Decentralized Privacy Preserving Services for Social Networking Sites

Towards User Engaging Privacy

Department of Theoretical and Applied Science
University of Insubria

Leila Bahri

Supervisors: Dr. Elena Ferrari (Full Professor), and Dr. Barbara Carminati (Associate Professor)

External Reviewers: Dr. George Pallis, and Dr. Amirreza Masoumzadeh

This dissertation is submitted for the degree of
Doctor of Philosophy in Computer Science

June 2016

"*And when your Lord made it known: If you are grateful, I would certainly give to you more, and if you are ungrateful, My chastisement is truly severe.*" – Quran (Ibrahim/Abraham Chapter, Verse 7 – SHAKIR Transalation)

To my mother ...
(wordless I am, mom! Promising to keep doing my best. You are observing me, I know.)

To my father ...
(scared I am, dad! It now does start, but for you I stand up to my fears.)

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except if otherwise specified in the text. However, every single bit of it could never be achieved without the support of so many people, for whom I shall always remain grateful.

Leila Bahri
June 2016

Acknowledgements

It is reported on the prophet Muhammad PBUH having said "*Who does not thank the people is not thankful to God.*" [Sunan Abi Dawud 4811. Book 43, Hadith 39]

I would like to open the long list of my owed thanks by acknowledging my appreciation and gratitude for my PhD mentors, *Dr. Elena Ferrari* and *Dr. Barbara Carminati*. What I have learned from you during my PhD years is a solid capital for my future career investments. Knowledge reproduces as it is shared. As I move ahead through my endeavors, I will always be thanking you, spreading the knowledge I received from you.

On the same line of knowledge reproduction, I would like to express my thanks to all the iSocial project members, both students and supervisors, for the instructive information and knowledge sharing we had. I further specify more due thanks for *Dr. Sarunas Girdzijauskas*, the project coordinator, for all the instructive feedback he carefully attributed to our (students) talks, and for having received me for my first secondment in the project. I equally thank my friend *Amira Soliman*, for all the moments we shared and for the fruitful collaboration we had. I keep looking forward to more exchanges and collaborations with you all, my iSocial fellows.

I do not forget to thank all my colleagues and friends in the STRICT Social Lab. Special thanks go to *Dr. Ngoc Tran*, who has been my fun-to-be-around friend throughout all my PhD time, and thanks to whom I learned much regarding the nutritional value of different "cold" vs. "hot" food. Our shared lunches in the fourth floor and the morning coffee cups make special memories that I shall always cherish. I also direct precise thanks to *Naeimeh Laleh* and to *Natasha Poposka*, without forgetting any of the other people whom I met in Varese city. I shall not forget to mention the administrative and technical assistants, *Roberta Viola* and *Mauro Santabarabara*, who both have been of tremendous administrative and technical help.

I also must thank my thesis reviewers and the evaluation committee for their instructive input and comments on my work.

Geographically we have been separated, but we have never been far from each other. Due floured thanks shall blossom for my dearest sister *Hajar Bousfiha*, who has always been of high emotional and intellectual support to me. I equally thank *Hanae Raiss* and *Khadija Akherfi* for their warm-heartening friendship, without forgetting all my other friends,

especially the ones who made me run marathons to attend their dear weddings. I am very proud of your friendship, and very grateful for your blissful existence in my life.

Have I forgotten my family? I do last what I cannot do well. I can never be good enough to express my gratitude to them. They are the source of infinite giving and unconditional support, and I am nothing but a little burgeon sprouting in their land. My dearest and best *dad* who has always explicitly and implicitly backed me up and provided me with the strength to keep moving forward. My beloved *brothers* and little *sister* from whom I always get inspired. My *mom* whose voice I might have forgotten, has left me shining clear words that have been lightning my path thus far, and without which I would have never been writing an acknowledgement on my PhD thesis. I can never thank you enough, if not by promising to do my best to ensure continuity to your generous giving.

Then, I cannot close this never ending list without giving credit to my *secret supporting angel*, the person who composed music from my heartbeats, making it my motivational rhythm and my source of inspiration whenever I was to fall in the monotony of silence. You have strengthened me throughout all my hardships, during those PhD years, and helped me to stand straight again whenever I stooped. I am sure so it will continue to be ...

Abstract

Decentralized Online Social Networks (DOSNs) have been investigated as an alternative transparent solution to protect users privacy and to avoid surveillance concerns resulting from the centralization model. The core idea is to give online social network (OSN) users full control over the management of their data without having to delegate it to a central service provider. However, there are various technical challenges that a decentralized architecture imposes, and that have been limiting DOSNs to compete against the centralized model.

One of the most prominent of these challenges is access control management. Most of the available proposals take a cryptographic approach. These solutions may provide strong security guarantees; however, they fail short at modeling the required fine-grain and dynamic access scenarios of OSNs, they introduce high operational costs, and they suffer to scale [110]. Therefore, our main research question in this thesis is on what other alternatives could be exploited to offer flexibility in defining fine grain access controls, and that would scale better than encryption based techniques.

Our research resulted in the design of two access control models that adopt approaches that have not yet been explored within OSNs. These are an aposteriori based, and a label based solutions.

On the path to achieving those solutions, and starting from an understanding of the access control process, we have found it necessary to dedicate efforts to the investigation of the problem of identity validation in OSNs as well. Identification comes as a critical step in ensuring the meaningfulness of any access control solution that could be put in place. In OSNs, identification is more of a user responsibility than it is a system one. The looseness in obtaining an identity in an OSN (i.e., creating an account with a valid email address), makes it up to the user whether or not to trust the claimed identity on a profile. Thus far, there have been no proposals to assist users with such a task of reliable profiles identity estimation. Therefore, our additional contribution in this thesis came to put a block in this identified gap.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

If "privacy" is mentioned in a sentence, the word that will most probably follow is "preservation". Preservation means to safeguard against danger or harm, to keep possession of, to retain, or to maintain something.[1] It implies that whatever is to be preserved is already acquired, or is an inherent inseparable characteristic. Thus far, we understand that privacy may refer to something that is already earned, fundamental, or probably innate to whatever it belonged to.

What is privacy?

This term is often used in ordinary language as well as in philosophical, political and legal discussions, yet there is no one definition or analysis of the word [41]. It is historically known to have roots in sociological and anthropological discussions, with philosophical origins in Aristotle's separation of political activity, that he qualified as public, and the private circle of family and personal life [41]. However, there remains confusion over the meaning, the value and the scope of privacy. This qualifies it to the level of "concept", or a construct of ideas and intellectual thoughts.

It was not until late in the nineteenth century that a systematic written discussion on the concept of privacy was made by Samuel Warren and Louis Brandeis in their 1890 essay titled "The Right to Privacy" [141]. The essay presented normative views on what the authors believed should be protected by the law under the scope of privacy. To the authors, privacy encompassed a spectrum that is larger than physical protection of one's home or one's physical property, and was defended as "the right to be let alone" [141]. Warren and Brandeis focused mostly on the press and on the publicity effects produced by the new emerging technological inventions of the time, such as photography and widely distributed newspapers. They shed the light on the possible invasion of a person's private life by vast public dissemination of details and information. They argued that new technology made it essential to explicitly

[1] Merriam-Webster: http://www.merriam-webster.com/dictionary/preservation

recognize a more general right to privacy that covered people's right to have control over how their thoughts, sentiments, and emotions could be shared with others.

After Warren and Brandeis rooted the foundation of *informational privacy*, a privacy that stretched to cover control over information about oneself, more systematic debates on the construction of the concept blossomed towards the end of the twentieth century, with the introduction of informational privacy protection in the law [41]. Among the many pioneering discussions on the concept of privacy was its defense as the quality by which people are granted the ability to control the access that others may have to them [56, 7, 99]. This ability was also viewed as a collection of norms that go beyond controlling access to promoting personal expression and choice [118].

With the development of more advanced technological products that enabled the conception, the carriage, the dissemination, and the persistence of information, such as the video-camera, video-tapes, the telephone, the fax, etc., informational privacy continued to attract valuable interest. With the appearance and the spread of Internet and of the World Wide Web, two technologies that connected all corners of the world and that enabled free and easy information sharing of all types, commentators in the field of informational privacy continued to realize how technological advances keep challenging and threatening privacy in ways that may be uncontrollable, irreversible, or mostly unfathomable by the concerned individuals [107]. Massive records of information about individuals financial and credit history, medical records, purchase history, telephone calls, etc., are a small subset of the diverse and dense information that is available over the Internet, about people that may not exactly know what information is stored about them, by whom, and who has access to it [102].

The relentless and rapid development of technological devices, networks, and hardware has not taken long before making the Internet and the WWW affordable and available to almost all people all over the world, and not a privilege accessible only to some [47]. This may have been what shifted the online world to an era of connected individuals on a social level, after it was mostly a web of connected computers, systems, and corporations mainly. What would be known as the social web has made it possible for people to connect, to produce, and to share information in ways that were not possible before [47]. The innate social characteristics of humans and their cognitive ability to build groups and to socially interact with others have simulated the creation and the development of online social networks (OSNs) and have also made their rise as a significant cultural phenomenon [19].

OSNs are networked systems that represent people as profiles that they create, and that they manage to establish connections with other profiles on the network. OSNs enable people to keep in touch with their contacts beyond physical and geographical boundaries, allow them free expression by means of uncensored creation and sharing of information, and facilitate their openness on wider social circles than was ever possible before [47]. OSNs connect millions of people of all ages and backgrounds across all the globe and offer them

an open space for autonomous exchange [47]. This experience of freedom of open personal and social expression that spans over geographical borders resulted in a deliberate sharing of information about oneself of which the consequences on one's privacy remain unfathomable and obscure to most of OSNs users.

The era of the social web shifted the grammatical position of an individual in normative statements about one's privacy from that of an object to that of a subject. In the social web, a person is no more a passive object whose privacy is subject to violation by some other entity, but is also an active subject who is deliberately staining her/his own privacy. The threatening challenge in this equation is that people, in most of the cases, are not aware of this paradigm shift or are not delivered with the needed tools to comprehend it and to have control over it.

1.1 Motivations

Humans have been socializing and exchanging information about themselves ever since they existed. However, this exchange has mainly taken place in a one-to-one fashion without the need for an intermediary. In the realms of OSNs, as they are majorly widespread and deployed nowadays, socializing has taken a completely new structure: the exchange between people is being facilitated through the interposition of a service provider. As such, commentating on one's informational privacy in OSNs requires the consideration of two levels of abstraction: 1) a traditional level that represents control over how one wants to appear to the others he/she socializes with, and 2) a newly established level that represents control over one's thoughts, emotions, behavior, trends, and personality that could be extracted from the diverse and dense collection of data they deliver to the intermediary service provider. To distinguish between these two levels, we describe the first one as *micro-privacy* and the second one as *macro-privacy*. In the technical literature, these could be described as online vs. offline privacy, or as online vs. statistical privacy, respectively [124]. Given that offline privacy could be used to describe privacy in non technological platforms as well, such as hard documents management, trash disposal, etc. (e.g., [92]), we majorly adopt in this document the suggested differentiation between micro and macro privacy, whilst interchangeably using online and micro privacy as well.

By *macro-privacy* we refer to the exploitation of users data and related meta-data, either explicitly shared or implicitly collected from their online behavior, at mass scales by technology providers (i.e., OSNs providers) for varying purposes, mainly for profit generation through advertising, analytic studies, etc. On the *micro level*, we refer to users privacy as related to their personal interactions with other individuals (i.e., friends), and to the limits they may desire to impose on their data accessibility by their friends.

Given its large scope of effects as a serious threat to users right "to be let alone" (defended by Warren and Brandeis as the fundamental element to the right to informational privacy

[141]), contemporary privacy advocates have been putting considerable focus on macro-privacy issues[69, 102]. One of their most systematic and substantial responses has been the consideration of an alternative and completely different architecture for an OSN service provision: a peer-to-peer design [102, 145, 38]. In point of fact, to eradicate privacy concerns in OSNs at the macro-level, it would be ideally sufficient to remove the central data storage and management entity; thus, substitute it by a decentralized design whereby every user locally holds, stores, and manages her/his data as is the case for pre-tehcnology socializing. This thread of conception led to important research aiming at building practical, efficient, and usable decentralized online social networks (DOSNs) [39, 71, 122, 22, 26].

By their decentralized abstraction, DOSNs intuitively clear away macro-privacy concerns; however, their achievement opens other technical and usage challenges of which micro-privacy, or the management of access control to the decentralized data, is one of the most significant [70]. In general, managing users privacy preferences w.r.t their interactions with their friends is a typical access control problem. Users express access policies that reflect the restrictions they desire to impose on their data access, and the managing system ensures the enforcement of these policies. In a centralized social networks, all the data and its related access policies are centrally stored and managed. Access requests to the data are then directed to one central entity that ensures their evaluation in an autonomous manner. However, in the absence of such a central repository for data storage and management where these access control policies could be centrally enforced and observed, managing access control in a DOSN becomes a technical challenge.

Thus far, the access control problem for DOSNs has been mainly addressed with a cryptography perspective [39, 122, 70]. The premise is to securely lock all pieces of users data and to allow the dissemination of appropriate keys to those allowed to gain access to it. This line of actions may provide deterministic solutions to the challenge of remote access control enforcement at the level of other peers, but fails at its best to provide the needed flexibility in both the formulation of fine grain access policies, and in their corresponding dissemination [123, 70].

Information sharing in social networks has been observed to be characterized by denseness, diversity, and dynamicity. People share huge amounts of pieces of information under diverse formats (e.g., text, photos, voice) in a stochastic manner with continuously changing needs w.r.t dissemination preferences [104]. Moreover, information ownership does not follow a direct one-to-one fashion. In contrast to single-ownership of data where one definite entity is known to hold complete control over it, data ownership in social networks is mostly collaborative and multiple. For instance, a person might share a photo that displays other people, or might share a text in which other people are tagged, or might comment on a video shared by another person. All these elements make the firmness and rigidness of encryption mechanisms insufficient, at best, in providing an acceptable balance between protection and usability and efficiency of an access control solution for DOSNs.

