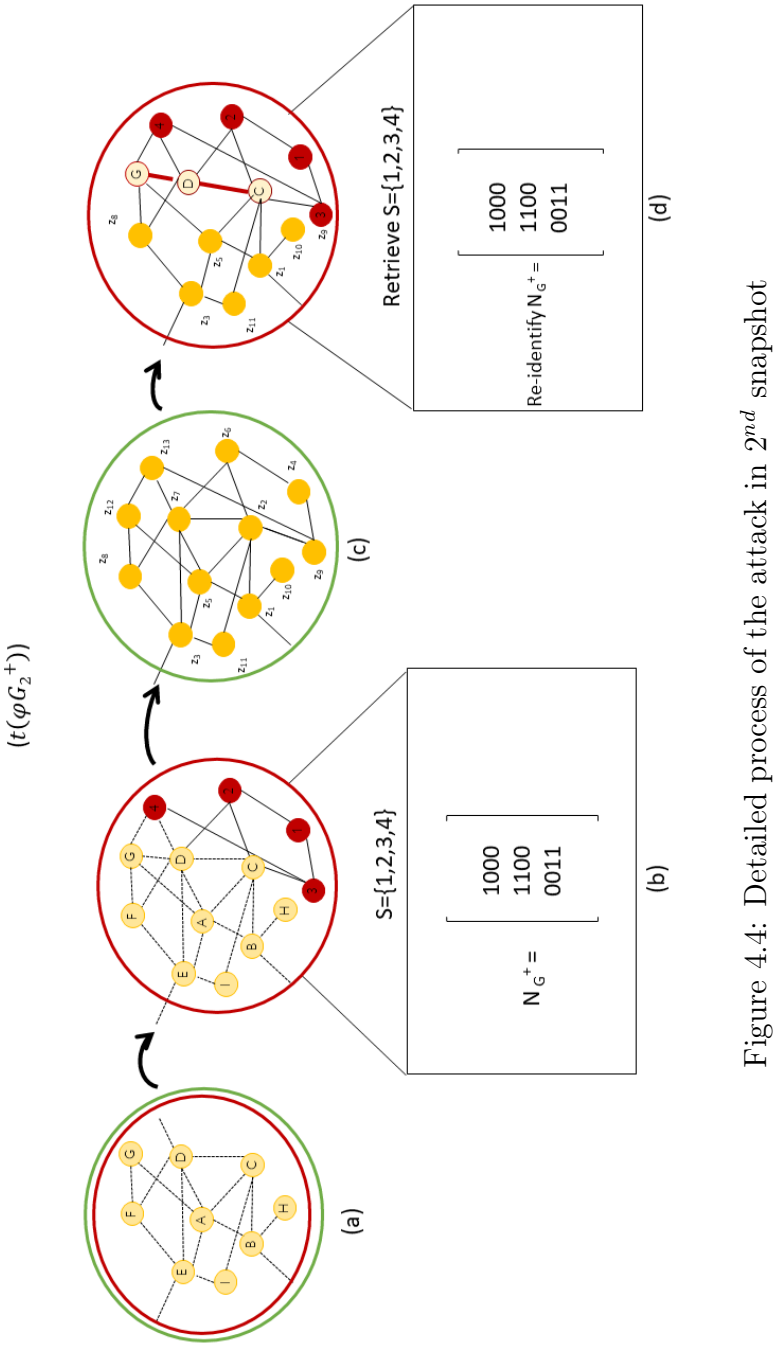


Figure 4.4: Detailed process of the attack in 2^{nd} snapshot

Comparing the sybil graphs in two snapshots, it is noticeable that the fingerprints illustrated in Figure 4.3(b) are shorter than the fingerprints in Figure 4.4(b) meaning that a new sybil has been added, hence the length of the fingerprints in snapshot 2. Figure 4.4(c) shows that the nodes are perturbed yet again, where the defender transforms the graph on top of the transformation in the previous snapshot. The defender assumes that this action makes it harder for the attacker to re-identify the fingerprints in the next attack. The evolved graph now contains more information that would be useful for the attacker. Connections between users, are a type of information that is hard to get from this social graph seeing that the identities of the vertices are hidden, the attacker now has more knowledge with the each release of the graph. Assuming that the pseudonymized nodes are

labeled consistently throughout the releases of the snapshots, the attacker can use its adaptation strategies iteratively for each snapshot. In such a way, the attacker adapts to the dynamical environment, by using the information from previously released snapshots and re-identifying users along the sequence of snapshots. This example is suitable to explain how a re-identification active attack occurs in a periodically released graph. Knowing that Carol and Dan are connected from the previous release, the attacker has the chance to analyze the social network that its now in their hands and starts by re-identifying its own sybil nodes and the victims as shown in Figure 4.4(d). This allows the attacker to suspect that Dan has another friend connection with Greg which is depicted as the capital letter G in Figure 4.4(b). The attacker understands that the existing connection is correct, since in the previous snapshot Dan had only one connection that the adversary had access to and now Greg was a targeted victim as well. This leads to the getting the information about the degrees of targeted nodes and the connections amongst them from one snapshot to the other. In an iterative way, the attacker can continue applying the adaptation strategies which take different forms for the sequence of snapshots, which make him capable of harvesting useful information.

Since there is no published work on active attacks in dynamic graphs, we devised some changes in the original attack and robust attack [2, 17], to introduce attack strategies that work in periodically released graphs, benefiting the attacker from snapshot to the other. This pushed us to creating more actions for the game, which will be described in the following section.

4.1.3 Strengthening the adversary capabilities

According Figure 4.2, we would like to structure the attack by introducing the actions on each of the snapshots and therefore explaining the stages of the extended attacker-defender game. Assuming to have a sequence of snapshots of graph $G_i = (V_i, E_i)$, where i is the snapshot identifier we formalize the stages of our proposed attacker-defender game.

1. Attacker subgraph extension
   As presented in Chapter 3, given a social graph $G_i = (V_i, E_i)$ where i is the snapshot identifier, the attacker constructs the sybil network using the same strategy as the widely-studied instance of the original active attack. The walk based attack allows the attacker to insert new nodes $S = \{x_1, x_2, ..., x_n\}$ into G_i, resulting in the graph $G_i^+ = V_i \cup X_i$,. The attacker then chooses an arbitrary set $Y_i = \{y_1, y_2, ...y_m\}$ as users in G_i as the target of attack. For creating the internal links amongst sybils, in the sybil network, we enforce a path amongst them. For example, our sybil set $S = \{1, 2, 3, 4, ...n\}$ will have a path such as $\{(1, 2), (2, 3), (3, 4), ..., (n - 1, n)\}$ and so on. Furthermore, the relations between other sybil nodes remain random with a probability of 0.5.
   The adaptation strategies play a huge part in extending the attacker subgraph. The proposed strategies involve addition of new sybil nodes and

targeting new victims after each release of the graph. These methods will be described as adaptation strategies in more details in this section.

(a) Extension of sybil nodes
The previously known set of sybils $S = \{x_1, x_2, ..., x_{|S|}\}$ will now have new additions to the set after each snapshot $S = \{x_1, x_2, ..., x_{|S|}\} \cup S'\{x'_1, x'_2.., x'_n\}$ and as a result S_i represents the sybil set in the i^{th} snapshot. In this stage, the attacker has the opportunity to choose how the nodes will be added, in a non-random way or random way. We start by introducing the non-random approach which is infiltrating in the graph only *log n* sybils. This approach makes sense if we assume that the data owner is using some sort of sybil defense mechanism and the goal is to keep the subgraph undetected so the attacker succeeds. In the random approach, the attacker can not control the number of sybil nodes that are added, he knows that randomly it will either be 1 new sybil node or 0. Even using the randomness approach it makes sense, since we do not escalate the number of sybil nodes.

These adaption strategies will help extending and strengthening the attackers capabilities of being more successful in a dynamic environment. Assuming that the graph grows over time, having more sybil in the network gives the upper hand to the attacker.

(b) Modifying the Fingerprints
This strategy involves two procedures, one being the targeting of the new victims and the other the modification of the existing unique fingerprints and respectfully constructing the new victims unique fingerprint. We create the new set of fingerprints for the newly targeted victims in our sybil network, by using the original walk based strategy[1] which is by generating random internal edges amongst the sybil set and the victim set.
In the same manner as in the previous step, we consider both non-random and random approaches in this adaptation strategy. The non-random approach that we consider as a adaptation strategy is called the hardest to re-identify victim. This proposed adaptation strategy method strengthen the attacker's capabilities to learn which of the victims are the hardest to re-identify. The attacker then has the opportunity to flip some connections to the specific fingerprint. Using the random approach, the idea of this adaptation strategy is to modify randomly one or three of the old fingerprints in case a new sybil is added and to add the necessary values in the unique fingerprint of each victim depending on the number of sybil nodes. Moreover, it might be the case that the perturbation action done by the attacker causes some of the connections to fail between the sybil nodes and the victims. The victim will have a uniquely modified fingerprint before the next release. All of the fingerprints will be represented in such a way that each victim has a unique fingerprint and the same fingerprint will not be allowed in the set of fingerprints. In this manner, the attacker can benefit from the

previous snapshot and knowing which victims he failed to re-identify.

We justify using both approaches for the assessment of the proposed attack strategies.

2. Re-identification

As described in the preliminaries, the attacker will execute its re-identification attack after obtaining the releases of the graph. Working in a dynamic graph the attacker will receive one graph at the time, assuming that is $G_i = (V_i, E_i)$. The attacker then computes the set of similar candidate sub-graphs while comparing them with the original attacker subgraph. Assuming that the adversary determines each fingerprint N_i with $i \in \{1, ..., m\}$ in the candidate set of vertices whose fingerprint to original graph are determined by N_i. For this stage we developed one additional method which will strengthen the capabilities of the attacker where we use the intersection strategy. The idea of presenting such an approach is to use only the intersected candidates of subgraphs that are similar to original sybil network. Recalling that the nodes of the graph are labeled, this strategy proves to be effective, when from snapshot to snapshot we consider only the intersected subgraphs that we found since the labels do not change, instead of recomputing all the potential subgraphs. In this way, we propose that this strengthens the attackers capabilities to conduct a successful attack.

- Attacker subgraph retrieval
  As presented in the preliminary the attacker starts by re-identifying its sybil nodes, by following the path of inter-sybil connections, he manages to find the newly added sybil nodes.

- Fingerprint matching.
  The attacker determines the real identities pseudonyms of the victims that map to the unique fingerprint, by re-identifying the unique fingerprint of each victim.!!

As mentioned above, the attacker can use all of the adaptation strategies during the game. The attack strategies that were devised in this work are separated in 2 categories:

1. Sybil Addition
   We always consider the non-random approach and the random approach.

   1.1. *Log n* sybil nodes
   If the attacker chooses to use this evolve method, it will add sybil nodes if and only if the number of vertices in the newly released graph has grown plenty such as *log n* value of the sybil nodes should change. As an example, we can say that a graph in the first release had 100 vertices which leads to having 7 sybil nodes, but if the graph evolved from one snapshot to the other for 100 extra nodes leading to a 200 vertices graph now the sybil node set changes from 7 sybils to 8.

   1.2. Random sybil addition
   The attacker when choosing to add random sybils assumes that the graph evolved over time and this method constitutes adding 1 or 0 sybils randomly. Since we have to assume that the defender might employ the sybil defense mechanism we keep the number of sybil nodes as low as possible from one release to another.

2. Targeting new victims and modifying fingerprints There are two issues that we need to address at this point, one of them being the targeting of new victims and the other is modifying the existing fingerprints. In these adaptation strategy, we offer the ability to grow the victim set, by randomly targeting new victims in the graph. Furthermore, we offer two approaches of modification, non-random and random.

   2.1. Hardest to re-identify victim
   Its main purpose is in modifying the victims fingerprint that proved to be the hardest to find in the re-identification stage. While doing the re-identification, the attacker might stumble upon a fingerprint which is similar to many other fingerprints. The adaption strategy will randomly select the most confused set of fingerprint and modify that fingerprint by randomly flipping the edges between the sybil and the attacker. This proves to be efficient since the attacker can keep track of which fingerprint was hard to find in previous releases and modify it so it will gain in resilience. This proposed adaptation strategy method strengthens the attacker's capabilities to learn which of the victims are the hardest to re-identify. The attacker then has the opportunity to flip some connections to the specific fingerprint. If there is a scenario where multiple fingerprints are producing the same success probability during re-identification of the victims, then the attacker modifies all of them.

   2.2. Random Victim fingerprint modification
   This method allows the attacker to randomly modify 1 to 3 fingerprints. In this particular methodology for each release of the graph, when using this adaptation strategy the attacker after randomly selecting the

fingerprints each of them has then its connections randomly flipped. Meaning that if there was a connection with one of the sybils and that specific victim we remove it, if not then we add one edge. If an attacker sees that he is loosing resilience during an attack, by using this method he can randomly benefit by choosing a fingerprint to modify. Considering that the snapshot is perturbed, it might be the case that the chosen fingerprint is exactly the one that was touched by the perturbation. Modifying the fingerprint with new connections and restarting the re-identification attack on the next release benefits the attacker, by giving him a fresh start with new fingerprints to re-identify.