Therefore, there is need for alternative solutions to the access control problem in DOSNs. These solutions should address a threefold challenge. Firstly, there is need for a mechanism that would allow users to express their fine grain data access requirements and policies in a usable and friendly manner. Secondly, there is the challenge of disseminating these access policies in a transparent and reliable manner between concerned peers in the network. Thirdly, comes the challenge of remotely enforcing these policies at the level of other peers providing required security and privacy guarantees.

1.2 Objective

With reference to the above discussed motivational factors, we formulate our objective as follows:

> **OBJECTIVE: Conceive alternative solutions for managing social network users' online privacy (micro-level), especially under the decentralized model, that are *practical*, *efficient*, *transparent*, and *users engaging*.**

In particular, we note that due to the subjective and abstract composition of the concept of informational privacy, especially in the realms of the social web, a certain level of threat may be unmitigated. In this thesis, we focus on operating in this specific area by designing privacy preserving solutions that would actively engage concerned stakeholders (i.e., users of the social web). A notable challenge is to design solutions that are secure, usable, efficient, but mostly transparent. Offering transparency would highlight the risks that users may have to run and would facilitate accountability.

A data sharing environment operated based on transparency and accountability has the potential to give back control to data owners, to promote open and uncensored sharing of information, and to empower legislation with technical tools that might aid its enforcement in action.

1.3 Essential Approach

In 1982, Edsger W. Dijkstra provided one of the most insightful essays on Computing as a Science and on scientific approaches to the design and implementation of software solutions. Titled "On the role of scientific thought", the paper defined and defended the idea of *separation of concerns* that would later have a significant impact on how computing technologies would be approached and designed:

> We know that a program must be correct and we can study it from that viewpoint only; we also know that it should be efficient and we can study its

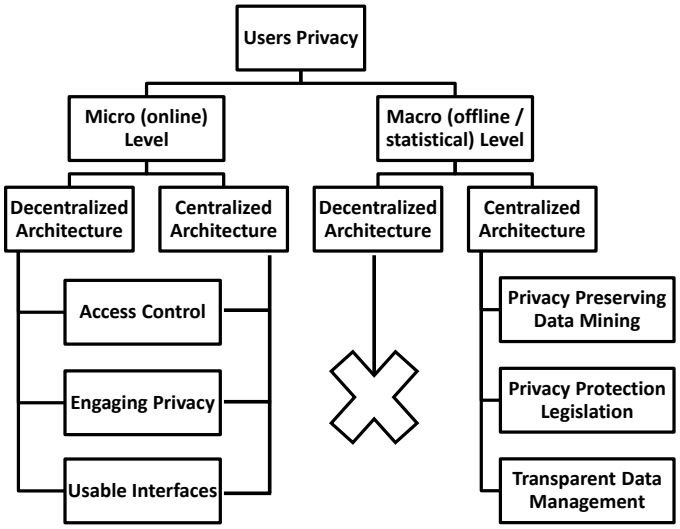


Figure 1.1 Micro (online) vs Macro (offline/statistical) Privacy under Centralized and Decentralized OSNs

> efficiency on another day, so to speak. In another mood we may ask ourselves whether, and if so: why, the program is desirable. But nothing is gained, -on the contrary!-, by tackling these various aspects simultaneously. It is what I sometimes have called *the separation of concerns*, which, even if not perfectly possible, is yet the only available technique for effective ordering of one's thoughts, that I know of. [43]

In practice, this idea has been adopted in most computational designs. We find it reflected in the Open Systems Interconnection (OSI) basic reference model for computers networking, in the separation of style from content in the development of web pages through the two standard protocols, the eXtensible HyperText Markup Language (XHTML) and Cascading Style Sheets (CSS), in the structure of largely used language paradigms such as object-oriented programming, etc [82].

Towards achieving the objective stated above, our methodology is inspired from this idea of separation of concerns. As a first step, and in order to better understand the composition of privacy concerns in the social web, we provide the decomposition suggested in the chart in Figure 1.1. The chart starts from contrasting the two high design levels for structuring humans online socializing, centralized and decentralized. The decentralized

design may be best descriptive of the pre-technology era of humans socializing; whereas the centralized design reflects the structure of major and successful OSNs available to today. At an informational privacy abstraction level, we differentiate between what we termed *macro-privacy* and *micro-privacy* under each of the centralized and decentralized designs. As on the chart, privacy concerns at the micro-level are shared between both the centralized and the decentralized designs. On the other hand, macro-privacy challenges are available under the centralized design but are, as discussed earlier, cut away under the decentralized model. As per our defined objective, our sphere of action is defined under the decentralized model. More precisely, we target the access control problem as a separate concern within this privacy preservation stack.

The general problem of access control has been extensively studied in the literature [50]. Solutions to a number of research problems in this area that have been already modeled and largely accepted should be explored and considered. Firstly, the access control problem is commonly defined in the literature as the process by which a requesting entity (i.e., *subject*) is granted or is denied access to a requested entity (i.e., *object*). As such, our problem should be modeled as *the process by which a requesting user in the DOSN gains access to allowed objects.* Secondly, as a further level in the separation of concerns, the access control problem has been decomposed in the literature into three different steps: 1) identification, 2) authentication, 3) authorization. Therefore, we start by studying the current literature to elicit the implicit techniques that would make the cornerstone of our methodology. The methodology follows the three identified steps of access control and extends some of the existing approaches to cover and fit the target scenario of DOSNs, and to answer the specific requirements defined in our objective.

Accordingly, we first address the *identification* step taking into account the set requirement of engaging users and of empowering them with tools to better understand and control their privacy. Thereafter, we explore the design of *authorization* mechanisms that are inspired from access control in other domains and adapted to serve our objective.

1.4 Terminology

To allow a smooth reading experience, we provide definitions to the key concepts that would be utilized throughout this document. These definitions are given based on the common understanding of the terms in the literature, but are also contextual to their usage in this thesis.

Online Social Network (OSN): An OSN is an online system that allows people to identify themselves by means of creating a profile that they can use to establish connections with other profiles in the system. The aim of an OSN is to provide a platform for socializing and data sharing between connected profiles. An OSN can be generally modeled as a graph

of users (i.e., nodes) where each established connection between two users in the system make an edge in the graph.

Decentralization Online Social Network (DOSN): A DOSN is an OSN that is designed following a peer-to-peer fashion. That is, every user holds and manages her/his data locally. The exchange of the data takes place in a direct peer-to-peer manner. Ideally, every user can only be aware of the connections she/he made in the network and no entity, either internal or external to the DOSN, may be able to have the full picture on all the users and all their connections.

User: A user is a member of an OSN and owner of a profile in it. A user is considered the ultimate owner of the data she/he creates in the OSN. We may refer to a user as a subject.

Access control: Access control is the mechanism, or set of mechanisms, that allows users to specify their privacy preferences and that ensures, at the same time, their enforcement in the OSN.

Accountability: In the context of this thesis, accountability refers to the quality of providing technical evidence that may prove whether a user had access to a given piece of information or not.

Transparency: We mean by transparency the value of making the access control mechanism adopted in the OSN available and easily understandable by users.

Identity validation: Identity in the scope of our work refers to the association of a profile in a social networking site to a reliable trustworthy entity, that truthfully owns and manages the profile. Identity validation is thus the process by which the trustworthiness of a profile, as a measure of its association to an honest owning and managing entity, is estimated.

1.5 Main Contribution

In summary, the studies, analyses, and models developed in this thesis provide the following research contributions:

- A methodology to study and to evaluate the construction of personal and community identities in social networks from a profile perspective, and in a collaborative fashion. It offers an identity validation framework for social network profiles that analyzes the constitution of the identities they reflect from a community collective trustworthiness perspective. The framework bases on its analysis to provide a model that offers guidance to users in assessing the validity and the reliability of profiles they interact with. The model is first studied and constructed under the assumption of a profiles central storage point, to test and prove its effectiveness. It is further scrutinized from a privacy preservation perspective to meet defined privacy and quality of service guarantees. It is

thereafter remodeled under a purely decentralized design, that does not assume any central point of control neither of data storage.

- Design and implementation of a an aposteriori access control model that bases on transparency and accountability as regulated by an accompanying audit process for decentralized social networks. The model calls for a new way of approaching access control in the social web promoting an open sharing environment of trust, regulated by transparent policies that are not pre-enforced but post verified for accountability enforcement. The key principal is to offer an alternative to access control under a decentralized setting that would be more efficient, more usable, and better scaling compared to commonly adopted cryptographic approaches.

- A label-based technique to manage privacy settings and preferences in OSNs that considers not only the relationships between users in the network but overpasses them to capture interactional aspects and their resulting relationships between created data objects. Considering the connections between objects in the definition and enforcement of access control in social networks allows easy modeling of the different levels of multi-ownership of the connected data elements. As a result, users may enjoy a larger spectrum to express more privacy preferences both in terms of granularity and easiness of management.

1.6 Thesis Organization

This chapter laid the ground for the motivation and objective behind the work reported in this thesis, and summarized its key contributions to the literature. The primary chapters of the thesis are outlined as follows:

Chapter 2 - This thesis has been developed within the framework of the EU Marie Curie Project iSocial[2] for the development of a DOSN. In this chapter, we present the wrapping project, we discuss its motivation, and we present background information about DOSNs and their status quo. We therefore position the work reported in this thesis within the context of the project.

Chapter 3 - Access control, related identity validation concerns, and privacy preservation techniques have known considerable attention from a research perspective. It is thus essential to review the related literature, that this chapter is dedicated to. This review also allows positioning our work and defining its fundamental concepts.

Chapter 4 - Starting from the first step in access control, this chapter describes and discusses the proposed framework for the validation of identities in social networks from a profile values perspective, and by reliance on collaboration between users.

[2]http://linc.ucy.ac.cy/isocial/

Chapter 5 - Focusing on the usability, efficiency, and scalability requirements for access control in social networks, especially under the decentralized approach, this chapter provides the solutions we have designed to provide aposteriori and label-based access control discussing how they address the specified target requirements.

Chapter 6 - To sum-up this thesis, we recapitulate its main arguments and contributions, we acknowledge and discuss its oversights, and we open doors on possible following future directions.

1.7 Related Publications

Published papers

- Bahri, L., Carminati, B., and Ferrari, E. (2014). Community-based identity validation on online social networks. In *Proceedings of the 2014 IEEE 34th International Conference on Distributed Computing Systems (ICDCS)*, pages 21–30. IEEE
- Soliman, A., Bahri, L., Carminati, B., Ferrari, E., and Girdzijauskas, S. (2015). Diva: Decentralized identity validation for social networks. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*, pages 383–391. ACM
- Bahri, L., Carminati, B., and Ferrari, E. (2015b). What happens to my online social estate when i am gone? an integrated approach to posthumous online data management. In *Proceedings of the 2015 IEEE International Conference on Information Reuse and Integration (IRI)*, pages 31–38. IEEE
- Bahri, L., Carminati, B., and Ferrari, E. (2015a). Cards - collaborative audit and report data sharing in decentralized social networks. In *Proceedings of the 2015 IEEE International Conference on Collaboration and Internet Computing (CIC)*. IEEE
- Bahri, L., Soliman, A., Squillaci, J., Carminati, B., Ferrari, E., and Girdzijauskas, S. (2016b). *BeatTheDIVa* - decentralized identity validation for online social networks. In *Proceedings of the 2016 IEEE 32nd International Conference on Data Engineering (ICDE)[Demo Track]*. IEEE
- Soliman, A., Bahri, L., Carminati, B., Ferrari, E., and Girdzijauskas, S. (2016). Cadiva - cooperative and adaptive decentralized identity validation model for social networks. *Social Network Analysis and Mining*, To appear

Papers under conference review

- Bahri, L., Carminati, B., Ferrari, E., and Lucia, W. (2016a). Lamps - label-based access control for more privacy settings in osns. In *Submission process*. Under revision

Papers under journal review (2nd round)

- Bahri, L., Carminati, B., and Ferrari, E. Coip - continuous, operable, impartial, and privacy-aware identity validity estimation for osn profiles. *ACM TWEB*

Chapter 2

Background - DOSNs and the iSocial Project

The work achieved in this thesis has been conducted as part of a larger research project targeting decentralized online social networks, named *iSocial*. *iSocial* is a Marie Curie Initial Training Network (ITN)[1] funded project for a duration of four years starting from October, 1st, 2012. *iSocial* is wheeled by an academic-industrial consortium of seven full partners and six associate ones, and embraces sixteen researchers out of which eleven are early stage researchers and five are experienced researchers. In addition to training the engaged researchers, the project's main objective is to produce cutting-edge research that would advance the status quo of DOSNs.

2.1 Why DOSNs?

Although the web has initially been conceived as a decentralized network of connected machines, services, and individuals, most of the successful and widely adopted applications built on top of it are centralized [102]. Centralization means that one entity owns and operates an application through which it offers an online service package that is available for consumption by other users of the web. Centralization here is mostly logical and is not to be confused with distribution. Indeed, with the increasing needs for higher performance and versatility, it is common practice that service providers may adopt distribution techniques of their applications using multi-servers architectures for performance and scalability improvements [83]. However, the control and the management of these distributed data centers is still owned by the one central provider.