3. The strategy of using intersection of candidate subgraphs
   The idea of the proposed strategy is straight forward, every time that we might have multiple candidates subgraphs to compare along the original attacker subgraph, this evolve method will only take the intersection of the candidate subgraphs. Meaning that with each release, the attacker remembers the set of candidate subgraphs that were re-identified during the previous snapshot and intersects those subgraphs with the newly found candidate subgraphs in the new snapshot.

Chapter 5

Experiments

In this chapter, we will firstly introduce how the periodically released graph snapshots are constructed. The experiments conducted show a probability of success with each release of the graph depending on the attacker's and defender's actions. We model an attacker that uses the proposed adaptation strategies and the simple defender that perturbs the graph by adding noise. The results show a gain in resiliency of the proposed evolve methods which are used in all of the proposed attacks constructed from [2, 17]. Moreover, the comparison between all attack strategies is analyzed. The experiments were performed on the, on the iris cluster [1] of the HPC platform of the University of Luxembourg [29]. The source-code is provided in the link presented in the Appendix 6.3.

5.1 Periodically released dynamic graph simulators

5.1.1 Network Growth

In this work we construct an architecture of periodically released graphs that comes closer to a dynamic graph model. The simulators introduced provide the construction of dynamic graph models. The existing graph model wrappers offered by the library known as JgraphT [20], work only for static graph models. Therefore, based on this we developed network simulators which allow the graph to be periodically released. In a dynamic environment, each release of the network after a certain period of time or after the evolution of the graph in a specific way is a snapshot of the graph. A snapshot is an image of the graph at the specific period of time in the time line or after some modifications in the original graph. The snapshots are released in a periodical manner to simulate as such the multiple releases of the graph as seen fit by the data owner.

Since the goal is to simulate a dynamic framework, the implementation of the

[1] https://hpc.uni.lu/systems/iris

wrappers included creating more effective methods with regards of adding new vertices and edges. Remembering the additions for each snapshot is another property provided by the simulators, so that the graph would grow in a period of time on top of the modification done in the previous snapshot. The implementation of edge and vertex addition methods is based on the specific generative graph model, to evolve a simple undirected graph. The generative graph models that were constructed for this research are :

- Barabasi-Albert graph model
- Barabasi-Albert Forest graph model
- Scale free graph model

For each of before mentioned models, the periodical released snapshots of the graph will have newly implemented addition methods for vertices and edges. Even though the experiments will be conducted only on the generative graph model known as Barabasi-Albert, the other two can be seen as future work analysis. Barabasi-Albert graph model is a preferential attachment model, which constructs the graph that for each newly added node in the graph it attaches it to the node with the biggest degree of nodes. When constructing a graph the parameters that we need to input are the order of the graph and the number of edges per each node added during the growth of the network. The number of edges per node will be used as a parameter in our experiments. The introduced simulators construct a graph which will have labeled nodes and these nodes are consistent between each snapshot of the graph. After each snapshot is released the graph will grow either by adding vertices or by adding new edges to the existing graph. This property expresses the addition of vertices in the graph, if a graph is 200 vertices and for each snapshot we want to simulate a growth of the graph by adding 50 new vertices, then this method will makes it possible to add 50 new vertices every time we release a snapshot of the graph. In the other hand, the growth of the graph can be done by adding edges as well, in the particular case the addition of vertices would be done depending on the parameters that we select beforehand in the construction of the graph. For example, if the graph has 200 vertices and 800 edges and we want to grow the graph for 200 more edges, the method will take the parameter that is the number of edges per node and make the calculation for how many new vertices we should add so that we reach the number of modifications in the edge set.

Considering this experimental setup, we model the attacker with adoption strategies. In the other hand we model a simple defender because it goes beyond the scope of this thesis, which emphasizes the attacker's capabilities.

5.1.2 Simulating attacker actions

Recalling the attackers actions, we separate the additional actions in two categories.

1. Non-random adaptation strategies
   Using these strategies, the adversary can have control on the evolution of its actions. The adaptation strategies that are suitable are:

   - Modifying the fingerprint of the hardest to re-identify victim
     By choosing to play this strategy, the attacker will benefit from the background knowledge that he has gained from the previous released snapshot. In this scenario, the attacker will get to see which of the victims is creating the confusion set and he will choose the fingerprint with the most confused set to modify it. The modification of the hardest to find victim is done by modifying the fingerprint and achieving the optimal result for re-identifying the specific victim.

   - Adding $log\ n$ sybil nodes
     By choosing to add sybil nodes on the optimal threshold of having $log\ n$ sybil nodes on the graph, in essence the attacker can somehow control of having his sybil network undetected under the assumption that if the data owner uses any kind of sybil defense mechanism the threshold is $log\ n$.

2. Random adaptation strategies
   In the same manner as creating the first set of strategies, we thought it would be interesting to see how the attack strategies would work if we did not control them.

   - Targeting new victims and modifying the fingerprints
     When choosing this strategy, the attacker will randomly select one of the existing nodes on the graph as a new victim and tries to establish connections between its sybil nodes and the new victim. Furthermore, after the new victim is targeted which means that the victim set now has grown, the attacker constructs the unique fingerprint of the newly targeted victim which will be added on the list of its existing fingerprints.

   - Random sybil addition
     By choosing to add random sybils, the attacker can add 0 or 1 new sybil node. This restriction is done based on the assumption that there might be a sybil defense mechanism and the attacker's subgraph should remain undetected. If a new sybil is added, then the developed method will also modify all of the existing fingerprints by randomly adding a connection between the specific victim and the newly added sybil.

3. Intersection of candidate subgraphs comparable with the original attacker subgraph
   The graph is constructed in such a way that the nodes labels are consistent throughout the releases of snapshots. Using this approach, the attacker will be able to find the same candidate subgraphs as in the previous release.

One of the main reasons to construct such a framework, is to compare the outcome of all the proposed approaches when the attacks are conducted.

5.1.3 Simulating defender actions

In the simulated network, we model a simple defender which adds perturbation just before the release of the graph. Since we modeled a dynamic framework, the defender will apply perturbation iteratively before each snapshot of the graph is released. The proposed action in this dynamic framework is adding perturbation to each snapshot release of the graph, which consists in accumulated noise from the sequence of snapshots. In addition, the defender publishes a noisy snapshot so the attacker will have a harder challenge trying to attack it. The random perturbation methods that we implemented depends on the choice of the defender whether the noise should be added based on the edge set or vertex set. In the conducted experiments, assuming n is the number of vertices in the graph, respectfully the number of edges in the graph and having p as an input percentage, the random perturbation formula is $p \cdot n$ which gives the number of flips that will occur during one snapshot. By number of flips we mean choosing randomly a vertex pair (v, u) and checking if there is an edge we remove it, if there is not an edge we add it. Each selection of the pair and the action that comes afterwards it is considered a flip. One being the random way of adding vertices/edges or the evolution of the graph through out a time line and the second one random perturbation, where we randomly remove edges from the graph. We defined these two approaches as the baseline of constructing the graphs that will be used for our synthetic collection of graphs.

Since we simulate dynamic graphs, with each snapshot the defender adds perturbation in an iterative way. We create multiple releases of the same graphs but in different period as seen in 4.2, the main idea of the perturbation method is that it remembers the changes that occur after each release. In this case the defender on the first snapshot might add 0.5% of noise and in the second snapshot proceeds with the same approach, what happens in the background is that in the second snapshot the noise added will be on top of the release of the first perturbed graph and the evolved graph, in conclusion it means having another 0.5% which adds up to 1% noise when the second snapshots is released. In such a manner, the probability of affecting the sybil network grows more with every snapshot that will be released. During our experiments each graph will be evolving over a period of time, so a graph of 100 nodes after 5 releases might be a graph of 500 nodes and the noise in the graph might be 5%, depending the values that we choose to evolve the graph. Based on the results, we can say that while using the perturbation, there is a good chance that the sybil network will have some changes, which leads to the attacker loosing the ability to perform the attack in a successful manner. In Table 5.1, we present the values that were used for different size graphs.

5.2 Datasets and experimental settings

As a dataset we created a collection of randomly generated synthetic graphs, where we used the Barabasi-Albert graph model which was adapted to work in a dynamic framework of graph releases. This collection is composed of 60,000 graphs, 1000

runs for each attack strategy and for edge density values, m=4,12,20. Each graph at the beginning has respectively 100 and 200 nodes, and its edge set depends on the edge density values shown in table 5.1. 1 run consists of constructing one graph of n order, with m edges per node, by periodically growing the graph for each snapshot and releasing the graph 5 times. In each release, the attacker performs the re-identification active attack in a perturbed graph. During this run of the experiment in 1 graph, the graph grows either for 5% of the vertex set or 10%, depending on the choice. The graph then is released 5 times with a perturbation percentage of 0.5%. When accumulated the final released snapshot will have a perturbation noise of 2.5% and 25% growth from the first released snapshot.

Runs	Graph Vertex Set	Edge Density	Noise Percentage	Growth percentage per release
1000	100	$\{4, 12, 20\}$	$\{0.5\%, 1\%\}$	$\{5\%, 10\%\}$
	200	$\{4, 12, 20\}$	$\{0.5\%, 1\%\}$	$\{5\%, 10\%\}$

Table 5.1: Properties of graphs in the conducted experiments

As discussed in [2, 17], for a graph that has n vertices, it will suffice to have *log* n sybil nodes for being able to compromise any number of victims. This was concluded from multiple work since, the sybil defence mechanisms tend to avoid having more than *log* n sybil nodes and at some point it will start removing the malicious nodes[31, 23, 34, 33]. When evaluating each attack strategy on the collection of randomly generated graphs, we use *log* n sybil nodes when creating the sybil network, respectively depending on the graph vertex set. In table 5.2 , we present the main attacks and their methodologies. Attack 1 is the attack

Attack	Description of the Attack
1	Original Walk-Based Sybil Retrieval + Exactly Matched Fingerprints
2	Robust Sybil Retrieval + Exactly Matched Fingerprints
3	Robust Sybil Retrieval + Approximately Matched Fingerprints

Table 5.2: Attacks used in experiments and their properties

presented by Backstrom et al. in [2], whereas attack 2 is a combination of the proposed attack in [17] and the original walk based attack and the attack 3 is the pure robust attack from [17], as seen previously in the Chapter 2 and 3.

In this section, we will present thoroughly the attacks that were conducted in 1000 graphs of each setting. Firstly on table 5.3, you will see 5 different attack strategies. Each attack has different evolve methods, such as adding new victims, using the same number of victims but modifying the hardest to find victim, adding random sybils and also keeping the number of sybils to *log n*. For every graph in the collection, after simulating the attacker subgraph creation stage of each attack, and the perturbation performed by the defender, we calculate the probability of success for each attack strategy with different graph settings and analyze the outcome.

1. Growing the graph by adding 5% of vertices in G_i^+ of the total number of vertices.

2. Growing the graph by adding 10% of vertices in G_i^+ of the total number of vertices.

3. Randomly flipping the graph edges for 0.5% after each snapshot, meaning that after 5 releases of the social graph the noise in the graph will be 2.5%. Each flip consists in randomly selecting a pair of vertices $u_i, v_i \in V_{Gi^+}$ in snapshot i, if there is an edge between the pair we remove it, if there is not we add the edge between the specific pair. Since G_i^+ can have order n, where n is 100 or 200, we will have the perturbation formula which is 0.05 * n.

4. Randomly flipping the graph edges for 1% after each snapshot, meaning that after 5 releases of the social graph the noise in the graph will be 5%.

Attack	Description of the Attack strategies
1	Targeting new random victims + log N sybils if needed
2	Targeting new random victims + random sybils
3	Modifying hardest to find victim and adding log N sybils if needed
4	Modifying hardest to find victim and adding random sybils
5	Without any adaptation strategies

Table 5.3: Attack strategies based on the adaptation strategies

In the experiments we have a combination of the above presented graph settings, meaning that we have collected results with 5% addition of vertices and 0.5% noise, respectively 1%. Furthermore, we conducted more experiments combining the 1% perturbation level and 10% addition of vertices. The results of the latter are presented in Appendix 6.3.

As discussed in the original work [2], for compromising victims, it suffices to insert *log n* sybils. When evaluating each for the attack strategies in the collection of synthetic periodically released graphs, we initially infiltrate *log n* sybil node in each graph creating a sybil network. Given the set S of sybil nodes, the original attack randomly creates connection between sybils and victims, creating unique fingerprints for each victim. In the case of the robust attack, in this work we use the random generated fingerprints rather than the uniformly distributed fingerprints. The threshold of maximum distance used in these experiment between the fingerprints in these experiments is $\beta = \theta = 4$ [17]. We use a smaller threshold since the robust sybil network retrieval works faster on bigger graphs, recalling as such that the simulation of multiple snapshots will lead on the the initial graph increasing the order n on all periodically released snapshots. The other attacks are conducted by creating the fingerprints and sybils in the same manner that was presented in the original paper [2]. Finally, we compute the probability of success of the re-identification stage for each variant of the perturbed graph.