Centralization owes its success to its underlying business model. Consuming a web service from one centrally controlled entity results in it collecting all possible types of data from the consumers. This data, in an era of information power, is monetized via advertising,

[1]The iSocial's official webpage: http://linc.ucy.ac.cy/isocial/

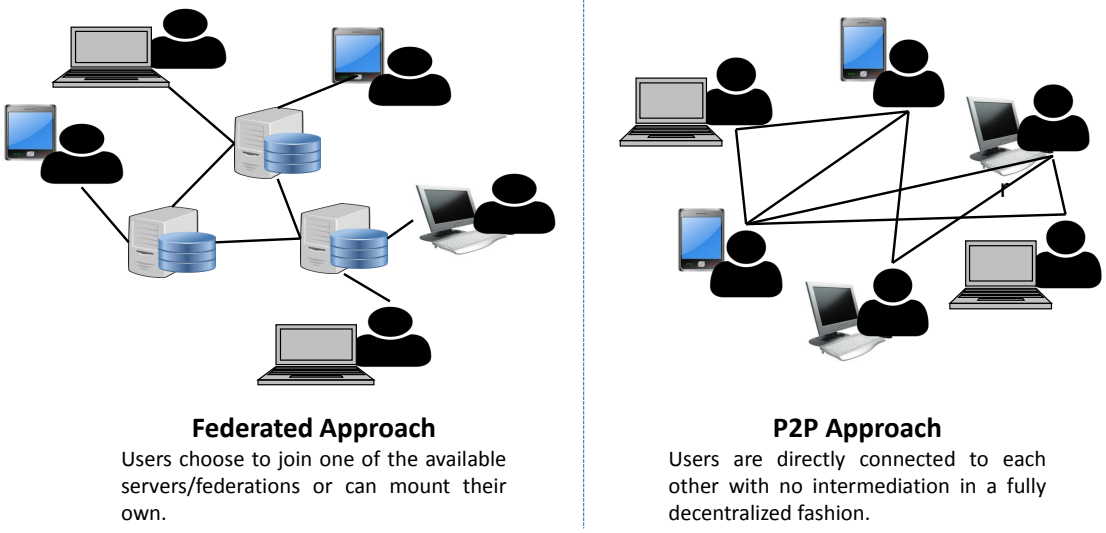


Figure 2.1 Federated vs p2p architectures for DOSNs conception

users profiling, etc [145]. Due to the central control that it settles in the hands of service providers, centralization resulted in having these entities amass unprecedented dense and diverse amounts of data about the behaviors, trends, and personalities of their consumers [102]. This calls, as we also briefly presented in Chapter 1, for major concerns about consumers privacy from a macro-level [69, 102, 145, 38].

Privacy and consumers protection advocates have therefore called for solutions to mitigate or to control these serious threats to individuals and societies "right to be let alone". Two main complementary streams of actions have been mainly noticed [62]. The first one defended that if centralization is to be maintained, proper legislation with follow up and control facilitating tools is required. Whereas, the second one called for alternative decentralized architectures as a pure and transparent response to the dangers imposed by centralization.

As an answer to the second paradigm, considerable research effort has been employed to produce a variety of decentralized proposals such as personal data stores, vendor relationship management, etc [102]. Given the importance of the amounts of personal data generated in their realms, OSNs, as one of the widely used and most successful centralized services of the web, could not escape from this argument or prove innocence regarding users privacy concerns. The research community has thus called for the study and the consideration of the alternative DOSNs, with the aim of making users regain full control and management over their data [38, 145, 39, 26, 122].

2.2 DOSNs Status Quo

The literature gives credit to research efforts that have already approached the conception and the design of DOSNs. Most of the available works in this direction would take one

Table 2.1 Federated and p2p DOSNs Pros and Cons and main available proposals

Approach	**Pros**	**Cons**	**Proposals**
Federated	Controllable QoS: availability, reliability, friendly service, etc	Might fall into centralization like design	Diaspora [22], SoNet [120], MANTLE [48], PrPl [121], etc.
p2p	No macro-privacy issues	QoS may be hardly manageable (including micro-privacy)	Safebook [39], PeerSon [26], Vis-a-vis [122], DECENT [71], CACHET [106], etc.

of two approaches (see Figure 2.1): 1) *a federated architecture*, or 2) a fully decentralized *peer-to-peer (p2p) architecture.*

Under the first approach, the main functionality of the DOSN (e.g., data storage, contacts search service, data exchange, etc.) is provided by servers (i.e., federations) that are deployed in the network. Users may decide to join one of the available federations, as they may choose to mount their own server that would also be an additional federation in the network that other users may potentially choose to join. Federations are connected to each other, ensuring the bridging between all users in the network regardless of which server/federation is serving them.

Under the p2p approach, every user in the system is modeled as a critical node in the network that contributes with its resources to provide and maintain the DOSN's functionality.

Designing decentralized systems is commonly known to be a challenge of trade-off between pure decentralization and quality of service (QoS) issues, such as availability, reliability, timeliness, etc [102]. The federation approach clearly gives up full decentralization, as the federations are centralized entities in the system, in favor of ensuring QoS of the DOSNs functionality. On the other hand, the p2p approach remains faithful to full decentralization, but still faces open challenges regarding the management of core functionality of the DOSNs service provision.

Table 2.1 provides a summary of the pros and cons of each of the federated and p2p approaches and lists the main proposals conceiving a DOSN under each one of them. As we can retrieve from the table, the federation approach makes it easier to manage the DOSNs functionality and allows the provision of a friendlier service to the users who would not be obliged to assume local management of their data and contribute with their device's computational powers. Instead, a user may simply join one of the available federations and enjoy the service. This could be seen as still a form of centralization especially in the presence of few federations that dominate most of the membership to the DOSN. However, from a privacy perspective, this still remains a better option than one single central authority. The idea is that a user may still have the control to choose between the available federations; thus, create a competition force among the federations to provide the most transparent, secure,

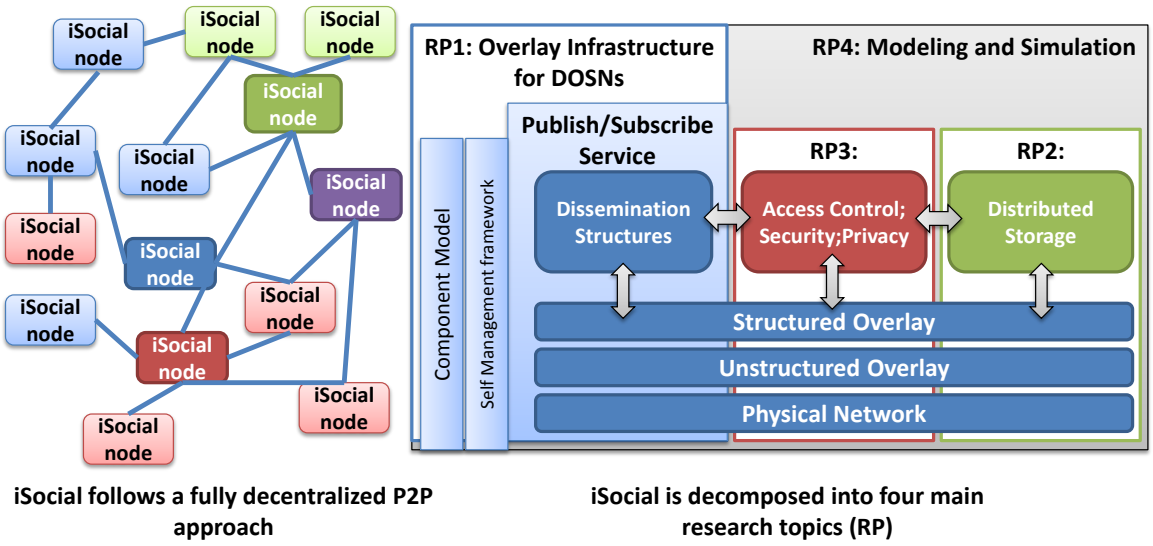


Figure 2.2 iSocial project p2p architecture and its four core research topics [iSocial Dissemination Documents]

and privacy aware service possible. The p2p approach, on the other hand, may eradicate these privacy concerns, and at the same time may introduce major inconveniences in terms of ease of adoption and QoS provision.

Most of the DOSNs proposals available in the literature have remained on paper only. The few ones that made it to deployment are mostly operated under the federated approach. The most prominent of these deployments in the Diaspora DOSN. This system knew relative success, especially when it was first launched, but still lags behind the fierce competition of the dominating centralized services, such as Facebook and Google+ [22]. To our knowledge, no fully decentralized p2p DOSN is available online for consumption. Moreover, no complete solution for a pure p2p DOSN that answers all QoS issues and provides most of the basic functionality expected from an OSN could be found.

2.3 iSocial Architecture and Components

iSocial project positions itself within the p2p approach with the aim of addressing some of the open challenges facing the conception and realization of p2p DOSNs. As visualized on Figure 2.2, the project has been decomposed into four research topics that it aims to address. In what follows, we provide a general description for what is intended from each of these four research topics:

- **Overlay Infrastructure for DOSNs:** this research topic concerns the study of appropriate overlays that would provide the base model on top of which information is

shared, searched, located, and retrieved. The main challenge under this topic is the establishment of a logical connection model based on which nodes can be aware of each other and communicate with each other.

- **Data Storage and Distribution:** the focus under this area is on ensuring the availability of data whenever requested. Under the most restrictive view of storing data only locally at each node, data availability would know serious limitations. The node may occasionally go offline, or more seriously may encounter failure problems that may result in corrupt data. Therefore, replication is required. The open challenges are w.r.t how to replicate, where, when, and under which security guarantees.

- **Security, Privacy and Trust:** major open research questions are hanging under this area of security, privacy, and trust (S.P.T) within decentralized systems in general and DOSNs in particular [125]. In the absence of a central manager, the control over the system is dispersed, and collaboration between all peers is required. However, for the specific issues of S.P.T, there is often conflict of interest between the collaborating peers. For instance, one peer that is required to collaborate for achieving a given privacy requirement for a piece of data should at the same time not be able to learn about its content.

- **Modeling and Simulation:** creating models and simulations might be required for validation and integration purposes of potential results achieved from the above research topics. Moreover, this activity allows the understanding of the dynamics of DOSNs in terms of bootstrapping, evolution, and maintenance.

The work presented in this thesis is performed under the third research topic on security, privacy, and trust. We briefly discuss this topic in the following section.

2.4 Security, Privacy, and Trust

Under the third research topic of iSocial, the concern is on security, privacy, and trust challenges. These concepts are often confused and might be understood as synonyms; however, from both pragmatic and theoretical views, these three concepts make different concerns, but not without fine and sensitive crossovers among them. To better explain these three concepts and the relationships between them, we suggest the representation on Figure 2.3.

We consider trust to be a function of achieving both security and privacy guarantees. Given that security and privacy cannot be practically met to perfection, we consider that proper risk management with defined recovery processes and insurances against failure is a critical component of achieving trust.

Figure 2.3 Privacy, Security, and Trust: Differences and Crossovers

Regarding security, it is commonly defined in the literature as a "C.I.A" triad of which the edges are *Confidentiality*, *Integrity*, and *Availability* (C.I.A) [36]. Confidentiality refers to the value of preventing information from reaching the wrong entities, while at the same time ensuring that it gets to the right entities [36]. Confidentiality is generally managed by proper access control that ensures both the formulation of confidentiality requirements and their enforcement in the system. Integrity contains maintaining the consistency, accuracy, and trustworthiness of the data at all times; whereas availability refers to the quality that the data is available and rendered by the system whenever requested.

Although confidentiality is a mainstay in the construction of privacy, we believe that there is much more to privacy than simply assuring proper access control. With reference to the discussion we presented in Chapter 1, privacy at the macro-level, for instance, surpasses the insurance of confidentiality of the data to providing transparent guarantees on its ownership, management, and usage at all levels of abstraction. As such, we represent privacy as a triad of *Transparency*, *Confidentiality*, and *Control* (T.C.C). By transparency, we refer to the quality of providing data owners with clear non confusing policies and accountable processes regarding all the stages of storage, management, movement, and sharing of their data. As for control, it refers to the value by which data owners have full control over their data in terms of retrieval, deletion, etc.

All these three concepts still make open research challenges especially within DOSNs p2p designs. As it should be clear by now, the contribution of this thesis is w.r.t the privacy component. We mostly focus on confidentiality, with also consideration of incorporating the control and the transparency factors.

Chapter 3

Review of Literature

We conclude from Chapter 2 that privacy is a triad of transparency, confidentiality, and control, and that it shares with security the confidentiality component. Confidentiality of information is one of the first topics to be addressed and to attract attention from an informational security and privacy points of view, especially within military systems where the power and sensitivity of information were first comprehended and valued [113]. Ensuring the confidentiality of information is managed by what is known in the literature as the *access control* problem. Access control has been extensively studied and the literature provides an ample of research results, collections of best practices, and well established models for approaching the access control problem under different requirements. In this chapter, we take the challenge of exploring the diversity and richness of works under this area and summarize the most prominent contributions to the access control problem independently from systems where it might be required (Section 3.1). After laying the foundation for the access control problem in general, we smoothly move to exploring how previous results and models have been adapted for the OSNs scenario, either actually deployed or suggested in research works on the shelf (Section 3.2). We also provide a discussion on the limitations of these proposals and on the major threats to access control in OSNs, especially those that this thesis is concerned with.

3.1 Access Control

Access is the flow of information from a passive entity that holds it (i.e., an object) to an active entity that requests it (i.e., a subject) [36]. Access control is the process by which this flow is regulated such that information contained in an object is granted to authorized identified subjects only and is denied to unauthorized or unidentified ones. Starting from this definition, we may conclude that access control should be composed of at least two steps. First, *subjects need to be identified* as to know who is requesting the information. Second, granting or denying access should logically be the *result of evaluating a given request based*

on some identified rules or policies. To make things more formal, the access control problem is known in the literature to be a process of three main systematic steps [36]:

- *identification*: refers to the mechanism by which subjects are identified as who they claim to be. In information systems, this could be mapped to an ID, a username, or a value that uniquely refers to every subject in the system.
- *authentication*: is the method that proves that a subject who claims an identity does actually own it. For instance, this could be proving authenticity of owning a username in the system by providing the corresponding assigned password. There is a wide range of techniques and technologies used to solve the authentication problem. They are categorize as something that the subject is (voice, facial or retina patterns recognition, fingerprints, biometrics, etc.), knows (PIN, password, etc.), or has (access cards, token, key, etc.)[36].
- *authorization*: is the process that actually evaluates the access request made by the identified and authenticated subject to the requested object.

Given that the authorization step is what actually results in the final answer to an access request, access control may be shortened as an authorization problem. In fact, the identification and authentication steps could be addressed as separate concerns that should be managed prior to the core step of access control; i.e., authorization. Throughout the remaining of this document, access control refers mainly to the authorization problem.

Access control has been explored and exploited even before technology. Some of the oldest conceptualizations of access control might be the construction of physical locks behind which something is secured, with access selectively restricted by means of owning the right unlocking key, the concept of security deposit boxes in banks or other established institutions, etc. With the emergence of information management systems, similar mechanisms were required to manage the selective restriction access to sensitive information. As information systems continued to develop, in terms of architectures, scopes, etc., access control, as one of the core building blocks of any information system, has continued to attract considerable research interest.

From an informational abstraction level, the essence that could be extracted from the related literature may be best represented as a methodology that walks us through a series of fundamental questions. First, it is required to comprehend the basic components of the task to be performed (i.e., access control) (Section 3.1.1). Second, comes the issue of defining who is controlling; that is, the model based on which the control is to be performed (Section 3.1.2). Once these are clarified, it is finally required to answer how the control is going to be enforced (i.e., deployment and enforcement mechanisms) (Section 3.1.3).

3.1.1 Basic Components

Access control is the mechanism in an information system that manages access decisions regarding access requests made by identified *subjects* (users, processes, or programs) to protected *objects* (information, processes, program). An access request is evaluated based on defined *policies* in the system that provide high level descriptions, mostly as administrative or preference clauses in a human friendly language, of the data access requirements to be imposed [50]. These policies are typically specified by a system controller or *administrator*, or by the *owner* of the object in the system. For instance, in a school management system, the school administration may generate an access policy that states that "grades of a course are to be viewed only by teachers giving the corresponding class and only while the class is active."

In order to enforce these policies in the system, each one of them is modeled as a set of deterministic *authorization rules* based on which access decisions are automatically made. An authorization rule might be abstracted as a structure that expresses a set of permissions that a subject is allowed to perform on an object [50]. These permissions, or authorization rules, may be translated and enforced using a variety of technical solutions (see Section 3.1.3); however, it is first required to specify who will define them and under which model.

3.1.2 Who controls? - Models

An access control model is a logical framework that shapes the general scope of how subjects access objects and who owns the controls [36]. There are three main traditional access control models in the literature: discretionary, mandatory, and non-discretionary (also known as role based) [50].

- **Discretionary access control (DAC)**: DAC is a user-centric access control model in the sense that data owners are who determine the access policies related to their owned content in the system. In DAC, there is usually no imposed central control or pre-specified rules on data owners. Data owners specify access policies and rights at their discretion, making of DAC a flexible model that could be adopted in distributed system settings. However, one of the drawbacks of the DAC model is its inability to control information flow. That is, when a user has the right to access an object, it may not be possible to control how this user could leak that object to other users who are not authorized to access it.

- **Mandatory access control (MAC)**: in MAC access is centrally controlled. That is, a system administrator with a supreme role defines the access rights allowed to each other user in the system. System users (i.e., not the central authority) cannot change the access rights they have on an object, as they cannot determine access rights to objects they themselves create in the system. Instead, the centrally specified rights

define and impose the access rights to be assigned to these objects based on the initial rights associated with their creators. As such, information flow is statically controlled in MAC, such as the access control system monitors the ways and types of information that are propagated from one user to another. This is typically achieved by classifying users and objects into ordered security levels or classes. All the valid channels along which information can flow between the classes are regulated by means of established order relationships between these classes as defined by the central authority in the system.

- **Role-based access control (RBAC)**: RBAC could be considered as a sort of middle ground between MAC and DAC. In fact, on the one hand, DAC could be viewed as too flexible and on the other hand, MAC could be too rigid. Therefore, RBAC combines both MAC and DAC in a trial to offer some flexibility to data owners all while being under some sort of central control. This is achieved by designing roles, that are typically a job function or an authorization level, that users could take in the system and that allow certain privileges w.r.t some resources. RBAC's flexibility comes mostly from the fact that users can be assigned several roles and a role can be associated with several users.

All these models are focused on the relationships between subjects and objects only without considering other environmental or contextual parameters. As access requirements in information systems continued to evolve, other models have been conceptualized. However, most of them could be decomposed either as hybrid combinations of DAC, MAC, or RBAC (e.g., label based access control -LBAC- model), or as extensions of one or more of those three traditional models. For instance, the newer attribute based access control (ABAC) model could be viewed as a generalization to MAC and DAC that adds the possibility of considering descriptive attributes related to subjects and objects and to the environment relevant to an access request [1]. That is, ABAC is capable of modeling both MAC and DAC with the possibility of modeling other contextual parameters, such as time of the day, location, etc.

Another example is the LBAC model that could be viewed as a combination of MAC and DAC [36]. in the LBAC model, a labeling strategy is deployed to specify access control requirements on both objects and subjects in a MAC like fashion; however, these labels could be specified at the discretionary of data owners under what is allowed by the system administrator. As such, LBAC benefits from the information flow management that MAC offers, whilst at the same time enjoying the flexibility of DAC. One of the examples of LBAC adoption is by Oracle databases.[2]

[1] ABAC Overview: http://csrc.nist.gov/projects/abac/

[2] Oracle Label Security Guide: http://docs.oracle.com/cd/B19306_01/network.102/b14267/intro.htm

Table 3.1 An access control matrix example

	$o1$	$o2$
Bob	{view, share}	{view, modify}
Mike	{-}	{view}
John	{view}	{-}

In general, an access control model specifies the high level logical view on how information flow in the system would be managed. To concertize a model, different deployment mechanisms could be exploited to offer practical methods for both the formulation and the enforcement of access requirements. In the following subsection, we provide a brief discussion on common deployment techniques as associated to each of the three basic models discussed above.

3.1.3 How and where to enforce control? - Deployment and Architecture

Under the DAC model, control is typically defined and enforced by means of access control matrices (ACM). As DAC is a user-centric model, ACMs are structures defined per data owner and are used to keep a global picture of the other users in the system and the access rights the data owner assigns to them w.r.t her/his owned objects [75]. An ACM is a matrix that organizes all the users and all the objects in the system that concern a given data owner and specifies for each intersection of a user u and an object o the access rights that u is granted to perform on o. For example, assume that Alice owns two objects $o1$ and $o2$ in a file management system and there are three other users in the system, Bob, Mike, and John. Alice's access requirements w.r.t her owned objects could be represented in a sample ACM as pictured on Table 3.1.

There are other possible variations for the implementation of ACMs such as access control lists (ACLs) and capability lists (CLs) [75]. A nice analogy to explain the difference between ACLs and CLs is provided in [75]:

> A capability list is analogous to a ticket for a concert, that is, a user with a "ticket" (access privilege) is allowed entry into the concert hall (file). The capability list (CL) specifies privileges of access to various files held by a user. The access control list (ACL) on the other hand, is analogous to a reservation book at a restaurant (file) where a customer (user) is allowed seating in the restaurant only if his/her name appears in the reservation book.

Similar constructions of ACMs or ACLs could also be used to implement RBAC based systems, by simply substituting users with roles. However, in RBAC, role permissions are assigned to specific operations with a specific meaning within an organization, rather than to low level files [75].

Under the MAC model, access control is usually managed by means of security labels that are assigned to both objects and subjects. One of the early practical implementations of the MAC model, if not the first, is the work presented by Bell LaPadula, known as the BLP model [75]. In BLP, subjects and objects are annotated with security levels on a defined ordered scale that express objects sensitivity levels and subjects access privilege or clearance levels. More precisely, objects are classified as top-secret, secret, confidential, or unclassified, and subjects are associated with corresponding clearance levels. The confidentiality of information and the control of its flow are then protected by two principles: 1) no read-up and 2) no write-down. The first principle enforces that subjects can only read objects that are at most equal to their clearance levels; whereas the second states that subjects write information only to at least equal levels than their owns. This ensures that information can never flow against the levels set in the system, as subjects having access to confidential information cannot, by mistake or maliciously, make it available to other subjects at lower levels.

Other implementations of the MAC model have been proposed under what has been known as multi-level security (MLS). Examples of these include the Biba Integrity Model, Chinese Wall, and Clark-Wilson models [75]. In addition to these implementations, cryptographic solutions have also been considered for the implementation of MLS systems that would provide security of data under different contexts, such as when data outsourcing is required [75, 36].

The non-repudiation principle

As we have solicited thus far, access control deployment mechanisms focus on keeping data under different types of control allowing only authorized entities to get their way to it. However, a complete access control solution should also provide detection and recovery mechanisms for potential failure instances of the imposed controls. This is known in the literature as the *non-repudiation principle*, or the ability of the access control process to prove accountability of who accessed what in the system and under which conditions [36]. This is commonly achieved by the establishment of a logging mechanism that keeps track of granted and denied access records. These records might server for auditing purposes such that problems could be detected and recovery actions could be taken [36].

3.2 Access Control in Social Networking Sites

OSNs may be abstracted as graphs made of nodes and a set of edges modeling the connections between them. In this abstraction, the OSN users are the nodes in the graph and the friendships established between the users are the edges. In today's popular and commonly

used OSNs[3] relationships are established only if jointly articulated.[4] That is, an edge between two users Bob and Alice is added to the network only when both Bob and Alice jointly express consent to the relationship. However, in the graph abstraction of an OSN, both directed and undirected variations of graphs may be considered [18]. Without loss of generality, we consider in most of our following discussions the undirected graphs abstraction.

In OSNs, access control is commonly approached with a relationship-based model (ReBAC) that explicitly tracks the interpersonal relationships established between users in the network and allows the formulation of access policies based on them [50].

Although ReBAC has been presented in the literature as a separate access control model, that was first coined as such in [55], it could be seen as a hybrid between the DAC and the RBAC models. First, OSN users, being the data owners, specify, at their discretion, the access policies for their data. Second, these policies could be mainly expressed in terms of the relationships between the data owner and the data requestor in the OSN. Satisfying a specified relationship could be viewed as holding the role corresponding to it. For instance, an OSN user, Alice, could specify an access policy for a photo requiring that only people with whom she has a family relationship in the network could access the photo. Therefore, all people in the network who hold the role of a family member of Alice would be authorized to access the photo. The ReBAC model could thus be considered as a form of DAC combined with a user-centric RBAC.

Most of the mechanisms and techniques that have been suggested for achieving ReBAC in OSNs would fall under one of two categories: trust-based or encryption-based. The trust-based approach has mainly been explored under the centralized design of OSNs where a central entity has full knowledge on the network's graph including its nodes, edges, and data ownership. On the other hand, the encryption-based approach has been mostly investigated to address the access control problem under the DOSNs scenario. We review proposals under each of these approaches in the following sub-sections.

3.2.1 Trust based methods

The suggestions under this category build their common rationale on controlling information flow based on the trust users have in each other. This trust may be as basic as establishing a connection/friendship between two users, as it may stretch to impose other parameters on these relationships.

The most basic method would be the consideration of simple friendship connections, such as friends or friends of friends, for the formulation and enforcement of access policies. That

[3]Facebok and Google+ could be considered as the two main giant OSN service providers monopolizing the OSNs arena.

[4]In Google+, it might be possible for user A to be in a circle of friends of user B without link reciprocation; however, this would generally not be reflected as a friendship but more as a *follower* directed relationships. We do not consider this type of relationships in the scope of this work.

is, access policies could be formulated and enforced based on whether or not a friendship link between data owner and requestor exists [55]. An example of these is the work presented in [79]. The authors suggest a friend of a friend (FOAF) ontology-based identity management system to manage authorizations and access rights checking. In their system, relationships are enhanced with a trust value that indicates the strength of friendships between users connected with a given link in the OSN.

On a second level, a notion of groups or lists may be introduced to offer relatively more flexibility for users to express their access policies [1]. In addition to expressing access policies based on whether or not a friendship exists between the data owner and other users (and possibly its corresponding trust), users have the possibility to organize their friends in groups or lists. Information will then be shared with a user's friends based on the group membership they are assigned with. For example, Alice could create a family list and a colleagues list and express an access policy that states that a given information could be accessible only by friends under the family list and not by those under the colleagues one. This method reflects the way access control and privacy settings are actually managed in current major OSNs, such as Facebook and Google+. Given the amount of laborious work this method requires, as a user has to arrange all her friends into lists or groups, and given the confusion that users might face regarding the basis on which they could make this grouping, some works took the challenge of automating this task. For instance, the authors in [1] have explored users sharing and interaction activities with their friends to automatically identify different social circles within a user's friends base.

On a more stretched level, other variations introduce more flexibility by associating other parameters with the relationships. For instance, in [29], the authors have added more flexibility to the formulation of access policies by introducing three attributes to describe a relationship in the OSN. These attributes are type, trust, and depth. This means that the evaluation of an access request is based on the existence of a connected path in the OSN graph, between data owner and data requestor, that satisfies the conditions specified on its type, its trust, and its depth or length (i.e., distance, in terms of number of edges in the path, between data owner and requestor). For instance, Alice would have the possibility of specifying an access policy stating that her photo could be authorized for all people with whom she has a connection that is of type family, that is made of at most three hops of depth, and that has a trust of at least a given value tr. This assumes that all edges in the OSN are associated with a type and a trust values.

Other researchers in this area found it necessary to explore options for automating the formulation of access policies such that the privacy preferences task is performed by the system on behalf of the concerned users. Examples of proposals on this line could be found in [130, 18]. In [130], the authors suggest a semi-automated framework in which users privacy preferences are collected and are treated with a process that considers data sensitivity and risk of disclosure to generate tailored access rules. On the other hand, the authors of [18]

exploit interaction intensity between users to predict the quality of a given friendship link. Access policies are then automatically generated based on previous interactions and sharing habits between concerned users.