5.3 Results and Discussion

In this section, we present our results and show how the proposed adaptation strategies increase the adversary's capabilities in a dynamic framework of graph releases. The experiments are conducted by applying three attacks as described in 5.2 and applying the adaptation strategies presented in 5.3. These attacks are conducted in graph of order n initially, where as after each release of the snapshot the order will be larger. We consider that the adversary succeeds if all the vertices in the set of victims are correctly re-identified. Therefore, the probability of the success used for these experiments and proposed in [17] is:

$$Pr = \begin{cases} \frac{\sum_{x \in \chi} px}{|\chi|}, & \text{if } \chi \neq \emptyset. \\ 0, & \text{otherwise.} \end{cases} \tag{5.1}$$

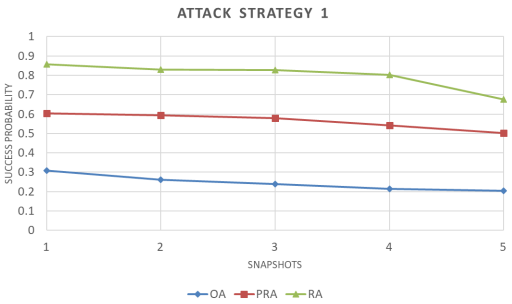
5.3.1 Success probability of all the attacks on periodically released graphs of order n

Graphs of 100 vertices with 4 edges per node and using 0.5% noise and 5% growth of the graph

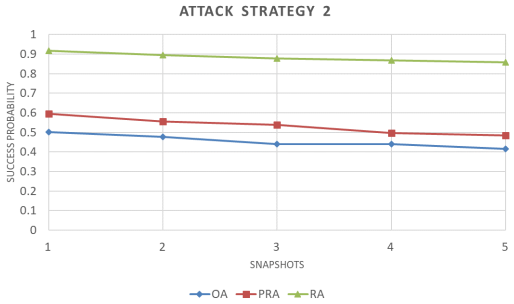
In this part we start by visualizing our 5 attack strategies applied by all of the attacks that are proposed in Table 5.3. We introduce that for each of the attacks we used short presentation of the actual name in the graphs. OA is the original walk based attack, PR is the partial robust attack and RA is the robust attack. Furthermore, the "-INT" shows that the attacks use one of the proposed evolve methods which is the intersection.

In the upcoming results, the number of snapshots per graph will be 5, meaning that one graph will be evolved 5 times and perturbed if the defender chooses to for 5 times or less. Figure 5.1, Figure 5.2 show the success probabilities of the three attacks on the perturbed graphs obtained by applying the strategies from (a) to (e) in a Barabasi-Albert graph with 100 initial nodes having 4 edges for each added node. The percentage of the noise was 0.5% for snapshot, meanwhile the graph grew 5% of the size of the vertex set for every snapshot. For this experiments, we use the attack strategies as presented in Table 5.3.

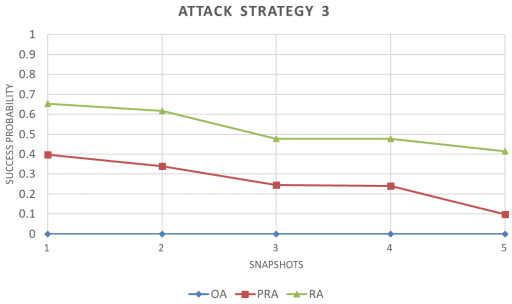
In Figure 5.1 from (a) to (d), we have illustrated the attack strategies that apply the proposed adaptation strategies, whereas in (e) the attack strategy does not use any adaptation strategy. Figure 5.1(b) shows that the robust attack has a consistently larger probability of success than the original-walk based attack and partial robust attack and also as the attack strategy proposed in (e). Even though, the other attack strategies performed fairly good, we have to highlight the attack strategy 2 and attack strategy 3. A fact worth pointing out, is that the robust attack using the adaptation strategies it is completely effective against the perturbation of the graph. Furthermore we continue with the comparison of



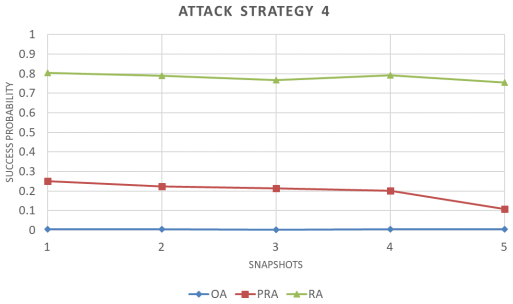
(a) Attack Strategy 1



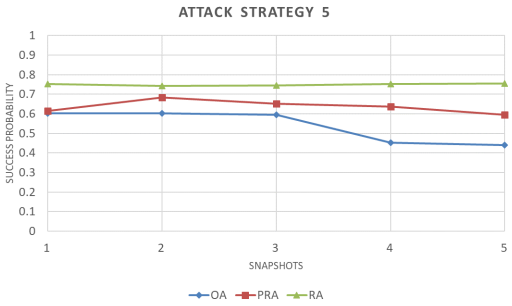
(b) Attack Strategy 2



(c) Attack Strategy 3



(d) Attack Strategy 4

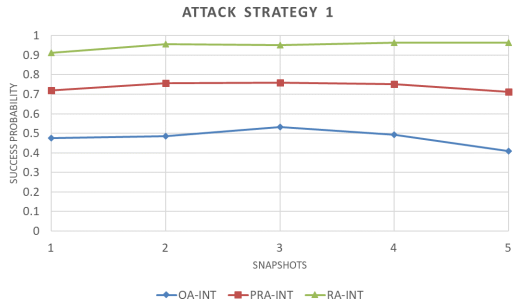


(e) Attack Strategy 5

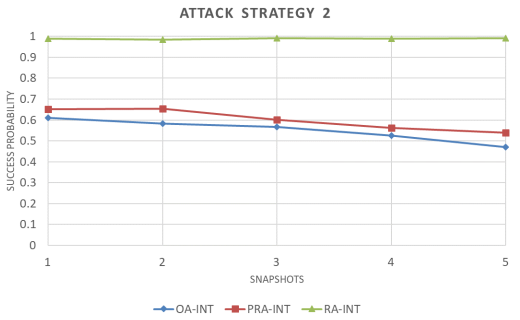
Figure 5.1: The average probability for all attacks without intersection of 1000 runs on graphs with 100 nodes, 4 edges per node, 0.5% noise and 5% evolve of the graph

the non-random approaches against random approaches. When analyzing the attack strategies (a) against (c) where (a) is the random approach and (c) is the non-random approach, it is noticeable that all of the attacks start to degrade on the attack strategy 3. The adaptation strategies used in (a) show better results that the adaptation strategies used in (c). The adaptation strategy used in (c) can take longer to perform the attack, simulating a real attacker, we deliberately consider the attack to fail if it takes a long time to compute, thus the difference in the result. In the other hand, if we compare (b) against (d), (b) being the random approach and (d) the non-random approach, we can still see a more efficient set of attacks in the random approach. If we take a look at Figure 5.1 (a),(b) and (d) comparing them with (e), the robust attack gains resilience and we also notice a growth on the probability success. With that in mind, we came to a conclusion

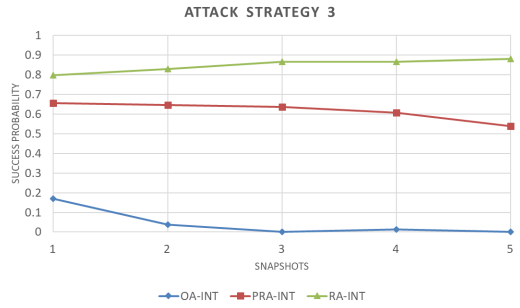
the the evolve methods used in the before mentioned strategies actually help the attacker from snapshot to snapshot to be more efficient and powerful against the defender.



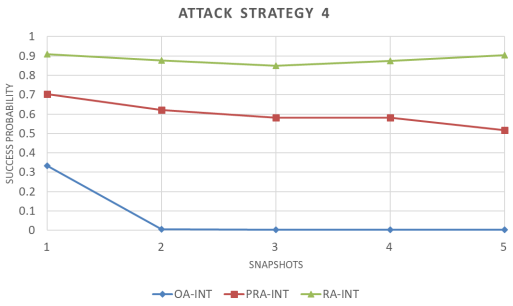
(a) Attack Strategy 1



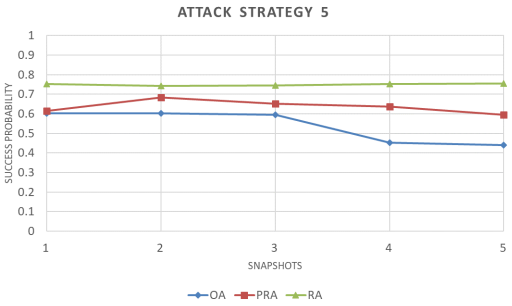
(b) Attack Strategy 2



(c) Attack Strategy 3



(d) Attack Strategy 4



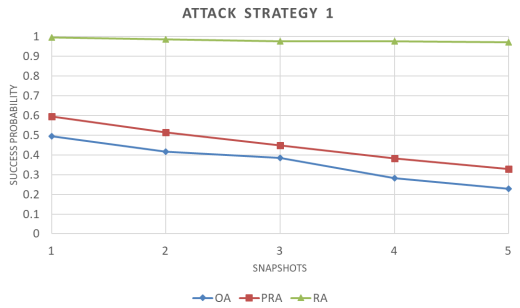
(e) Attack Strategy 5

Figure 5.2: The average probability for all attacks with intersection of 1000 runs on graphs with 100 nodes, 4 edges per node, 0.5% noise and 5% evolve of the graph

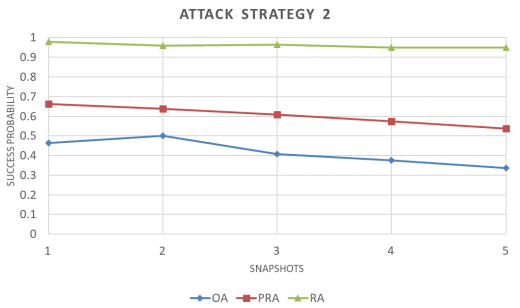
In Figure 5.2 from (a) to (e) we demonstrate the success probability of each of the attack strategies using the evolve method of intersection. We observe a gain in resilience in most of the attacks in Figure 5.2 from (a) to (d). If we compare Figure 5.1 from (a) to (d) with results presented in Figure 5.2 (a) to (d), we can clearly see an obvious change. The fact that along with the adaptation strategies we apply the intersection strategy we can observe a significant positive change that supports our argument that the adaptation strategies strengthen the adversary's capabilities of succeeding. An interesting result is the one shown in Figure 5.2, where the adaption strategy to modify the fingerprint of the hardest to re-identify victim shows a growth of probability in each release. This represents the fact that the attacker corrects its errors from the previous re-identifying process and improves them in the new release of the graph, by re-identifying the hardest to re-identify victim in the previous snapshot. If we make a comparison between the attacks we can say that the robust attack once more proves to be the most efficient one compared to the other two attacks. As for the original walk based attack in all strategies we can state that if the perturbation touched the sybil nodes the attack failed immediately or in the other hand it will perform well but nothing compared to the other two attacks. Furthermore, for the partial robust attack we see that the adaptation strategy using intersection slightly make the attacker more successful than without using any of the adaptation strategy methods. Another observation is that for all the attack strategies in these graph parameters the number of edges per node plays a role as well. Therefore, the experiments will also be conducted on graphs of 100 vertices with different number of edges per node.

Graphs of 100 vertices with 12 edges per node and using 0.5% noise and 5% growth of the graph

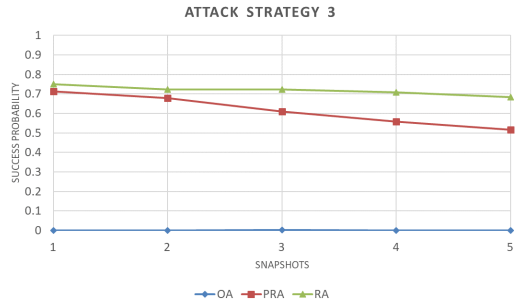
In the following section, we will analyze the results that were conducted in a 100 node graph with 12 edges per node. In Figure 5.3, analyzing the attack strategies from (a) to (e), we can see a consistency of the robust attack in each released snapshot.We highlight the attack strategy in (a) and (b), where we observe the robust attack being consistent and the success probability for 5 releases of the graphs proves that the adaptation strategies, strengthen the adversary's capabilities. We would like to emphasize on the fact that all attacks in Figure 5.3(e) gradually start to fail after the 5^{th} snapshot. In Figure 5.3(c) and (d) we can observe that the partial robust attack and the original fail immediately from the first snapshot. We assume that the reason is that the perturbation caused by the defender touched the connection of the sybil nodes. The observation on the random and non-random approach, points out that the non-random approaches (c) and (d) do not give the optimal results when comparing them with the random approach results. This behaviour is noticeable when re-identifying the hardest to find victim, sometimes it takes a long time to find the victim and as such we consider that the attack has failed, which leaves room as future work to optimize it.