In other trials to offer more flexibility is expressing access policies, some proposals suggested the exploitation of notions of dual profit or exchange of benefits such that a user releases information to a friend based on how much data she receives from that friend. Herein, the notion of trust is bound to the achieved benefit from the relationship. One example is the work in [131] where the authors deploy game theory to manage the data release challenge between users. Their scheme targets mostly the scenario of co-ownership of content where the different stakeholders may impose different access requirements. Basically, co-owners are allowed to specify their access requirements separately and the Clarke-Tax mechanism is adopted to facilitate consensus between parties. One of the strongest critics to this work rises from the complexity of the underlying mechanism and its doubted suitability to the general OSN population. Indeed, in that model, users are required to comprehend the Clarke-Tax mechanism, to make bids, and win auctions in the process of making decisions about their data exchange. This would clearly make usability limitations to the adoption of the proposal.

Using the same concept of exploring game-theory based techniques, the authors of two different proposals, [133, 68], suggested negotiation protocols to reach compromise on conflictual access policies, that might be imposed by the different stakeholders in a data item, based on achieving a Nash equilibrium in the system. Nash equilibrium is a well known game theoretic solution concept that describes a system state where all parties have achieved equal benefits and no party has enough incentives to change its strategy [68]. The problem with such approaches is that users privacy preferences in real life might not follow the same categorical or ideal rational of game theories.

Following this thread of allowing flexible access policies for collaboratively owned content, the authors in [67] have proposed a multiparty access control model that allows the specification of access rules based on the roles that associated stakeholders hold w.r.t a shared content. More precisely, they have differentiated between four roles that a user could play as a controller of a shared data. Namely, a user could be the owner of the data, a contributor to it (such as a user creating a post on another user's profile), a stakeholder in it (e.g., users tagged in a picture), or a disseminator of it (i.e., sharing someone else data). Users can specify access policies based on each of these roles and access decisions are taken based on a simple majority voting mechanism whereby access is granted if the majority of the collaborators approve it.

Always on the same line of managing collaborative access control, the authors of [93] proposed an ontology based scheme that allows the management of multi-authority policies. This is basically supported in two simple ways: conjunctive and disjunctive multi-authorizations, such that access is granted if all collaborators agree to it or if only one of the collaborators approves it, respectively.

In addition to all these proposals, the literature is rich of many others in which authors try to model access rule specifications that could offer more flexibility to users in expressing their access control needs. Some of these examples are the model in [33] that extends the ReBAC model to cover relationships between users and resources and among resources as well. The model allows for the specification of access policies in terms of patterns in these relationship paths with the consideration of a hop count limit. For instance, a user Alice may be able to express an access policy that allows to reveal a data element to users in the OSN who are not necessarily friends with her, but who might be connected to her via a shared resource. A typical example is the case by which Bob is tagged on one's of Alice's photos but is not friends with her. Alice would still be able to allow Bob's access to other photos of hers over the resource connection path that the common picture creates between them. This model widens the scope for policy specification but might not be reflecting to the real needs of OSN users. For instance, with reference to the example above, Alice might not comprehend the existence of this connection between her and Bob that the common photo between them would have created in the model's extended graph. This might cause more confusion to the users and open paths for more unfathomable privacy breaches.

More recently, another research effort has resulted in a model that suggests the incorporation of public information about users as a parameter in the specification and enforcement of access control in an OSN [109]. Their motivation was that public data elements, such as information about countries, schools, public figures, centers of interest, etc., could be modeled as parameters based on which connections between users could be inferred. For example, users who are from the same country or who declare having attended the same school or worked at the same institution. The incorporation of public information as part of the access control scheme would allow users to express access policies beyond relationships among them and their friends. For instance, under this model, Alice may be able to specify an access policy by which she grants access to her location information to all those who are from the same country as her.

All these proposals could be considered as successful trials to offer more flexibility to OSN users in the expression of their access policies and the definition of their privacy preferences. However, the question remains on the extent to which they actually meet the needs of OSN users, as well as on their usability and comprehensibility by a standard user. Moreover, none of these models offer transparency guarantees to users and none of them incorporates accountability. As we have discussed earlier in this chapter (see the discussion on the non-repudiation principle under Section 3.3), accountability, or the ability to trace who got access to what and under which authorization, is a critical component of reliable, transparent, and complete access control.

3.2.2 Encryption based methods

As the name suggests, these methods would deploy encryption mechanisms to implement ReBAC in OSNs. These methods have mainly been explored for the scenario of DOSNs, since data cannot be managed and observed in a centralized database that would be aware of all the users and their relationships in the network, and hence allow the deployment of the proposals discussed above. In fact, in a DOSN, every user locally holds her/his data and is only aware of their direct friends. Moreover, data dissemination is carried out in a peer to peer manner, making as such every peer responsible of managing access to their own data or to the data of other peers replicated on their node. Here we note that it is common practice in decentralized architectures to have data replicated between peers to ensure higher availability and to allow for data recovery in the event of a node's failure [110]. Whilst most of the proposals for DOSNs would opt for encryption to ensure the security of the out-sourced data at the end of the other peers, other proposals might overpass the need for encryption by making replicas only at the level of peers who are allowed to have access to it [110]. However, under both scenarios, a mechanism for the enforcement of access control within the decentralized environment is required and almost all the solutions available in the literature deploy encryption based techniques. Only a small subset of DOSN proposals come without encryption and rely on trust in friends only. An example is the work presented in [103] where authors suggest that users select a circle of trusted friends whose nodes would be used for both data replication and access control management. Clearly, this model assumes blind trust in the chosen circle of friends. Such an assumption makes it easier to address the access control and data security problems in DOSNs; however, it bases on an idealistic view that might not be practical for all users in an OSN realms. Data encryption remains, thus, the common and practically applicable alternative.

Deploying data encryption to manage access control means that anyone could retrieve the encrypted content but only those who have the corresponding keys can interpret it. This implies that one of the requirements is to offer a mechanism for the distribution and management of the corresponding keys. Within the context of OSNs, characterized by massive amounts of data, huge users base, dynamic and frequently changing access requirements, etc., encryption keys management becomes an essential challenge [110]. Moreover, access scenarios in OSNs require fine grain levels of specification and span to cover relationships between users, between resources, and between users and resources. As such, the main technical challenges related to the implementation of access control using encryption in OSNs are related to supporting fine-grain flexible access policies on one hand, and to providing an efficient mechanism for the management of the related keys on the other hand.

Therefore, most research work under this area have focused on finding usable ways for the dissemination, management, and revocation of the security keys related to the deployed

encryption [70, 9, 39, 106], or on deploying cryptography techniques that can offer finer levels of granularity, such as attribute based cryptography [9, 106, 71].

Regarding the issue of keys management, most of the proposals in the literature assume that the exchange of all access keys (or fingerprints that could be used to retrieve the keys) between users happens outside of the system (i.e., out of band - OOB) [110]. This results in the inconvenience of finding an appropriate and trustworthy OOB channel through which this keys exchange could take place. In a trial to overcome this problem, some proposals suggested the consideration of trusted nodes or super nodes in the DOSN to act as credential authorities [39, 26]. These super nodes are only used to initiate the DOSN and are not implicated in the mediation of communication between nodes; hence cannot trace interactions. However, this approach still assumes the trustworthiness of these super nodes and their availability. As an alternative, some other works have suggested the exploitation of dynamic hash tables (DHTs) to include the management and distribution of keys (e.g., the proposal in [60]). This cancels the reliance on any central node, but does not provide any kind of identification. That is, the DHT is solely used to distribute the keys and the generation of identities requires to be managed with a different parallel service. However, the real problem and still open challenge with all these proposals is with the efficient management of keys revocation and/or update that should accompany the change of users access policies, especially given that such changes are so frequent and unpredictable in the realms of OSNs.

On the other hand, the problem of allowing the management of fine-grain and flexible access policies has been of more challenge to the division of an encryption based access control for OSNs. In general, three encryption types are exploited. These are, 1) public key encryption (PKE), 2) attribute based encryption (ABE), or 3) boradcast encyrption (BE) [110]:

- **PKE:** known also as asymmetric cryptography, PKE uses two kinds of keys: public and private. Public keys may be disseminated widely, whilst private keys are known only to the owner. In PKE, any person can encrypt a message using the public key of the receiver. Such a message can be decrypted only with the receiver's private key. In contrast to the more basic symmetric encryption, that relies on the same key to perform both encryption and decryption, PKE is considered to be more costly. As such, it is common to use group key management mechanisms based on symmetric keys. That is, one symmetric key is distributed among a group of recipients, making one symmetric encryption process sufficient to share a content with a group of recipients.

- **ABE:** Under this category, instead of relying on encrypting data for a given decrypting key or identity, an encryption is made by labeling the generated ciphertext with a set of attributes. To be able to decrypt the resulting ciphertext, the user shall meet those attributes.

- **BE:** Relying on either symmetric or asymmetric encryption algorithms, the idea of BE is to generate ciphertexts that could be decrypted using multiple keys that are defined by the deployed mechanism during the encryption process. The goal is to allow the sharing of one ciphertext with many recipients in a cost effective way (i.e., perform one encryption and distribute multiple decryption keys).

Given the requirements of OSNs, in terms of efficient management of revocation, cost efficiency in terms of performing as few encryptions as possible, etc., related proposals focused on exploiting and combining these types of encryption to devise cyrptographic mechanisms that are more suitable for the OSNs scenario. For instance, in [9], the authors adopted ABE where every user in the DOSN generates an ABE public key (APK) and a master secret key (AMSK). The notion of groups is then used to map to the attributes used in the encryption. As such, a user u would create for every friend f an ABE secret key (ASK) that corresponds to the set of attributes that express the groups that u assigns to f.

In [70], the authors adopted ABE and worked on offering support for dynamic group memberships and management of access right revocation without the need for reissuing keys or re-encrypting the data. They achieved this by introducing a proxy that needs to be contacted in order to be able to execute any decryption in the system. Users send the target ciphertext to the proxy that runs a transformation process on it. The transformed ciphertext can only be decrypted if the access right has not been revoked from the requesting user.

In [136], the notion of relationship attestations was suggested. That is, the system generates a certificate for every two users who established a relationship in the network. These certificates could then be used, across multiple platforms that support the suggested protocol, to prove the existence of the attested relationship.

To sum up, these proposals demonstrate that there is room for devising encyrption mechanisms that may be used to solve the access control problem in DOSNs. However, given the amounts of data and the number of connections users maintain in OSNs, and with the required granularity and changing requirements for privacy settings, encryption mechanisms, thus far, may not overcome their by-design inherent limitations, especially in terms of efficiency [26].

3.2.3 Limitations, challenges, and threats

All the proposals reviewed and presented above demonstrate that there are still open challenges with considerable margins of improvement that may be brought to the access control in OSNs in general, and in DOSNs in particular. We identify two main axes of action that we believe are of the most relevant challenges with regard to users experience in social networking sites: 1) flexibility and richness of controls available for users to express their data's access policies, especially for co-owned data, and 2) efficiency in access control enforcement in the absence of the central enforcement authority (i.e., DOSNs scenario).

Regarding the first point, the main focus in almost all available proposals has been to find ways that could implement the ReBAC model offering as much flexibility in access policies expression as possible. However, those implementations are all bound to relationships between users without fully considering relationships and hierarchies between data elements, and without covering the diversity of interactions available in OSNs. For instance, when Alice puts a comment on a photo shared by Bob, this "commenting" interaction results in the creation of a data element (i.e., Alice's comment) that is dependent on Bob's photo, but that is at the same time a separate object in itself with a separate owner (i.e., Alice). The grand majority of proposals available thus far do not consider the management of such types of information at an atomic ownership level,[5] rather this issue is majorly addressed from a co-ownership or multiple access control management perspective [131, 67, 133, 68]. That is, a co-owned object is treated as one integral block over which access policies of the different stakeholders are defined, and for which a generic conflict resolution mechanism is applied. This makes the suggested solutions complex for users, in the sense that a stakeholder in a co-owned object cannot guarantee the enforcement of her desired privacy policy on the portion she contributed with to the target object. Indeed, co-owned objects in OSNs are not blocks of inseparable elements, rather they are structures of linked pieces of information that are uniquely owned (e.g., a photo annotated with a like, a comment, and a tag). Decomposing the ownership of co-owned objects, and the consideration of the relationships between atomic object elements instead of focusing on relationships between users only, is expected to simplify the problem and make it possible for users to own full control over the pieces of information they contribute with in the OSN.

On this line, a work presented in [95], has taken a closely similar approach in terms of looking to co-owned data elements as compositions of what the authors have termed as annotations. The authors point to the same problem of allowing authors of an annotation to have a say on the privacy policy that protects the content. However, in their approach, an annotation would still inherit the stakeholders of its parent annotation, thus inherit the preferred policies of the other co-owners of the composite object. Moreover, the work is mostly presenting a high-level design approach, formulated as a set of design principles, to the problem of co-owned data. A proof of concept implementation is provided using relational databases by means of the creation of views that connect users to the annotations they are allowed to access, and only basic levels of privacy settings are offered (i.e., only me, friends, friend of friends, public).

To address this identified issue, our suggestion is made to consider an alternative to the ReBAC model by exploring the division of a MAC based system for the management of access control and privacy preferences in OSNs, at separate levels of ownership. We provide the details of this contribution under **Chapter 5, Section 5.2.**

[5] Although some works provide schemes for the specification of fine-grained access policies, such as the work in [93] where ontologies were exploited, these are managed from a multi-authority perspective.