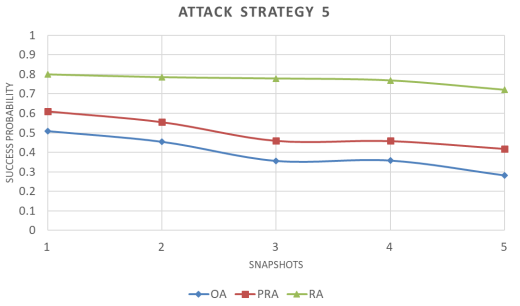
(a) Attack Strategy 1



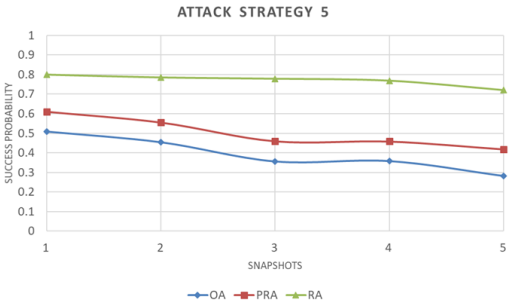
(b) Attack Strategy 2



(c) Attack Strategy 3



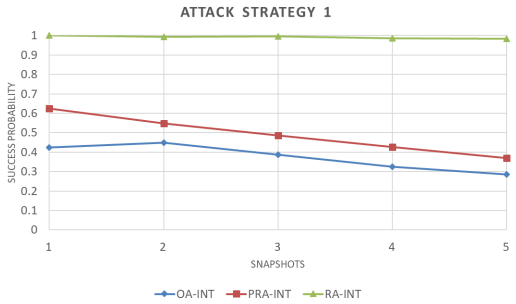
(d) Attack Strategy 4



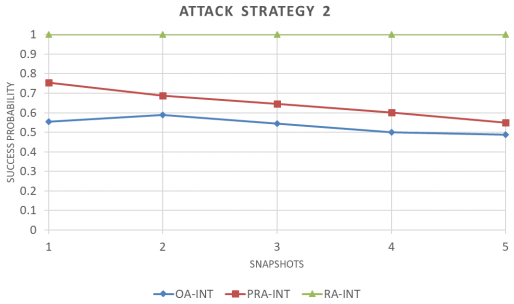
(e) Attack Strategy 5

Figure 5.3: The average probability for all attacks without intersection of 1000 runs on graphs with 100 nodes, 12 edges per node, 0.5% noise and 5% evolve of the graph

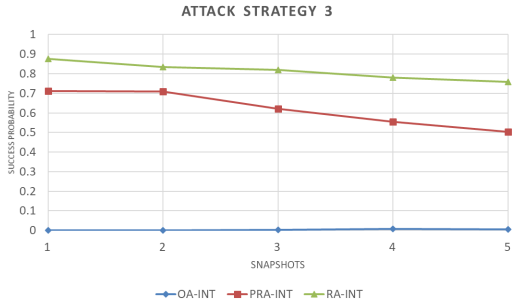
We start by comparing the adaptation strategies using intersection without the ones that do not use intersection. As depicted in Figure 5.4, all the adaptation strategies performed significantly better than the ones illustrated in 5.3. We highlight the results of (a),(b) and (d), which still proves that using the adaptation strategies the attacker shows superiority against the defender, by being able to infiltrate the nodes and create the fingerprints and in between the snapshots, re-identify the attacker's subgraph. The Figure 5.4 (c) shows that the re-identifying the hardest to find victim together with the intersection strategy still performs better than without using the intersection as was depicted in Figure 5.1(c).



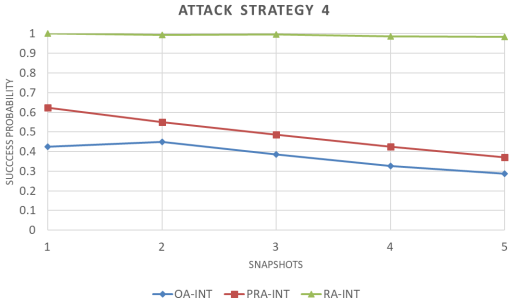
(a) Attack Strategy 1



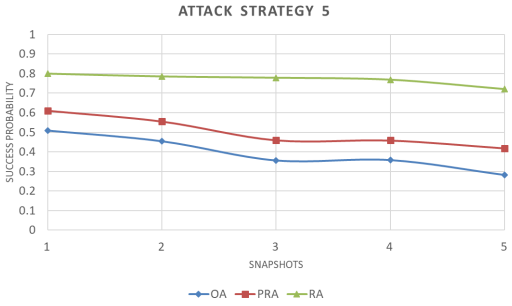
(b) Attack Strategy 2



(c) Attack Strategy 3



(d) Attack Strategy 4



(e) Attack Strategy 5

Figure 5.4: The average probability for all attacks with intersection for 1000 runs on graphs with 100 nodes, 12 edges per node, 0.5% noise and 5% evolve of the graph

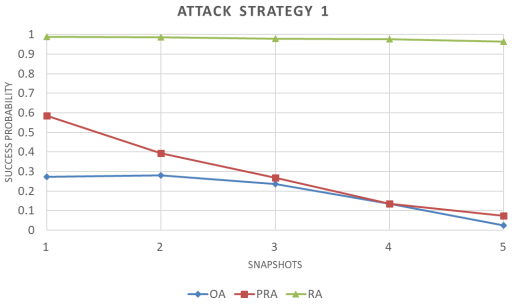
As we mentioned before, the robust attack looses its consistency, when the re-identification attack takes to long to compute. Furthermore, we noticed how the original attack in the Figure 5.4 (d) produced better results compared with Figure 5.3 (d) where the original attack was thwarted.

Also from this experiment, we learn that the number of edges per node when constructing the graph plays a huge part, specifically for the original walk-based attack and the partially robust attack. One can conclude that the attack strategies that use random-approach of adaptation strategies are somehow more successful than the ones that do not. This leaves room for improvement and will be considered as future work.

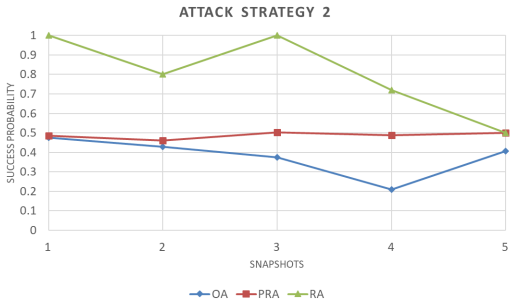
Graphs of 100 vertices with 20 edges per node and using 0.5% noise and 5% growth of the graph

As the before mentioned experiments, we conduct the attacks of all adaptation strategies in graphs with 100 nodes and 20 edges per node by introducing 0.5% noise and 5% growth in the graph.

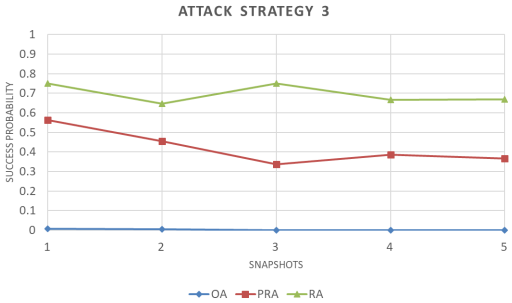
In Figure 5.5 from (a) to (e) we can see very interesting values of the success probability. In (b) we can see that the robust attack values of success probability are dropping fast from the 3^{rd} snapshot. Whereas in the other attack strategies the robust attack is having more constant values than the other attacks. In Figure 5.5(c) and (d) we see that from the 3^{rd} snapshot both of the strategies show a drop in the success probability, where in (d) from the 4^{th} to the 5^{th} snapshot there is a slightly better result. One can assume, that the attacker using the strategy of the hardest to re-identify victim, corrected it's errors from the previous snapshot and successfully re-identified the victim. The best proven strategy in this set of experiments, its the Figure 5.5(a), where the robust attack proves to be the most powerful of the attacks while using the adaptation strategies.



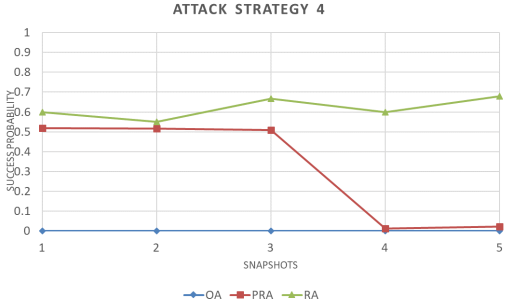
(a) Attack Strategy 1



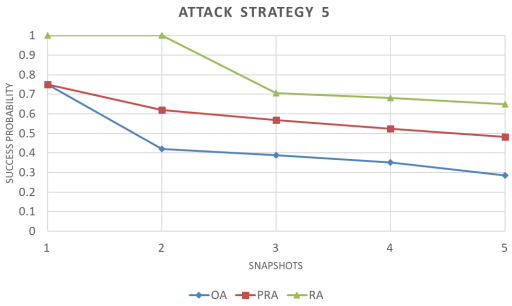
(b) Attack Strategy 2



(c) Attack Strategy 3



(d) Attack Strategy 4

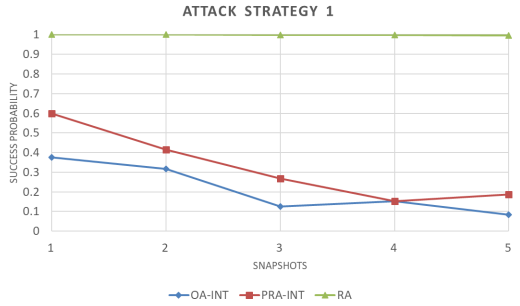


(e) Attack Strategy 5

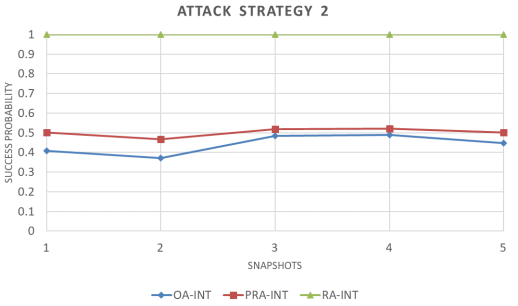
Figure 5.5: The average probability for all attacks without intersection of 1000 runs on graphs with 100 nodes, 20 edges per node, 0.5% noise and 5% evolve of the graph

In Figure 5.6, we observe significantly better results than in Figure 5.5. The biggest difference that is worth mentioning yet again, is the impact that the intersection property has on each of the attack strategies. Being aware, that with each release of the snapshot the graph is perturbed with 0.5% meaning that at the 5^{th} snapshot the noise on the graph is 2.5%, the robust attack shows superior results compared to the other two active attacks. Comparing the adaptation strategies from (a) to (d) with (e), we can see that the attack strategy 5 shows a degradation in success probability between the snapshots, where we can imagine than in more than 5 snapshots all of the attacks will fail at some point. In Figure 5.6 (a) and (b), the random approach appears to benefit the attacker in each snapshot, by randomly targeting new victims and adding sybil nodes. The re-identification of the sybil subgraph is proving to be successful when considering only the intersected

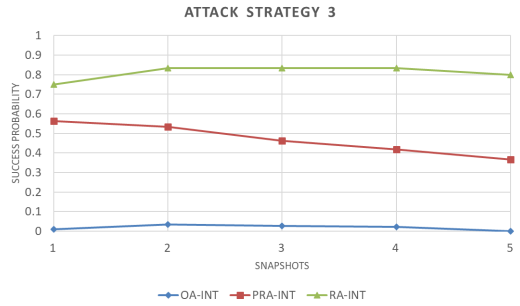
candidate subgraphs.



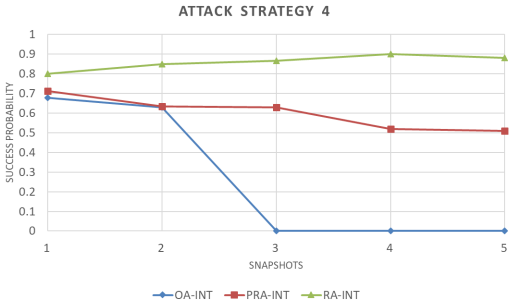
(a) Attack Strategy 1



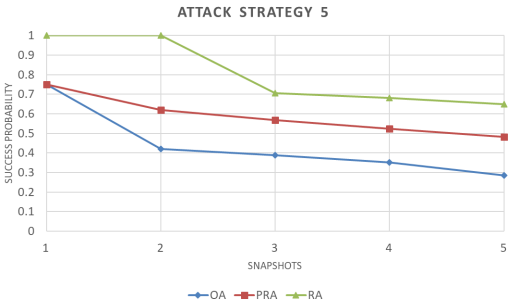
(b) Attack Strategy 2



(c) Attack Strategy 3



(d) Attack Configuration 4



(e) Attack Strategy 5

Figure 5.6: The average probability for all attacks with intersection of 1000 runs on graphs with 100 nodes, 20 edges per node, 0.5% noise and 5% evolve of the graph

Moreover, a difference is observed when comparing the three different values of edges per node, were we see that when we apply the attacks on 100 nodes graph with 4 edges comparing them with 12 and 20, the original walk based attack and the partial robust attack perform better when the graph is sparsely, where as the robust attack shows superiority even in dense graphs, when using 12 or 20 edges per node.

Furthermore, we conducted and presented more experiments. On the next section the experiments are performed on a 100 node graph with all different edges per node values and using 1% of noise per release and 5% growth of the graph. In this set of experiments we want to present a challenging environment for the attacker.

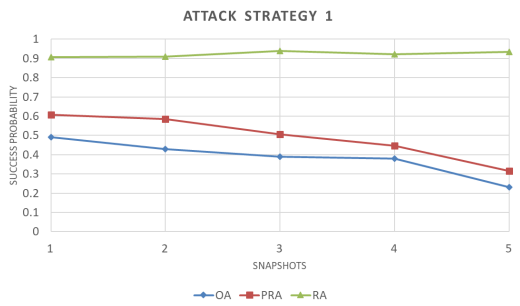
Graphs of 100 vertices with 4 edges per node and using 1% noise and 5% growth of the graph

In this section, all of the results represent a superior robust attack in all of the attack strategies compared to the original walk based and the partial robust. Figure 5.7 (a) we can see that the adaptation strategy used is proving to be effective considering the noisy and the growth of the graph during releases. In 5.7(b), the success probabilities after the first snapshot shows degradation, which if we assume that in more snapshots all of the attacks will fail due to the added noise. We highlight the performance of the attack strategy presented in 5.7(d), where we can see that using the hardest to re-identify victim, the attacker gains in resilience specifically with the robust attack in 2 graph releases and then starts to slowly degrade. The reason why this happens, is the noise in the graph and that potentially the connections between the sybil nodes and victims were removed. We can highlight the fact that the intersection property is the greatest impact for the attacker to succeed and get these results.

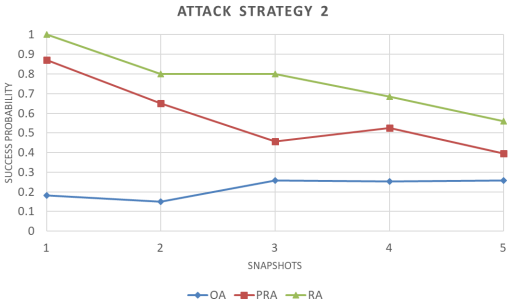
As seen with the other experiments, in Figure 5.8 from (a) to (d) the robust attack is shown to be successful even with the amount of noise in the graph by using the adaptation strategy along with the intersection strategy, whereas (e) shows that the all of the attacks including the robust attack will fail if we do not use any of the adaptation strategies. Comparing the results presented in the Figure 5.8 with the results of Figure 5.8, we can clearly see that the adaptation strategies with intersection produce significantly good results for the latter experiment. Another comparison can be done, by choosing to analyze the results of the non-random approach in Figure 5.8 (c) and random approach in (a), we can state that the non-random approach is gaining resilience and producing convincing success probabilities compared to (a). This shows that having non-random approach in our strategies also benefits the attacker.

Graphs of 100 vertices with 12 edges per node and using 1% noise and 5% growth of the graph

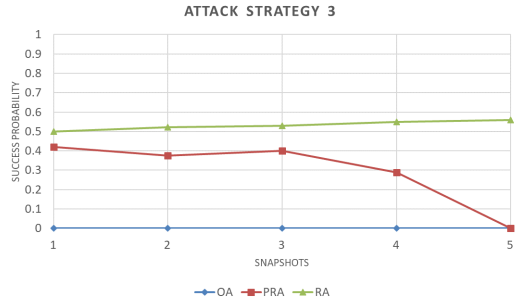
As observed in Figure 5.9 (a) to (e), the robust attack is still producing consistent results even with the high amount of noise in the graph. We can highlight that the attack strategy 1, illustrated in Figure 5.9 (a), shows better results than other attack strategies. One observation, is that all of the attack strategies in this particular experiment start to produce significantly weaker results after the 5^{th} snapshot. In Figure 5.10, we observe that all of the strategies are still proving that the robust attack is well suited using the proposed adaptation strategies. One plot that one should highlight is the attack strategy 3 as a non-random approach depicted in Figure 5.10(c) where is proving to be more reliable than the random approach Figure 5.10(a). The attack strategy is showing gain in resilience due to the reason that the attacker successfully re-identifies the victims that he could not identify in the previously released snapshots. Regardless, because of the noise added in the graph, the robust attack starts to loose its efficiency but still produces



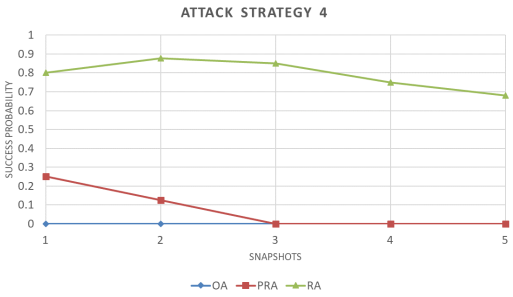
(a) Attack Strategy 1



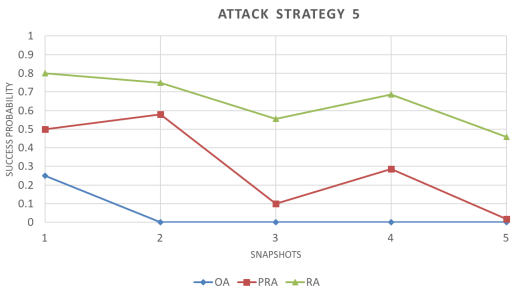
(b) Attack Strategy 2



(c) Attack Strategy 3



(d) Attack Strategy 4



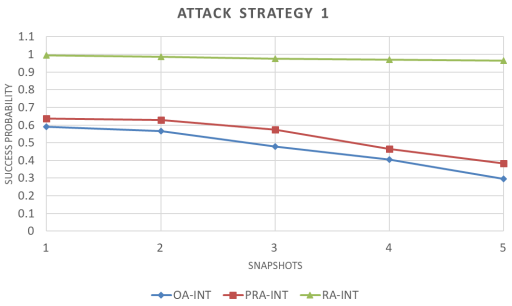
(e) Attack Strategy 5

Figure 5.7: The average probability for all attacks without intersection of 1000 runs on graphs with 100 nodes and 4 edges per node with noise of 1% and 5% growth of the graph for each release

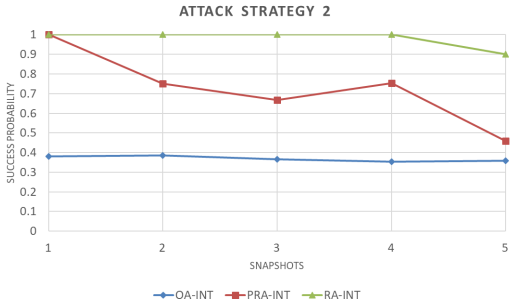
good results.

Graphs of 100 vertices with 20 edges per node and using 1% noise and 5% growth of the graph

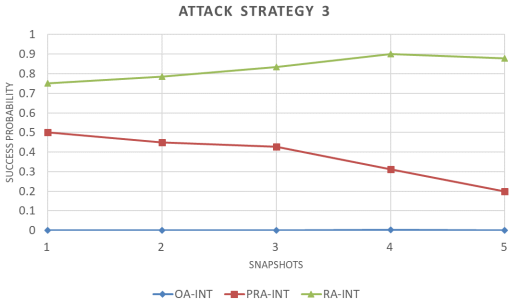
In the Figure 5.11 (a) to (e) we can see that the original attack has failed due to the noise in the graph, whereas the partial robust attack surprisingly produces better results if we do not use the adaptation strategies. The adaptation strategies are proving to be the best fit for the robust attack against the noisy graph throughout all of the snapshots. We would like to highlight again the values that the attack strategy 1 and 3 are producing and their consistency during all of the releases.



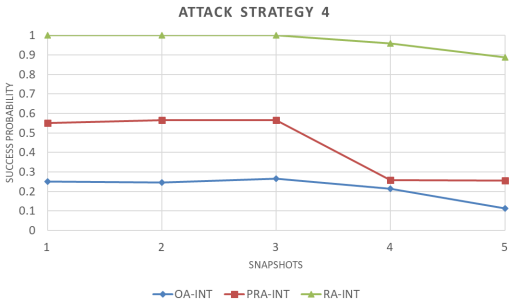
(a) Attack Strategy 1



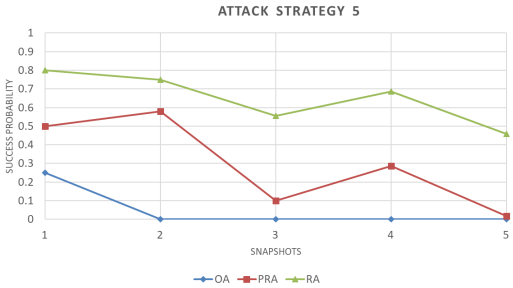
(b) Attack Strategy 2



(c) Attack Strategy 3



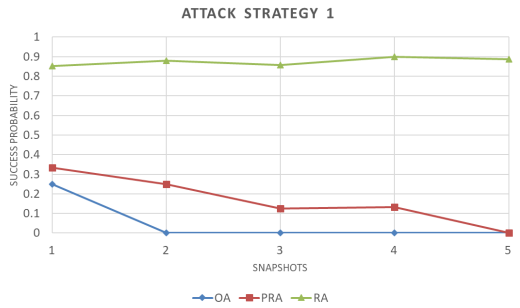
(d) Attack Strategy 4



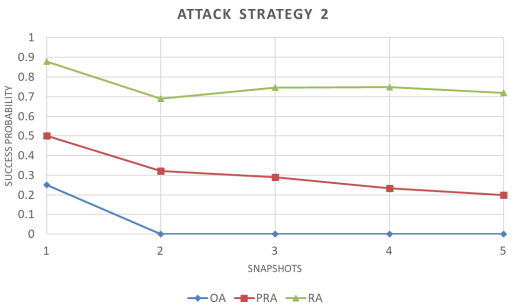
(e) Attack Strategy 5

Figure 5.8: The average probability for all attacks with intersection of 1000 runs on graphs with 100 nodes and 4 edges per node with noise of 1% and 5% growth of the graph for each release

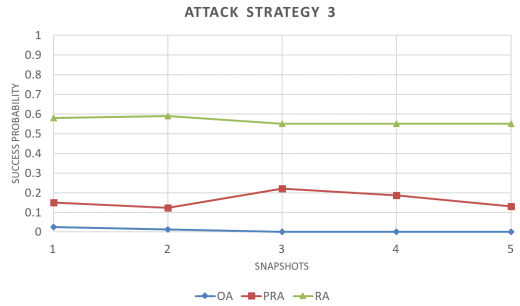
An interesting result is that of the attack strategy 2 depicted in Figure 5.11 (b), where we can see that the robust attack starts to produce weaker probability and then still regains its power. As we can observe, the intersection property in Figure 5.12 proves to be the game changer for all attack strategies. If we compare Figure 5.11 from (a) to (d) with Figure 5.12(a) to (d), we can distinctly observe that the adaptation strategies with intersection prove to be the best combination against a graph with 5% noise at the last snapshot. In Figure 5.12 (b) and (c) the robust attack shows complete superiority against (e) and also against (d). In (d) we can assume that the noise removed changed most of the fingerprints and the attacker trying to re-identify the hardest to find victims could not succeed in re-identifying all of the perturbed fingerprints.