On the second point, the omnipotent concern rises from the serious efficiency and scalability limitations that the cryptography based access control, commonly adopted for DOSNs, introduces. A straightforward alternative to cryptographic approaches is the exploitation of the trust based approach. However, under a decentralized setting, relying on trust only would leave the associated risks uncontrolled; thus, making the system unsafe. To address this, we explore the installment of accompanying guarantees on transparency and accountability. Motivated from this, we have worked on designing a framework for data sharing in DOSNs that relies on trust, but that is regulated with transparency and accountability by means of an installed auditing mechanism. We present this work under **Chapter 5, Section 5.1.**

3.2.4 The identification step and related threats

As we have seen earlier in this chapter, the general process of access control starts with an identification and an authentication steps before getting to the core of authorization management. All the access control models and methods discussed thus far assume a mechanism in the system by which subjects have been identified and authenticated. Authentication in OSNs is commonly managed using password verification, or identity keys verification. These techniques are generally vulnerable to account theft attacks that may exploit a variety of social engineering techniques to get hold of a user's password or identity key [63]. Acknowledging the importance of this threat, the vulnerabilities it may exploit are usually related to elements that are outside the boundaries of the system. For instance, the system might impose a strong password policy; however, it would not be possible to technically prevent the corresponding user from writing it down on a piece of paper, or from responding to phishing attacks[6] that a potential attacker could exploit, for example. However, identification in OSNs remains one of the vulnerable processes that attackers may technically exploit to masquerade as valid users, hence easily gain access to protected data.

In fact, identities in OSNs are very loose. To facilitate their adoption and encourage people to join them, only a valid email address is required for a user to create an identity in the OSN. The problem with this is that a user could claim any identity even if they do not actually represent it. From a system perspective, such fake identity accounts are correctly identified (i.e., they hold valid email addresses); however, from a user perspective they introduce unreliability in correctly identifying one's contacts in the network. For instance, a malicious user may claim to be Alice, a close friend to Bob, and fool Bob to accept a friendship with its fake account. The danger in this scenario is that this malicious user will be getting legitimate access, from a system's access control correctness perspective, to Bob's data that he intended to share with his friend Alice. Therefore, the access control mechanism,

[6] A phishing attack is a form of social engineering where the attacker passes itself for a system process, by adopting similarly designed interfaces, email addresses that mimic real system ones, etc., to ask the user for their private data such as passwords.

regardless of its robustness, is virtually over-passed because of a mis-identification from Bob's side.

The problem of fake accounts (also known as sybil accounts) and identity related attacks in OSNs has attracted considerable interest from the research body. There are several proposals that aim at designing strategies to detect fake accounts to be able to remove them from the system. For instance, H. Yu, et al. suggest SybilyGuard [147] and SybilLimit [146], two separate poposals that leverage on the fast mixing principle, by which honest nodes should converge to having high connectivity to the rest of the network [51], to detect Sybil attacks. L. Jin et al. suggest in [72] a framework for the detection of cloning attacks based on attribute and friends' network similarities. A clone attack is achieved when a fake account is created by means of cloning the attribute values of an honest profile. B. Zhe he et al. address identity theft attack across multiple social networks and suggest a solution to protect users from them [63]. All these work, and others following the same approach, aim at detecting malicious nodes that follow identified and formalized attack trends.

Without denying the importance of formalizing attacks and suggesting solutions for their detection, there is also a need for empowering users to help identify the validity of the online accounts they interact with. Along this spirit, M. Sirivianos et al. suggest a framework to elicit the opinion of a user's friends on a social network to confirm her identity on another one, by exploiting social voting to validate identity claims [126]. X. Cai et al. focus on people to people recommendations for friendships' acceptance, by relying on collaborative filtering techniques [27].

In other work, authors leverage on machine processing to verify users identities. For instance, P. Chairunnanda et al. identify users from typing patterns [32], whereas Roffo et al. base their identification on chatting patterns [112]. More recently, O. Goga et al. explore identifying users across networks based on geo-location and time-stamp information attached to their posts and on their writing styles framed using language models [57]. Almost all these pieces of work do not provide to users a means to judge the validity of the identities of profiles they interact with, and when they do, they base the judgment on previous transactions between users. Relying on previous transactions to validate identities makes the identity validation vulnerable to both the availability of such a transaction and also to the possibility of collusion as users know who they are evaluating.

Another available perspective in validating identities on the social web is related to physical and hard verification of users. For example, Westernunion online service[7] validates the identities of accounts' holders by the analysis of a scanned copy they are asked to upload for one of their government issues id cards. The travelers socializing service provided by Couchsurfing website[8] performs the validation of its accounts through means of verification codes sent on postcards to their provided living address. The online community hosting

[7] www.westernunion.com
[8] www.counchsurfing.com

service provided by airbnb[9] validates identities by confirming their mobile phone numbers or by a hard verification of a scanned copy of their id cards. When these validation methods are not to be questioned for effectiveness, their efficiency within a general purpose online social network, especially given the number of users they have, remains subject to doubts.

Realizing the importance of empowering OSN users with tools that would assist them in reliably identifying the contacts they interact with in the network, we have worked on the study and design of solutions that could close this gap. We claim that this is of higher importance within DOSNs more specifically, as possible alternative solutions, such as the proposals reviewed above, would not be applicable in the absence of a central authority in the system. For instance, under a centralized architecture, the service provider may impose policies for validating joiners identities, respond to complaints reporting fake accounts, etc. Such actions may not be possible under a decentralized architecture where every node is the master of itself with no central management. We present in **Chapter 4** the results of our work on identity validation in OSNs with the motivation of assisting users in the task of identifying other accounts in the network, in a users collaborative manner that does not require the usage of any centralized controls.

[9] www.airbnb.com

Chapter 4

Identification Services for OSNs

We have learned from Chapter 1 that identification is a separate concern, indeed the first, in the decomposition stack of access control, and in Chapter 2, we have discussed how access control is one of the essential services in the make up of privacy provision. In Chapter 3 we have re-concluded that access control inherently assumes an underlying identification mechanism by which it confidently considers the authenticity of *subjects* in the system. Therefore, an unreliable identification mechanism may nullify the effectiveness of any access control authorization service regardless of its correctness and robustness. Unauthorized subjects may simply masquerade behind identities of authorized ones and gain technically legitimate access to functionally unauthorized objects. This is illustrated at its best in the scenario of OSNs.

Identities in OSNs are very loose. An OSN user's identity is typically authenticated by proved ownership of a valid email address only. Therefore, users may freely claim any identity they want by simply providing corresponding descriptive values to the attributes of the profiles that represent their membership in the OSN. This has facilitated and led to a large spectrum of security and privacy attacks through identity exploitation, of which identity theft, sybil accounts, and clone accounts are the most famous in the reality of current OSNs (refer to Chapter 3, Section 3.2.3 for more details on that).

As discussed in Chapter 3, access control in OSNs is mainly modeled following a relationship-based approach. As such, authorizations are formulated and evaluated based on the relationships between requesting subjects and requested objects' owners. This suggests that the above mentioned identity exploiting attacks may not have practical effect from an access control perspective unless they establish relationships with the owners of protected objects. The task of identification is thus shifted from being a responsibility of the system to being in the hands of users.

Users may implicitly activate unintended access authorizations to their objects simply by accepting misidentified accounts in their circle of friendship in the network. From an access control technical correctness view, these authorization would be legitimate; however,

by users intended policies they would be putting their privacy at uncomprehended risks. Therefore, this may not call for a revision of the access control authorization model, but for a user engaging service that would empower them to better undertake the identification task of the contacts they establish on the OSN. This becomes of higher importance under the decentralized model of OSNs (i.e., DOSNs).

In fact, although this identified identity validation problem is applicable for OSNs in general, both under centralized and decentralized architectures, there may be more potential for addressing it under the centralized model than under the decentralized one. With the availability of a central authority, ideas such as the enforcement of an identity validation mechanism prior or upon joining may be considered. Moreover, there is the possibility of reporting fake accounts that the central authority could verify and respond to, basically by shutting down the identified fake accounts. This practice is actually available in Facebook OSN, that continuously reports huge numbers of fake accounts that have been identified and their membership was ended.[1] Such possibilities are cut out in the absence of a controlling central authority, calling for other alternatives that would rely on collaboration between peers under decentralized settings.

In this chapter, we devote our effort to the suggestion of a system that would assist OSN users in performing the identification task of their potential relationships in a collaborative systematic manner. The overall objective is to achieve an OSN environment where users are systematically and knowledgeably engaged to minimize the chances of attackers to gain access to protected objects in the system; thus, maximize the chances of segregating fake accounts as outliers or as disconnected groups of which the existence may not have effective impact on the privacy of the OSN users.

Towards achieving this, our goal was to exploit the minimum resources possible, both from the system and from the users, to not subvert the privacy of the users, and to solely rely on collaboration between peers without the employment of any central controls.

We first started by studying the potential of exploiting profile information only with crowd-sourced feedback to estimate the validity of a target profile. The result of this first study was a two phase base model that formally learns hidden and meaningful correlations between profile attributes, and that uses them to assist users in the estimation of the validity of target profiles. We present this community based identity validation (CbIV) base model under **Section 4.1**.

In a second step, we improved the CbIV base model to incorporate anonymity and other quality of service (QoS) guarantees, as we detail and discuss in **Section 4.2**.

Thus far, the aim in the previous two steps has been to study and to prove the value of the model in achieving our set objective of providing an assisting identity validation service in a collaborative systematic way. However, under these two steps, the existence of a central

[1] https://zephoria.com/top-15-valuable-facebook-statistics/

data storage point, from which all profiles could be retrieved, was assumed. After proving the effectiveness of the adopted approach, we moved into studying the achievement of similar results under a completely decentralized architecture wherein no central storage point is required. From this effort, a decentralized identity validation (DIVa) model for DOSNs emanated. Work on DIVa has been conducted in collaboration with colleagues from the Royal Institute of Technology (KTH) in Stockholm. We describe this under **Section 4.3**.

We have also designed a game application that employs the techniques suggested throughout this chapter to challenge players to create fake accounts that would beat the system and integrate within honest users communities. We provide the details of this game under **Appendix B**.

Chapter's List of Abbreviations

CbIV	Community based Identity Validation
CAG	Correlated Attribute Group
CR	Coherence Relation
CA	Candidate Attribute
ITL	Identity Trustworthiness Level
COIP	Continuous, Operable, Impartial, and Privacy Preserving
ES	Evaluation Stream
DIVa	Decentralized Identity Validation
LCAG	Local Correlated Attribute Group
LPC	Local Profile Collection

4.1 Community-based Identity Validation-CbIV Model

The intention is to design a solution to assist OSN users in estimating the reliability and trustworthiness of the accounts that they may connect with, with the goal of using as few resources as possible. The explored idea is to estimate identity trustworthiness on the basis of the coherence within attributes' values on a target profile. This is mainly motivated from sociological works that offer theories and analyses of the identity formation process and its related conflictual issues. Most of these works tend to agree that the final formed identity converges to satisfy both an inner consistency and a sense of coherence based on homogeneity with regard to some "socially accepted assimilative models" [115]. This is also present in Erik Erikson's theory of identity formation, where he concludes, as mentioned in [119], that "the final identity, [...] includes all significant identifications, but it also alters them in order to make a unique and reasonably coherent whole of them." As such, it is expected that profiles reflecting real identities contain and maintain coherent pieces of information. For example, a profile advertising a person as a University professor, a Ph.D. holder, and earning a salary of about 3K euros per month is coherent as per these three values from some society's perspective.

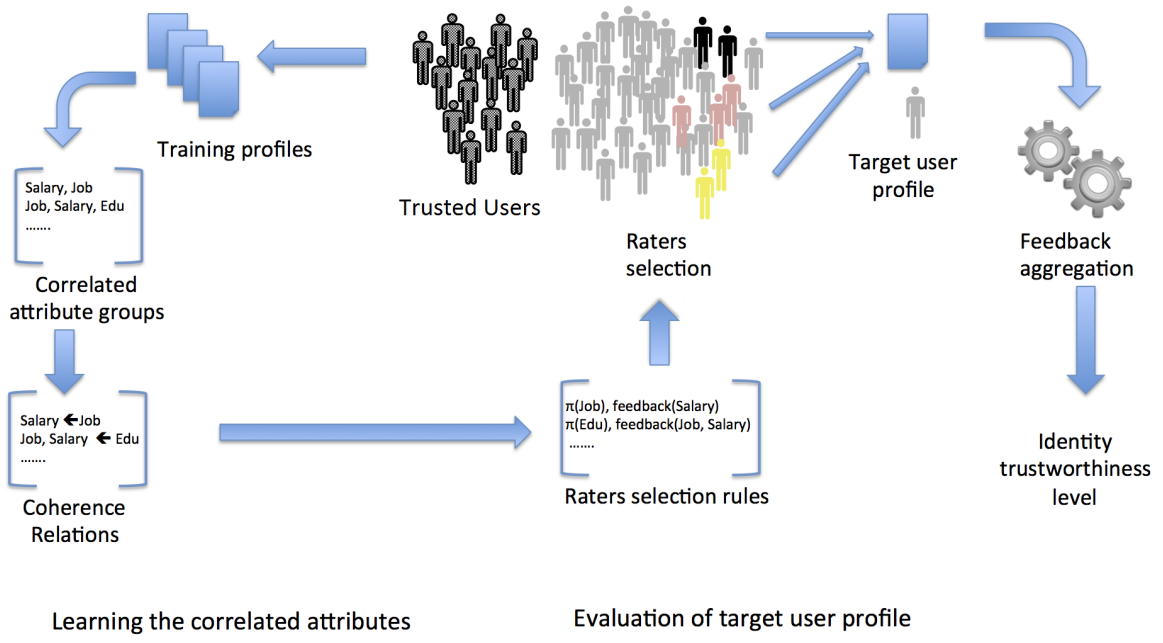


Figure 4.1 Community-based identity validation

Having said that, a first step of our proposed process concerns learning the groups of profile attributes for which it is expected to have correlated values. Once these groups are identified, the system evaluates the profile of a target user by asking the community to evaluate the homogeneity between values in the user's profile for the identified correlated

attributes. The gathered feedback are then aggregated to estimate an identity trustworthiness level for the target user. For example, if the system learns that job, education, and salary are correlated, then part of the identity trustworthiness level for the target user is estimated based on community feedback on his/her job, education and salary values altogether.

Therefore, the proposed approach implies using the community over two phases (see Figure 4.1): (1) supervised learning about correlated attribute groups (CAGs) over a set of training profiles, and (2) evaluating values for these attributes on a target user profile.

The first phase is run as bootstrapping of the system over a predefined training profiles' set and a selected group of participants.[2]

Learning correlated attribute groups (CAGs). In general, not all attribute values are expected to be coherent. In fact, profiles on OSNs mostly represent multiple aspects of identities. A profile, for example, could be advertising both the professional and the personal characteristics of a person which are not necessarily related to each other. A person with a specific job, for instance, can be a parent, single, or in a relationship whatsoever with equal probabilities as job is not determinant of personal relationships. Moreover, some attributes of a profile are obviously not related to each others (e.g., gender and race, or age and gender). Consequently, capturing the coherence of a profile requires first finding those groups of profile attributes whose values are expected to be correlated. We refer to these groups as *correlated attribute groups - CAGs*. To find out these groups, we exploit community-based supervised learning. As it will be explained later, the learning is performed using a group of OSN community participants, referred to as *trusted users*, who are well informed so as to maximize giving reliable feedback for a reliable learning. We assume that the selection of such users is performed by some mechanism, which we do not address in this paper.

Evaluation of a target user profile. In the second phase, the profile of a target user is evaluated by the larger OSN community (which we refer to as *raters*). In general, we expect more meaningful evaluation from *raters* who share the same ground as per some of the values they are asked to provide their opinion on. We propose then to select raters based on the values of correlated attributes of the target profile that has to be judged. For example, assuming that job and salary are correlated, it would be meaningless to ask a student to rate the coherence of the values set *(Job=Plumber, Salary=3K$/month)*, as a student is most probably, if not surely, not knowledgeable about the salary ranges of plumbers. Instead, it would be possible for a rater sharing the same job or working in the same domain to give an informed opinion about that. The opposite, however, is not quite straightforward to claim as people sharing the same range for salary value might not be expected to give an informed opinion on the job corresponding to it. This suggests that there should be a relation between the elements of CAGs specifying which one(s) are determinant to the others.This relation should drive the raters selection for profile evaluation. Therefore, our system needs not only

[2]To answer the highly changing dynamics of OSNs with profiles being added, deleted, or modified, this phase is periodically executed by extending the training set over profiles of new OSN joiners.

to be aware of the CAGs, but it also has to learn the direction of relation between them. We call these relations *coherence relations* and we detail their learning below. Based on these relations, the system selects raters to be involved in the evaluation of correlated attributes over a given target profile. Once all pieces of evaluation are gathered from the selected raters, the system aggregates their values and obtains the estimated *identity trustworthiness level* for the target user.

In what follows, we formalize and describe in detail each of the two phases.

4.1.1 Learning of CAGs and of Coherence Relations

Our problem has some similarities with association mining, where detecting relations among ordered items is done by means of counting the number of occurrences of a group of items across all available historical transactions to infer the strength of the relation between them [20]. Similarly in our scenario, we are interested in finding the groups of correlated attributes and the dependency relations between them; however, the occurrence measure cannot be automatically applied within our environment. This is mainly due to the possible wide range of values each attribute can take, as attributes on OSN profiles are in general non-categorical (i.e., users can insert free text in their profile attributes). If we simply count the frequency of occurrence of values, the measure will be very sparse and not informative. Moreover, we will be losing semantics especially in an OSN environment where semiotic patterns are diverse and mostly informal.[3] To overcome this, we measure the strength of relations among attributes by relying on trusted users, who are asked to express a judgment on the coherence they see among their values. Our goal is to learn relations among attributes in a profile rather than relations among their values; we need to learn the strength of the correlation between 'Salary Range' and 'Job' attributes not between their values. This is what we detail in what follows.

Correlated Attribute Groups

Towards learning correlated attribute groups, we consider all possible combinations over the profile schema as candidate ones. Trusted users are asked to provide feedback on coherence among values of these candidate groups over the profiles of a training set. This process implies the exposure of profile information, and hence particular attention to users' privacy is required. As a matter of fact, our system shall consider managing the identities of profiles without making the trusted users and the raters able to re-identify the user they are evaluating. This is ensured, in a first instance, by discarding the quasi-identifier attributes from all the reasoning of the system.[4]

The formal definition of candidate attribute groups is given below.

[3] Many forms of informal languages are being adopted in digital socializing environments, such as chat-language, abbreviations, etc.

[4] We consider quasi-identifiers those attributes for which the values can be used, alone or together with some other external or internal information, to approximate or to determine the identify of their owner [34].

Table 4.1 Feedback types and corresponding values for the learning phase

'Definitely yes', 'Definitely no'	1
'Most probably', 'Less likely'	0.5
'Not meaningful to judge', 'I do not know'	0

Definition 4.1.1 *Candidate Attribute Group. Let* $\boldsymbol{S}$ *be the profile schema adopted in the OSN. Let* $\boldsymbol{QI} \subset \boldsymbol{S}$ *be the set of quasi-identifier attributes in* $\boldsymbol{S}$*. The set of candidate attribute groups of order* m *over* $\boldsymbol{S}$ *(*$|\boldsymbol{S}| > m > 1$*), denoted as* CA_m*, contains all possible combinations of size* m *of attributes in* $\boldsymbol{S} \backslash \boldsymbol{QI}$*. We denote as* $\boldsymbol{CA}$ *the set of the candidate attribute groups for any size* m *over* $\boldsymbol{S}$*,* $|\boldsymbol{S}| > m > 1$*.*

Trusted users' feedback is simply a discrete answer to a question on selected values of attributes, which belong to a candidate attribute group, from the profiles in the training set. To make it clearer, let us consider Joanna to be a member of the social network whose profile is in the training set and which contains values *Black African* for race and *Kenya* for country of origin. The question on the profile values of Joanna over {'Race', 'Country of Origin'} is: *'Do you think a real person (not faking an identity) can hold as race* Black African *and as country of origin* Kenya *altogether?'*. Six answers are possible with each of them being related to a discrete value, as specified in Table 4.1. YES and NO answers result, in this phase, in equal feedback scores because both answers imply that users capture in the group of attributes something that makes them able to make a clear judgment.

To evaluate the correlation within a candidate group, we introduce the following definitions. Hereafter, given a user u, we denote the values of attributes in $\mathbf{Y} \subseteq \mathbf{S}$ for user u as a vector $\mathbf{Q}_u^Y$.

Definition 4.1.2 *Feedback on Candidate Attribute Groups. Let* $\mathcal{TU}$ *be the set of available trusted users in the OSN, let* u *be a user in the OSN, with* $u \notin \mathcal{TU}$*. Let* $\boldsymbol{Y} \in \boldsymbol{CA}$ *be a candidate attribute group, and* $\boldsymbol{Q}_u^Y$ *be the values for attributes* $\boldsymbol{Y}$ *in* u*'s profile. The feedback of* $\mathcal{TU}$ *on* u *w.r.t.* $\boldsymbol{Y}$*, denoted as* $f_{\mathcal{TU}}(\boldsymbol{Y}_u) \in [0, 1]$*, is computed as:*

$$f_{\mathcal{TU}}(\boldsymbol{Y}_u) = \frac{1}{|\mathcal{TU}|} * \sum_{j \in \mathcal{TU}} f_j(\boldsymbol{Q}_u^Y)$$

where $f_j(\boldsymbol{Q}_u^Y)$ *is the feedback expressed by the trusted user* j *on* $\boldsymbol{Q}_u^Y$*.*

The received feedback per candidate group is aggregated to compute its support:

We assume some attributes out of the profile schema are identified as quasi-identifiers and we do not cover in this paper the process of such identification.

Definition 4.1.3 *Support. Let $\mathcal{TU}$ be the set of trusted users in the OSN, let $\mathcal{T}$ be a set of users in the OSN whose profiles are in the training set, with $\mathcal{T} \cap \mathcal{TU} = \emptyset$. Let $\boldsymbol{Y} \in \boldsymbol{CA}$ be a candidate attribute group, the support for $\boldsymbol{Y}$ is computed as:*

$$supp(\boldsymbol{Y}) = \frac{1}{|\mathcal{T}|} * \sum_{i \in \mathcal{T}} f_{\mathcal{TU}}(\boldsymbol{Y}_i)$$

Once all the supports are computed, the groups having a support high enough to justify a correlation between their elements are considered *correlated attribute groups*.

Definition 4.1.4 *Correlated Attribute Group. Let* **CA** *be the candidate attribute groups defined over a profile scheme $\boldsymbol{S}$. Let $sh \in [0,1]$ be a threshold value. The Correlated Attribute Groups are defined as $\boldsymbol{CAG} = \{\boldsymbol{Y} \in \boldsymbol{CA} | supp(\boldsymbol{Y}) \geq sh\}$.*

The value of the threshold *sh* is determined dynamically based on all the computed supports and their distribution. In a preliminary step, and for simplicity, we set it to the mean of all computed supports.

Example 4.1.1 *Consider the candidate group, {Job, Education}. Suppose we have 3 profiles in the training set and 2 trusted users. This results in 3 feedback questions each corresponding to the values of the candidate group's attributes in a training profile, and in 6 feedback rates (1 per question per trusted user). The support of this group will be the average of these 6 feedback rates. Suppose this support is equal to 0.8. Assuming sh = 0.5, then this candidate group is a correlated attribute one.*

Coherence Relations

Always in analogy with association mining, the detection of coherence relations within correlated attribute groups is done by computing their corresponding confidence. In association mining, given two items R and L, the confidence $Conf(R \Longrightarrow L)$ is the ratio between the support of the two items (i.e., the occurrences where both items appear) and the support of the single item R, that is, $Conf(R \Longrightarrow L) = \frac{supp(R \cup L)}{supp(R)}$. If the resulting value is greater than a threshold, then $R \Longrightarrow L$ can be considered a meaningful rule [20]. As the support of a correlated attribute group is given, in our case, by the average of trusted users' feedback, this recalled confidence definition cannot be directly applied. In designing a new confidence definition, we assumed that the best way to decide if $R \Longrightarrow L$ is a coherence relation is to consider feedback of trusted users that are informed on R (i.e., have values for attributes in R similar to those of training profiles that they are judging) against the feedback of all other trusted users. Therefore, we introduce the following definition:

Definition 4.1.5 *Conditional Support. Let $\mathcal{TU}$ be the set of trusted users in the OSN, let $\mathcal{T}$ be a set of users in the OSN whose profiles are in the training set, with $\mathcal{T} \cap \mathcal{TU} = \emptyset$.*

Let $\boldsymbol{Y} \in \boldsymbol{CAG}$ *be a correlated attribute group. The conditional support for* $\boldsymbol{Y}$*, given* $\boldsymbol{B} \in \boldsymbol{Y}$*, denoted as* $supp(\boldsymbol{Y}|\boldsymbol{B})$*, is computed as:*
$supp(\boldsymbol{Y}|\boldsymbol{B}) = \frac{1}{|\mathcal{T}|} * \sum_{i \in \mathcal{T}} f_X(\boldsymbol{Y}_i)$*, where* $X \subseteq \mathcal{TU}$ *such that* $X = \{x \in \mathcal{TU} | \forall b \in \boldsymbol{B}, \boldsymbol{Q}_x^b \in [\boldsymbol{Q}_i^b - \epsilon, \boldsymbol{Q}_i^b + \epsilon]$*, if* b *is a numerical attribute;* $\boldsymbol{Q}_x^b = \boldsymbol{Q}_i^b$*, otherwise}, where* ϵ *is a small tolerance value.*

Based on the above definition, we define the following:

Definition 4.1.6 *Confidence. Let* $\mathcal{TU}$ *be the set of trusted users in the OSN, let* $\mathcal{T}$ *be a set of users in the OSN whose profiles are in the training set, with* $\mathcal{T} \cap \mathcal{TU} = \emptyset$*. Let* $\boldsymbol{Y} \in \mathbf{CAG}$ *be a correlated attributes group, and let* $\boldsymbol{L} \subset \boldsymbol{Y}$*. We compute the confidence of* $\boldsymbol{L}$ *over* $\boldsymbol{Y}$ *as:*

$$Conf(\boldsymbol{L}, \boldsymbol{Y}) = \frac{supp(\boldsymbol{Y}|\boldsymbol{L})}{supp(\boldsymbol{Y})}.$$

Definition 4.1.7 *Coherence Relation. Let* $\boldsymbol{Y} \in \mathbf{CAG}$ *be a correlated attributes group, and let* $\boldsymbol{L} \subset \boldsymbol{Y}$ *be a set of attributes such that* $\boldsymbol{R}$*=* $\boldsymbol{Y}$ *-* $\boldsymbol{L}$*. We define* $p = (\boldsymbol{L} \Longrightarrow \boldsymbol{R})$ *as a coherence relation over* $\boldsymbol{Y}$*, if* $Conf(\boldsymbol{L}, \boldsymbol{Y}) > ch$*, where* ch *is a given threshold.*

Example 4.1.2 *Considering the CAG from Example 4.1.1, and supposing that Conf(Job, {Job, Education}) = 1.1, Conf(Education, {Job, Education}) = 1.8, and ch = 1.2, then p =(****Education*** $\Longrightarrow$ ***Job****) is a coherence relation over {Job, Education}.*

4.1.2 Estimation of Identity Trustworthiness Level

The second phase concerns evaluating the profile of a target user, by a group of selected raters, to estimate its Identity Trustworthiness Level (*ITL*). Raters selection is driven by the assumption that more meaningful feedback is expected from raters sharing the same ground/values as per the attributes they are asked about. For this purpose, we exploit coherence relations to define a set of conditions for raters selection:

Definition 4.1.8 *Raters Selection. Let* $\boldsymbol{Y} \in \mathbf{CAG}$ *be a correlated attribute group, and let* $\mathcal{P}$ *be the set of coherence relations defined over* $\boldsymbol{Y}$*. Let* $\mathcal{R}$ *be a set of available raters in the OSN, and let* u *be a user in the OSN whose profile is going under evaluation, with* $u \notin \mathcal{R}$*. The raters selected for* $\boldsymbol{Y}$ *and* u*, based on the coherence relations in* $\mathcal{P}$*, are computed as follows:*

$$RS_u^Y = \{r \in \mathcal{R} | \forall p \in \mathcal{P}, \forall s \in p.\boldsymbol{L}, Q_r^s \in [Q_u^s - \epsilon, Q_u^s + \epsilon], \text{ if } s \text{ is a numerical attribute;}$$
$$Q_r^s = Q_u^s, \text{ otherwise}\}.$$

where ϵ *is a small tolerance value, and* $p.\boldsymbol{L}$ *denotes the subset* $L \in Y$ *in the coherence relation* p*.*

Table 4.2 Feedback types and corresponding values for evaluation phase

'Definitely yes'	1	'Most probably'	0.5
'Definitely no'	-1	'Less likely'	-0.5
'I do not know'		0	

Example 4.1.3 *Let us recall from Example 2, p = (**Education** $\Longrightarrow$ **Job**). Assume our user Joanna has on her profile: Education = 'Fine arts graduate' and Job = 'Fitness trainer'. Rater selection implies choosing raters who are 'Fine arts graduate' to evaluate Joanna on this group.*

ITL is defined as the aggregation of feedback received by selected raters. This feedback is collected as answers to feedback questions as provided above. However, given that the aim here is to rate the profiles and not to learn relationships between attributes, the values corresponding to the questions are different in that negative and positive answers are judgmental at this level (see Table 4.2).