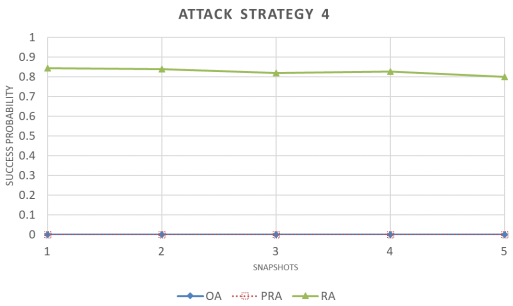
(a) Attack Strategy 1



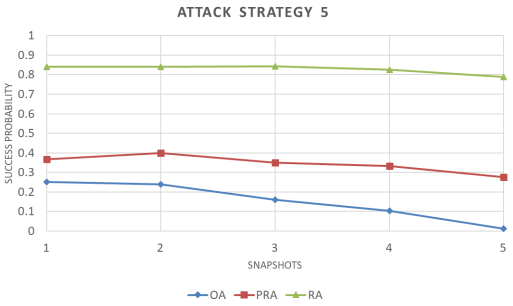
(b) Attack Strategy 2



(c) Attack Strategy 3



(d) Attack Strategy 4



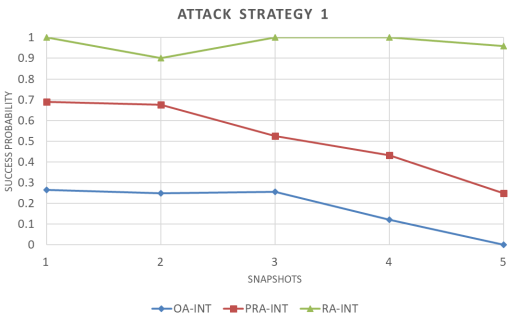
(e) Attack Strategy 5

Figure 5.9: The average probability for all attacks without intersection of 1000 runs on graphs with 100 nodes and 12 edges per node with noise of 1% and 5% growth of the graph for each release

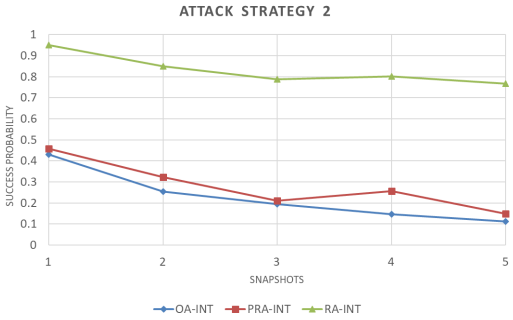
As the last observation we can say that the robust attack performed better for all node densities values but as the graph grew the robust attack was slower than the two other attacks. The results of the robust attack are incomparable with the other two attacks in each of the experimental settings that we constructed even with the amount of noise and the rapid graph growth.

5.3.2 Summary of Results

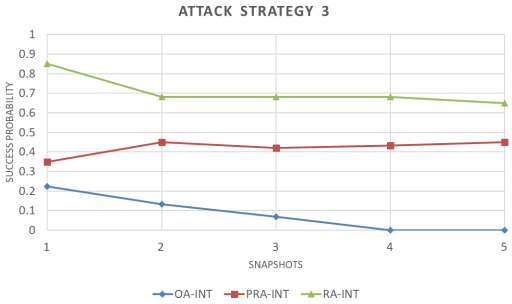
As a conclusion, based on the conducted results we can state that the adaptation strategies benefits the attacker and produces efficient and effective results



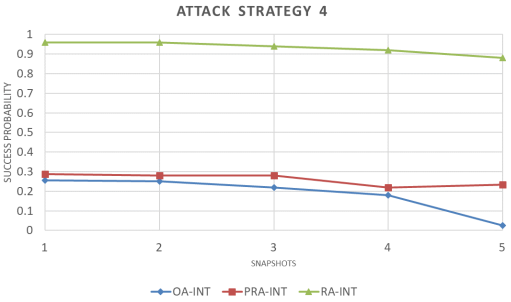
(a) Attack Strategy 1



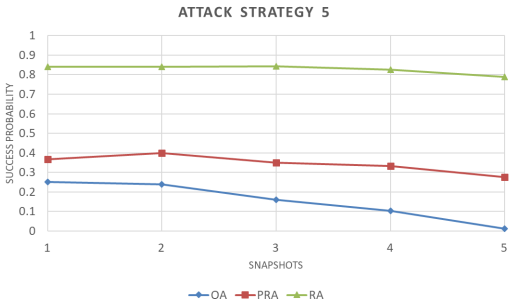
(b) Attack Strategy 2



(c) Attack Strategy 3



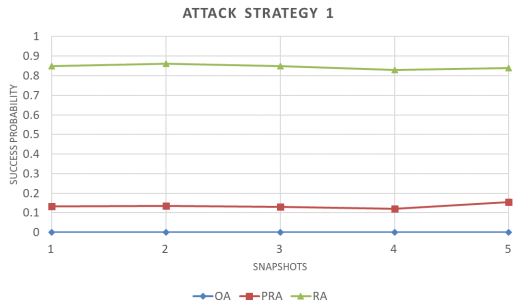
(d) Attack Strategy 4



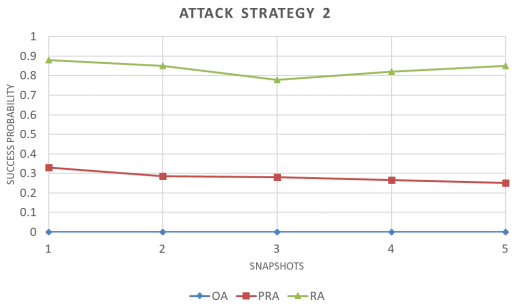
(e) Attack Strategy 5

Figure 5.10: The average probability for all attacks with intersection of 1000 runs on graphs with 100 nodes and 12 edges per node with noise of 1% and 5% growth of the graph for each release

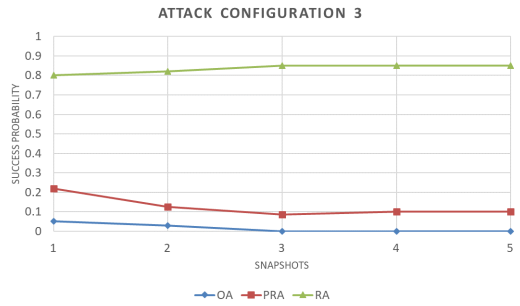
compared to old work in static graphs. The adaptation strategies prove to be well-suited for the robust attack than for the other two attacks. Using the intersection property is a game changer, where the attacker can use the knowledge that he had in the previous snapshot and only use the intersected candidates from the new snapshot with the old one. We can say that targeting new victims per each run and each snapshot was successful as well, proving that the robust attack is efficient in growing the targeted set of victims and successfully re-identifying those in periodically released graphs while using the proposed adaptation strategies. Having non-random and random strategies makes sense, because in one side we can control the amount of changes that we apply to the attacker's subgraph and in the other hand the randomness attack strategies perform faster than the non-random ones.



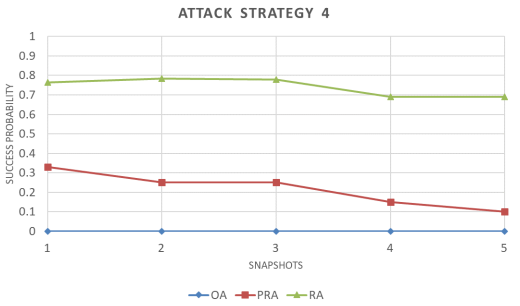
(a) Attack Strategy 1



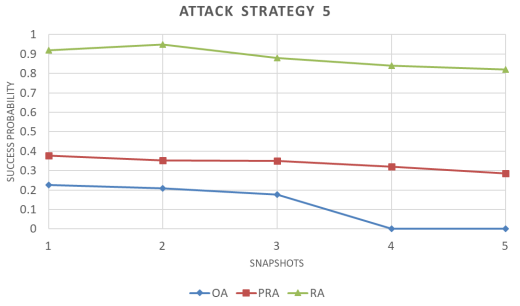
(b) Attack Strategy 2



(c) Attack Strategy 3



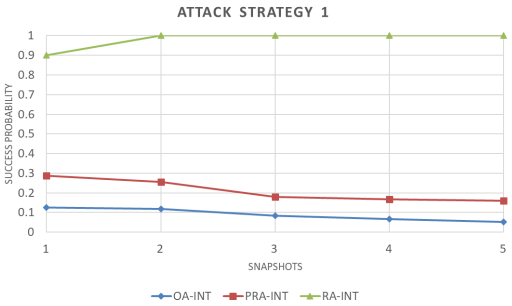
(d) Attack Strategy 4



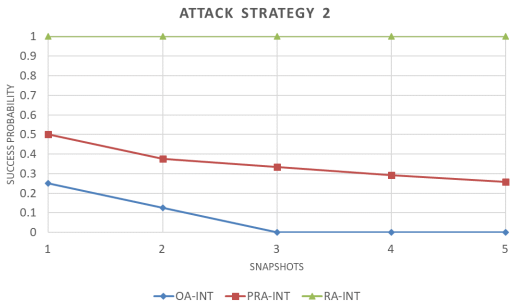
(e) Attack Strategy 5

Figure 5.11: The average probability for all attacks without intersection of 1000 runs on graphs with 100 nodes and 20 edges per node with noise of 1% and 5% growth of the graph for each release

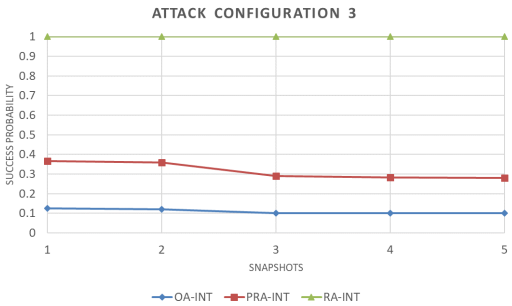
Simulating a dynamic framework, and adapting the attacks to the new framework was the goal was proven to be successful with the results we got. Furthermore, applying the adaptation strategies improved the results and strengthened the attackers capabilities from one snapshot to the other. In Appendix we show more experiments where we can state that all of the attacks, especially the robust attack performs slower once the graph is growing rapidly. Nevertheless, the robust attack is proving to be the most adaptable attack compared to the other two. Even with 5% of noise it is still showing a success probability above 0.75 for each of the attack strategies that were proposed.



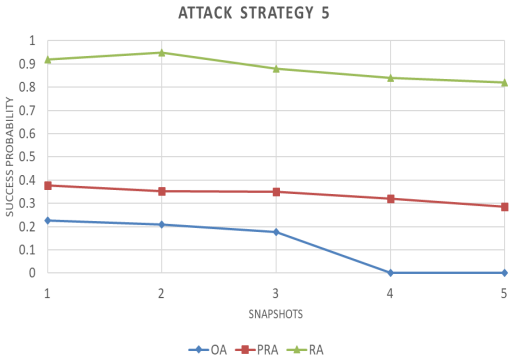
(a) Attack Strategy 1



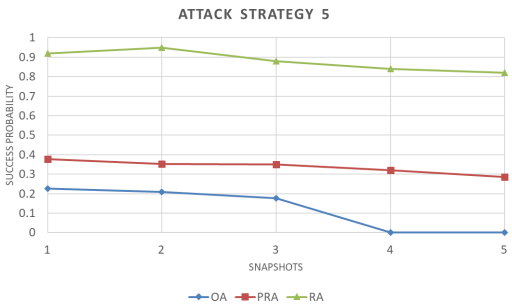
(b) Attack Strategy 2



(c) Attack Strategy 3



(d) Attack Strategy 4



(e) Attack Strategy 5

Figure 5.12: The average probability for all attacks with intersection of 1000 runs on graphs with 100 nodes and 20 edges per node with noise of 1% and 5% growth of the graph for each release

Chapter 6

Conclusion and Perspectives

6.1 Conclusion

In this study, one of the contributions was the construction of a dynamic architecture, where we can simulate periodically released graphs and as such perform re-identification active attacks. The core of our study was the extension the existing original adversarial model in a dynamic environment, where the snapshots of a dynamic social graph are periodically released. Therefore, we have strengthened the capabilities of an adversary when performing an active attack on a periodically released social graph, by devising new attack strategies which make the attacks adaptable in between snapshots. In particular, we have demonstrated different robustness levels of attacks both theoretically and empirically, showing that different levels of robustness combined with the adaptation strategies that we provided lead to considerably better and more successful attacks. In addition, we will showcase attack strategies allowing the adversary to target larger numbers of victims without the need to enrol more sybil nodes than $log\ n$, , thus reducing the likelihood of the attack being thwarted by sybil defences. As the final contribution, we experimentally show that the attacker's capabilities were proven to be stronger in a dynamic scenario. The experiments that were conducted, have proven a general effectiveness in all of the proposed adaptation strategies and also most of them have proven to be fast when it comes to the computational performance. The latter was subject to different parameters of the graph on which the attacks were performed.

Our work provides a new framework as the baseline so that privacy-preserving techniques can be introduced and tested against the proposed strategies. This work introduced the simplest of the anonymization methods, which was presented as the perturbation or the noise in the graph. As a potential approach to work on this framework, we would say that these attack strategies should be tested along side with the anonymization methods as presented in Chapter 2. The attack strategies introduced in this work are intuitive and well-justified, but the attacker model is general.Hence, as a future perspective we can consider using alternative techniques and some resources in game theory. Additionally, the evaluation presented in this

thesis is constructed for specific graph sizes. A possible research direction can be proposing to use larger graphs where we consider more graph parameters or a more complex experimental settings, such as different combination of the proposed attack strategies.

From a machine learning point of view, regarding re-identification active attacks, one might choose to introduce some machine learning techniques in the evolve methods that were developed. One can be, instead of randomly picking the hardest to find fingerprints and modify them, with the implementation of a recommender system [35] the algorithm might be more efficient on giving the specific victim to modify rather than choosing randomly the most confusing set.

Appendix

We present in this chapter some additional information about our study, such as, the link to the source-code developed; more additional experimental results.

6.2 Source-code link

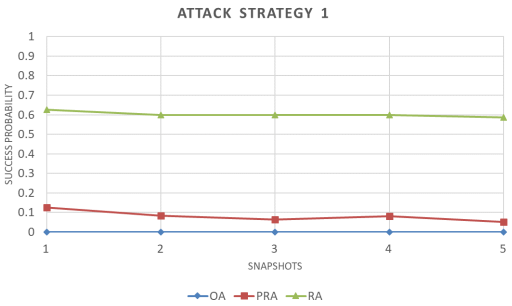
The source-code that was developed for this work is presented in this link:

https://github.com/emakepuska2/Dynamic-graph-releases–Re-identification-active-attacks

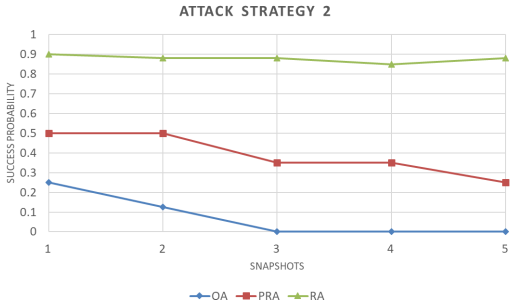
6.3 Additional experiments on graphs of 100 nodes with 1% noise and 10 % growth of the graph

Additional experiments show that the robust attack performs surprisingly well with 5% perturbation throughout the snapshots and with 50% growth of the graph in the 5^{th} snapshot in all presented scenarios where the adaption strategies are combined with the intersection strategy.

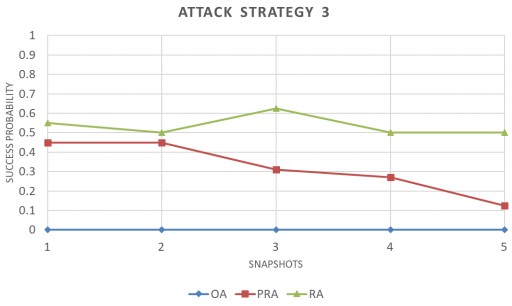
Graphs of 100 vertices with 4 edges per node and using 1% noise and 10% growth of the graph



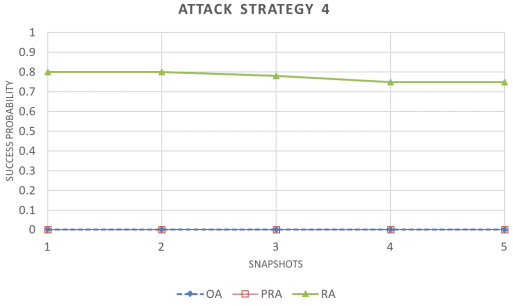
(a) Attack Strategy 1



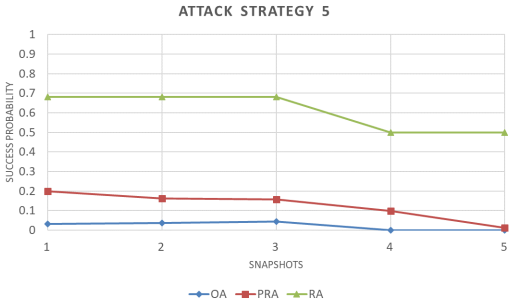
(b) Attack Strategy 2



(c) Attack Strategy 3

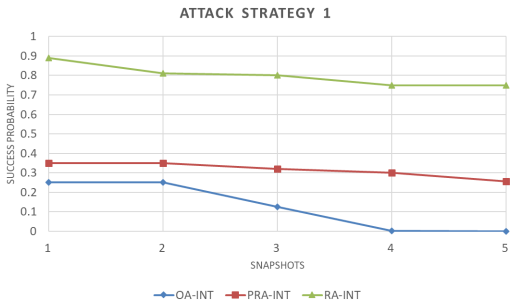


(d) Attack Strategy 4

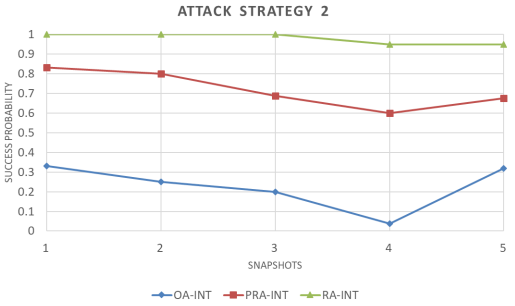


(e) Attack Strategy 5

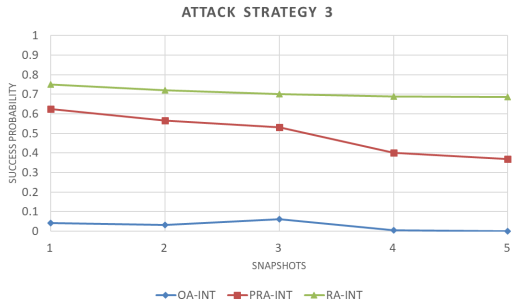
Figure 6.1: The average probability for all attacks without intersection of 1000 runs on graphs with 100 nodes and 4 edges per node with noise of 1% and 10% growth of the graph for each release



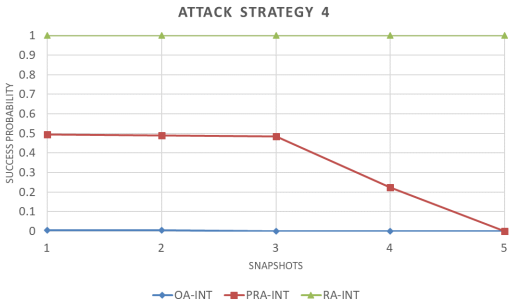
(a) Attack Strategy 1



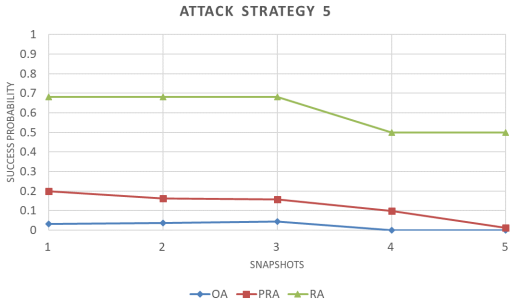
(b) Attack Strategy 2



(c) Attack Strategy 3



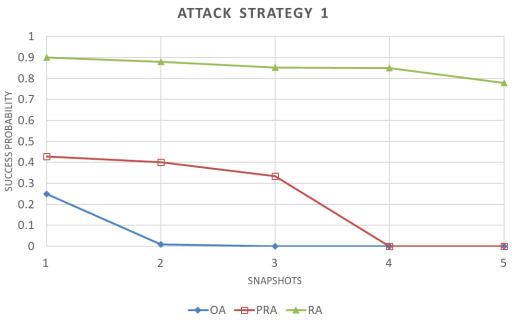
(d) Attack Strategy 4



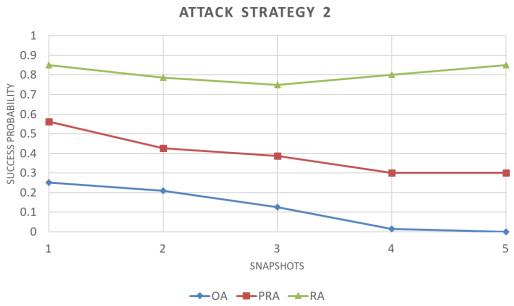
(e) Attack Strategy 5

Figure 6.2: The average probability for all attacks with intersection of 1000 runs on graphs with 100 nodes and 4 edges per node with noise of 1% and 10% growth of the graph for each release

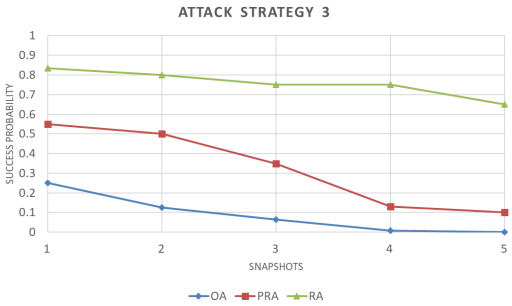
Graphs of 100 vertices with 12 edges per node and using 1% noise and 10% growth of the graph



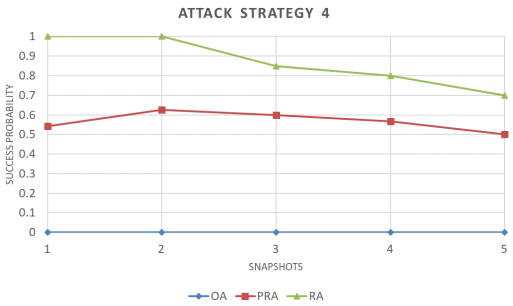
(a) Attack Strategy 1



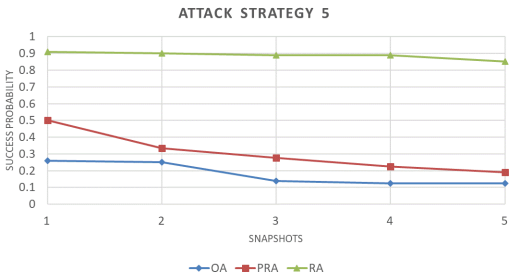
(b) Attack Strategy 2



(c) Attack Strategy 3

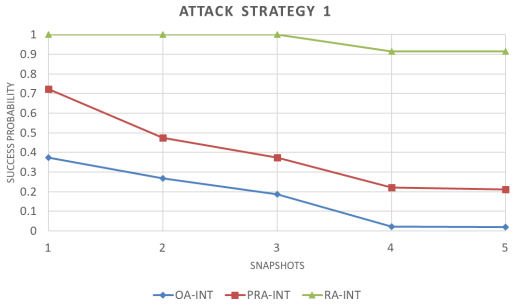


(d) Attack Strategy 4

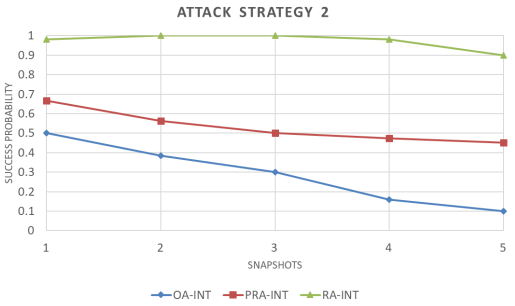


(e) Attack Strategy 5

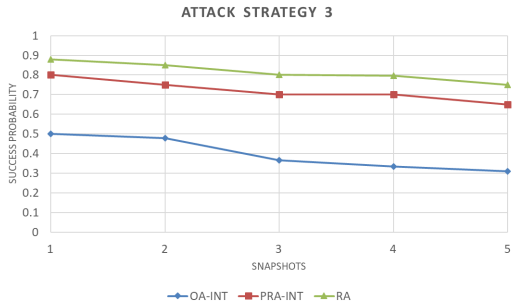
Figure 6.3: The average probability for all attacks without intersection of 1000 runs on graphs with 100 nodes and 12 edges per node with noise of 1% and 10% growth of the graph for each release



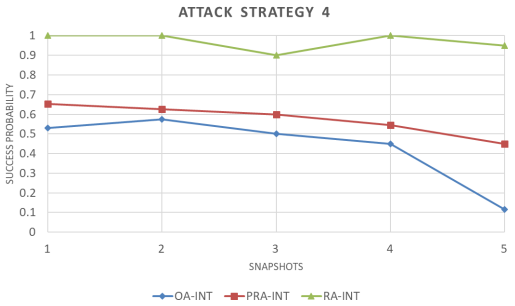
(a) Attack Strategy 1



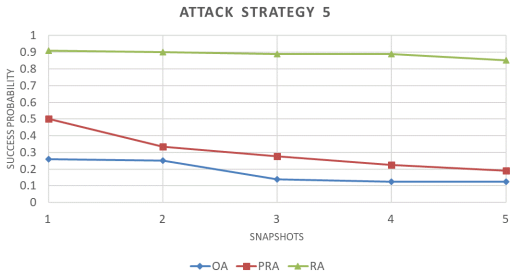
(b) Attack Strategy 2



(c) Attack Strategy 3



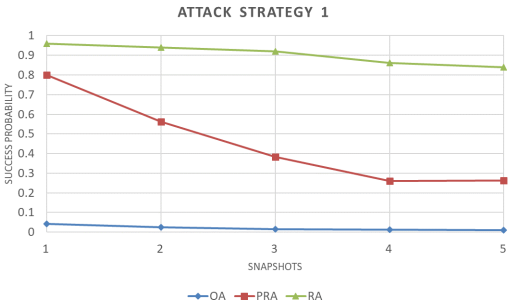
(d) Attack Strategy 4



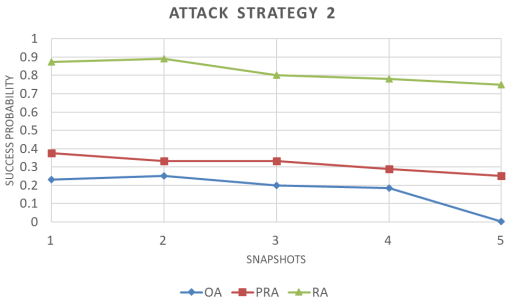
(e) Attack Strategy 5

Figure 6.4: The average probability for all attacks with intersection of 1000 runs on graphs with 100 nodes and 12 edges per node with noise of 1% and 10% growth of the graph for each release

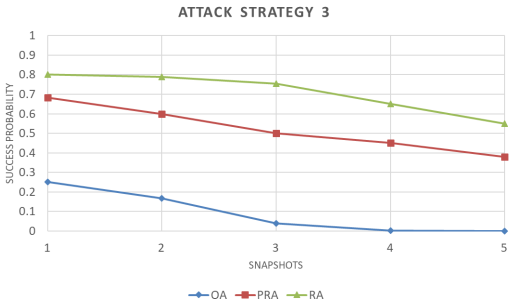
Graphs of 100 vertices with 20 edges per node and using 1% noise and 10% growth of the graph



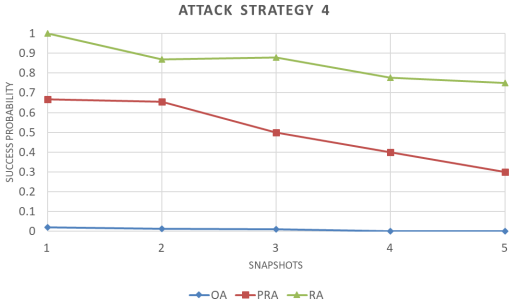
(a) Attack Strategy 1



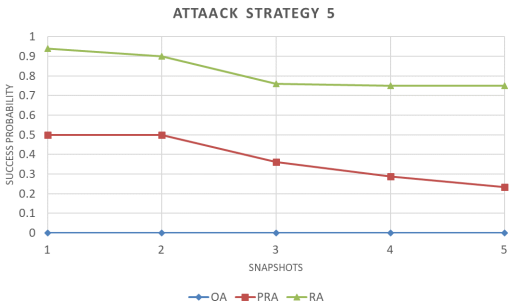
(b) Attack Strategy 2



(c) Attack Strategy 3

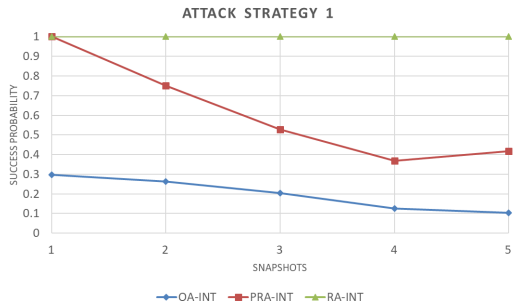


(d) Attack Strategy 4

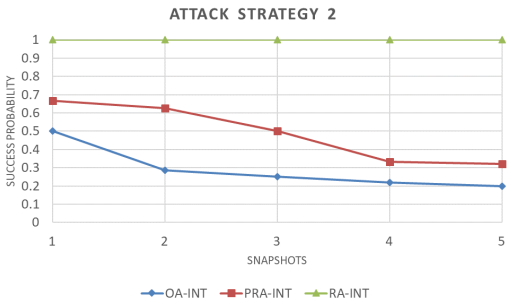


(e) Attack Strategy 5

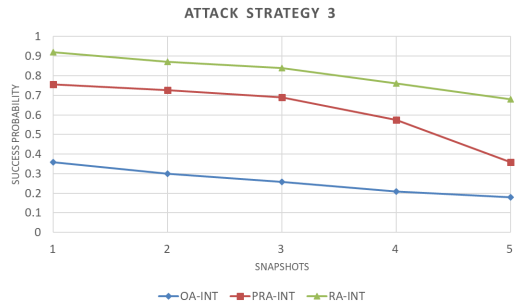
Figure 6.5: The average probability for all attacks without intersection of 1000 runs on graphs with 100 nodes and 20 edges per node with noise of 1% and 10% growth of the graph for each release



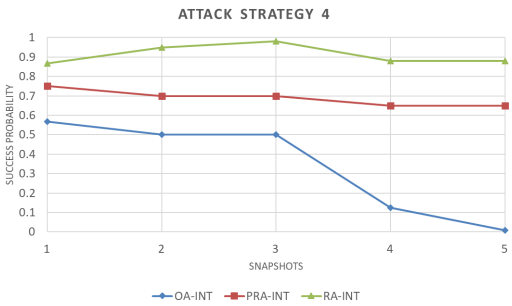
(a) Attack Strategy 1



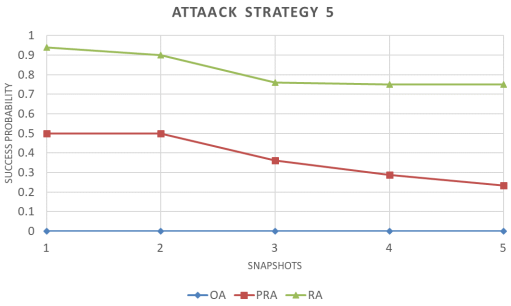
(b) Attack Strategy 2



(c) Attack Strategy 3



(d) Attack Strategy 4



(e) Attack Strategy 5

Figure 6.6: The average probability for all attacks with intersection of 1000 runs on graphs with 100 nodes and 20 edges per node with noise of 1% and 10% growth of the graph for each release

Bibliography

[1] L. Adamic and E. Adar. How to search a social network. In *Social Networks 27*, 2005.

[2] L. Backstrom, C. Dwork, and J. Kleinberg. Wherefore art thou r3579x? anonymized social networks, hidden patterns, and structural steganography. In *Proceedings of the 16th international conference on World Wide Web, WWW'07*, pages 181–190, New York, NY, USA, 2007. ACM.

[3] L. Backstrom, D. Huttenlocher, J.Kleinberg, and X. Lan. Group formation in large social networks: Membership, growth and evolution. In *Proc. KDD*, 2006.

[4] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[5] Jordi Casas-Roma, Jordi Herrera-Joancomartí, and Vicenç Torra. k-degree anonymity and edge selection: improving data utility in large networks. *Knowledge and Information Systems*, 50(2):447–474, Feb 2017.

[6] Jordi Casas-Roma, Jordi Herrera-Joancomartí, and Vicenç Torra. An algorithm for k -degree anonymity on large networks. pages 671–675, 08 2013.

[7] Sean Chester, Bruce M. Kapron, Ganesh Ramesh, Gautam Srivastava, Alex Thomo, and S. Venkatesh. Why waldo befriended the dummy? k-anonymization of social networks with pseudo-nodes. *Social Network Analysis and Mining*, 3(3):381–399, Sep 2013.

[8] Wajeb Gharibi and Abdulrahman Mirza. Software vulnerabilities, banking threats, botnets and malware self-protection technologies. *Computing Research Repository - CORR*, 8, 05 2011.

[9] Rupesh Gunturu. Survey of sybil attacks in social networks. *CoRR*, abs/1504.05522, 2015.

[10] Ravi Kumar, Jasmine Novak, and Andrew Tomkins. Structure and evolution of online social networks. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, pages 611–617, New York, NY, USA, 2006. ACM.

[11] Jure Leskovec, Deepayan Chakrabarti, Jon Kleinberg, Christos Faloutsos, and Zoubin Ghahramani. Kronecker graphs: An approach to modeling networks. *J. Mach. Learn. Res.*, 11:985–1042, March 2010.

[12] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Trans Knowledge Discov Data*, 1, 04 2006.

[13] Kun Liu and Evimaria Terzi. Towards identity anonymization on graphs. In *SIGMOD Conference*, 2008.

[14] Xuesong Lu, Yi Song, and Stéphane Bressan. Fast identity anonymization on graphs. In Stephen W. Liddle, Klaus-Dieter Schewe, A. Min Tjoa, and Xiaofang Zhou, editors, *Database and Expert Systems Applications*, pages 281–295, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[15] Tinghuai Ma, Yuliang Zhang, Jie Cao, Jian Shen, Meili Tang, Yuan Tian, Abdullah Al-Dhelaan, and Mznah Al-Rodhaan. KDVEM: a k-degree anonymity with vertex and edge modification algorithm. *Computing*, 97(12):1165–1184, Dec 2015.

[16] S. Mauw, Y. Ramirez-Cruz, and R.Trujillo-Rasua. Anonymising social graphs in the presence of active attackers. In *Transactions on Data Privacy, 11(2)*, pages 169–198, 2018.

[17] Sjouke Mauw, Yunior Ramírez-Cruz, and Rolando Trujillo-Rasua. Conditional adjacency anonymity in social graphs under active attacks. *Knowledge and Information Systems*, pages 1–27, Dec 2018.

[18] Sjouke Mauw, Yunior Ramírez-Cruz, and Rolando Trujillo-Rasua. Robust active attacks on social graphs. *Data Mining and Knowledge Discovery*, 33(5):1357–1392, Sep 2019.

[19] Sjouke Mauw, Rolando Trujillo-Rasua, and Bochuan Xuan. Counteracting active attacks in social network graphs. In Silvio Ranise and Vipin Swarup, editors, *Data and Applications Security and Privacy XXX*, pages 233–248, Cham, 2016. Springer International Publishing.

[20] Dimitrios Michail, Joris Kinable, Barak Naveh, and John V Sichi. Jgrapht–a java library for graph data structures and algorithms. *arXiv preprint arXiv:1904.08355*, 2019.

[21] A. Narayanan and V. Shmatikov. De-anonymizing social networks. In *2009 30th IEEE Symposium on Security and Privacy*, pages 173–187, May 2009.

[22] W. Peng, F. Li, X. Zou, and J. Wu. Seed and grow: An attack against anonymized social networks. In *2012 9th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, pages 587–595, June 2012.

[23] W. Peng, F. Li, X. Zou, and J. Wu. A two-stage deanonymization attack against anonymized social networks. *IEEE Transactions on Computers*, 63(2):290–303, Feb 2014.

[24] François Rousseau, Jordi Casas-Roma, and Michalis Vazirgiannis. Community-preserving anonymization of graphs. *Knowledge and Information Systems*, 54(2):315–343, Feb 2018.

[25] Alessandra Sala, Xiaohan Zhao, Christo Wilson, Haitao Zheng, and Ben Y. Zhao. Sharing graphs using differentially private graph models. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, IMC '11, pages 81–98, New York, NY, USA, 2011. ACM.

[26] Julián Salas and Vicenç Torra. Graphic sequences, distances and -degree anonymity. *Discrete Applied Mathematics*, 188, 06 2015.

[27] P. Samarati. Protecting respondents identities in microdata release. *IEEE Transactions on Knowledge and Data Engineering*, 13(6):1010–1027, Nov 2001.

[28] Latanya Sweeney. K-anonymity: A model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10(5):557–570, October 2002.

[29] S. Varrette, P. Bouvry, H. Cartiaux, and F. Georgatos. Management of an academic hpc cluster: The ul experience. In *Proc. of the 2014 Intl. Conf. on High Performance Computing & Simulation (HPCS 2014)*, pages 959–967, Bologna, Italy, July 2014. IEEE.

[30] Alexei Vazquez. Growing network with local rules: Preferential attachment, clustering hierarchy, and degree correlations. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 67:056104, 06 2003.

[31] Wei Wei, Fengyuan Xu, C. C. Tan, and Qun Li. Sybildefender: Defend against sybil attacks in large social networks. In *2012 Proceedings IEEE INFOCOM*, pages 1951–1959, March 2012.

[32] Mingqiang Xue, Panagiotis Karras, Chedy Raïssi, Panos Kalnis, and Hung Keng Pung. Delineating social network data anonymization via random edge perturbation. pages 475–484, 10 2012.

[33] H. Yu, P. B. Gibbons, M. Kaminsky, and F. Xiao. Sybillimit: A near-optimal social network defense against sybil attacks. In *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pages 3–17, May 2008.

[34] H. Yu, M. Kaminsky, P. B. Gibbons, and A. D. Flaxman. Sybilguard: Defending against sybil attacks via social networks. *IEEE/ACM Transactions on Networking*, 16(3):576–589, June 2008.

[35] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. Deep learning based recommender system: A survey and new perspectives. *ACM Comput. Surv.*, 52(1):5:1–5:38, February 2019.

[36] B. Zhou and J. Pei. Preserving privacy in social networks against neighborhood attacks. In *2008 IEEE 24th International Conference on Data Engineering*, pages 506–515, April 2008.

[37] Lei Zou, Lei Chen, and M. Tamer Özsu. K-automorphism: A general framework for privacy preserving network publication. *Proc. VLDB Endow.*, 2(1):946–957, August 2009.