Definition 4.1.9 *Evaluation of a user profile w.r.t. a correlated attribute group. Let u be the user in the OSN to be evaluated. Let $\boldsymbol{Y} \in \boldsymbol{CAG}$ be a correlated attribute group. Let RS_u^Y be the corresponding selected raters (see Definition 4.1). Let $\boldsymbol{Q}_u^Y$ be the values of $\boldsymbol{Y}$ on the profile of u, and let $f_j(\boldsymbol{Q}_u^{\boldsymbol{Y}})$ be the feedback of rater j on $\boldsymbol{Q}_u^Y$. The evaluation of u w.r.t. $\boldsymbol{Y}$ is defined as:*

$$E_Y(\boldsymbol{Q}_u^Y) = \frac{1}{|RS_Y|} * \sum_{æ \in RS_u^Y} f_j(\boldsymbol{Q}_u^{\boldsymbol{Y}})$$

Based on the above definition, the *Identity Trustworthiness Level* for a target user u is computed as follows:

Definition 4.1.10 *Identity Trustworthiness Level. Let $u \in \mathcal{U}$ be a user in the OSN. Let $\mathcal{Y} \subseteq \boldsymbol{CAG}$ be the set of correlated attribute groups such that $\boldsymbol{Q}_u^y \neq \emptyset \ \forall y \in \mathcal{Y}$. The identity Trustworthiness Level for u is:*

$$\boldsymbol{ITL}_u = \frac{1}{|\boldsymbol{CAG}|} * \sum_{Y \in \mathcal{Y}} E_Y(\boldsymbol{Q}_u^Y);$$

Example 4.1.4 *For simplicity, assume we only have the correlated attribute group and the coherence relation identified in Examples 1 and 2 respectively. Consider user Joanna as provided in Examples 3 and assume the system selects two raters who provided feedback corresponding to the values: {0, 0.5}. ITL_{joanna} will consequently be equal to 0.25.*

4.1.3 Experiments and Results on CbIV

We test the utility of the CbIV model, hereafter referred to as CB, under two different experimental environments. On the first hand, we test the effectiveness of CB for learning and detecting correlated attribute groups, by comparing it to a competing alternative on a census dataset. On the other hand, we test the effectiveness and the efficiency of our CB method in rating users' profiles within a real OSN environment.

Effectiveness of CB Learning

An obvious alternative to our method in learning correlated attribute groups is opting for machine based association mining techniques. For this reason, we have compared CB learning with association mining learning. The goal of this experiment is to find whether CB learning can capture correlated attributes which cannot be detected by simple machine learning from the data (hereafter, we refer to the machine based learning by MB). In running this experiment, we took in consideration that the performance of MB learning is optimized on categorical and sanitized datasets. For this purpose, we run the experiment on a census dataset.

The dataset:

The dataset is from the US Census Bureau, made available under the name of Adults dataset.[5] It contains 45,222 census descriptive and anonymized records capturing 14 attributes. The dataset comes distributed into 2/3 of its records as training data and 1/3 as validation data. We have considered the 2/3 training records to make our training dataset. This dataset fits most of the requirements for the objective of this experiment. First, it is representative of a user profile and it is rich enough in terms of the attributes it covers. Second, it contains a large number of records. Finally, its values are categorical and well sanitized which makes it favorable for running an association mining algorithm. However, for few attributes, the dataset goes into very fine grained levels of detail as per their values, thing which is not expected on an OSN profile. For example, the education level attribute can take one of sixteen values each referring to the exact schooling year. In order to make this more representative of an OSN user profile, we have over grouped some of these values into one to result in ten possible values only out of the original sixteen. The aim is to make the values in the dataset better understood by the participants in the experiment. In addition to this, 3 attributes have been discarded. These are capital loss, capital gain, and gained-salary.[6] Table 4.3 lists all the considered attributes.

[5] http://archive.ics.uci.edu/ml/machine-learning-databases/adult

[6] Capital loss and capital gain are not expected as part of a social profile that one shares on an OSN. The gained-salary attribute is represented in the dataset as a binary salary-class field only.

Table 4.3 Adults dataset adopted profile schema

Attribute	**Description**
Age	Age
Work-class	Work Class
Education	Education Level
Educ-num	Number of years spent at school
Marital-status	Marital Status
Occupation	Job
Social-role	Social Role
Race	Race
Sex	Gender
hrsperweek	Number of hours worked per week
Country	Country of origin

Settings and Design:

We run CB learning and MB learning on the training dataset. More specifically, we evaluated the correlation between all possible and non-trivial candidate groups, i.e., **CA**, of size 2 over attributes in Table 4.3.[7] For each of these methods, we set the support threshold, sh, for correlated attribute groups' identification (see Definition 3.4) to the average of all the computed supports.

MB learning. Given that our training dataset is categorical and all its values are sanitized, we have opted for a version of the Apriori algorithm [3] to make our MB learning. This algorithm computes the support of a given CA by counting the number of mutual occurrences of its couple-values. We have slightly altered the algorithm to compute the support for only the 34 CAs we have.

CB learning. We made available to the public an online survey in both English and Italian languages. A survey question has the same format as the feedback question stated above. 38 participants, mainly from Morocco and Italy with little representation of other nationalities such as Iran, Lebanon, Turkey and Tunisia, took the survey. The majority (85%) of our participants are University students or fresh graduates who have recently joined the job market as engineers or salespersons (age is in the range [20, 30]). The remaining 15% of our participants are older professionals within the age range of 35 to 55 and with professions such as, medical doctor, government officer, and University professor. To have a reliable feedback, all our participants have been presented with a thorough explanation about the experiment and its aim, and most of them have accepted to participate by providing careful

[7]For proof of concept and for simplicity in results' presentation and discussion, we limit the experiment to CAs of size 2 only. Trivial combinations are ones such as, {country, education}, {race, education}, {country, gender}, etc.

attention and focus to the survey questions. Each one of the participants answered at least 55 feedback questions corresponding to our 34 candidate groups (CAs). The support of each of the 34 candidate groups has been computed as suggested by the equations in Definitions 3.2 and 3.3.

Achieved Results:

Utility in learning correlations. Table 4.4 shows a summary of the MB and CB evaluations of our 34 candidate groups. The table shows only candidate groups evaluated as correlated attributes either by MB or CB. The value 'insig' in the supports column in the table means that the candidate group received an insignificant support using the method to which the sub-column refers (i.e., the support by that method was smaller than the threshold *sh*).

As a first observation, the first 19 candidate groups passed the support test using MB, but failed to prove meaningful correlation using CB. Hereafter, we refer to these groups as CAG_{MB}. Second, we see that only three candidate groups, i.e., the second group of rows in the table, are evaluated as correlated by both MB and CB. This disparity in results between MB and CB can be mainly explained by two elements each specific to one of these methods. On the one hand, MB method captures repeated/common trends in the dataset which are inherent to the statistical and distribution trends of the censused population (i.e., the Adults dataset). This means that the detected CAG_{MB} would be expected to change if the dataset changes. On the other hand, CB method captures logical and social connections or definitions of the participants with regard to what they perceive as a socially conforming combination. This point brings us to stress on the fact that our method is community dependent and justifies its reliance on human feedback instead of relying on statistical or association mining techniques. This can be better understood when examining some of the combinations in CAG_{MB}, such as the group {Gender, Education}. The proved correlation in this combination by MB reflects nothing but that people censused in the Adults dataset tend to have one gender dominance within some educational categories of the data. This group would not be expected to pass the support test using CB because there shall be no relationship between gender and achieved education, unless some strong stereotypes undermine.

In contrast to that, the trends of CB are not related to population but to people's common judgment on what makes sense for them as valid attribute values combinations. Results in Table 4.4 confirm our theory/assumptions and prove the existence of correlations between some attributes which cannot be captured by pure learning from or statistical analysis of the data. Indeed, CB method has identified eight of such correlations that MB considered insignificant, i.e., the last group of rows in Table 4.4, to which we refer hereafter as CAG_{CB}.

Value of learned correlations. To better understand and validate the value of our CB learning, we compare the usefulness of CAG_{CB} to the one of CAG_{MB} in the estimation

Table 4.4 Candidate groups considered as correlated attributes either by MB or by CB

	Supports	
Candidate Group	**MB**	**CB**
educ-num, gender	0.36	insig
hrsperweek, gender	0.66	insig
educ-num, race	0.34	insig
hrsperweek, race	0.36	insig
gender, race	0.44	insig
educ-num, social-role	0.29	insig
hrsperweek, social-role	0.30	insig
gender, social-role	0.38	insig
educ-num, marital-status	0.27	insig
hrsperweek, marital-status	0.26	insig
gender, marital-status	0.36	insig
gender, education	0.25	insig
educ-num, work-class	0.29	insig
hrsperweek, work-class	0.30	insig
gender, work-class	0.37	insig
race, work-class	0.21	insig
educ-num, age	0.28	insig
race, age	0.21	insig
gender, age	0.37	insig
hrsperweek, age	0.35	0.56
social-role, marital-status	0.21	0.56
educ-num, education	0.37	0.52
education, hrsperweek	insig	0.66
age, marital-status	insig	0.58
education, occupation	insig	0.59
occupation, hrsperweek	insig	0.67
occupation, educ-num	insig	0.63
occupation, work-class	insig	0.63
country, race	insig	0.56
work-class, educ-num	insig	0.57

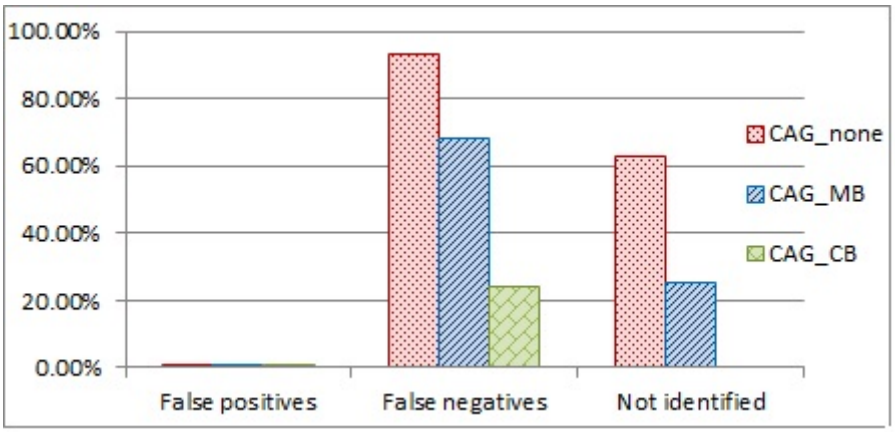


Figure 4.2 False positives, false negatives, and not identified profiles under CAG_{CB}, CAG_{MB}, and CAG_{none}

of records' trustworthiness. For this purpose, we have considered the remaining 1/3 of the records in the Adults dataset and we have randomly scrambled 50% of them to simulate fake profiles.[8] The scrambled records were labeled as fake (F), whereas the original ones as real (R). The union of these real and fake records makes our validation dataset.

We made available a new survey through which we asked participants to rate records from the validation dataset based on CAG_{CB}, CAG_{MB}, and CAG_{none} (this refers to the groups which did not pass the support neither by CB or MB) independently. We note that the participants in this second phase of the experiment are different from the ones who identified the CAGs. We compute the resulting ITL of every record under each of the three scenarios. The record is estimated real if ITL is positive. It is estimated fake if ITL is negative. Some profiles could not be identified by the raters as fake or real (i.e., their ITL approximates 0). We refer to these by *NI* - Not-Identified category. Figure 4.2 summarizes the obtained results; it shows the percentages of false positives, false negatives, and NI category achieved under each of the three scenarios.

We can clearly see on Figure 4.2 that the reliability of rating records based on CAG_{CB} is the highest compared to CAG_{MB} and to CAG_{none} for all the cases. For example, participants incorrectly rated only 24 % of fake profiles (false negatives) using CAG_{CB} against 68% using CAG_{MB}. By using CAG_{CB}, we also obtain the lowest value for not identified profiles, which means that raters could give an opinion, positive or negative, on almost all the questions they received on CAG_{CB}. We must mention that using a group of participants (for the rating phase) different from those who performed the learning of the CAGs does in a sense reconfirm the existence of specific correlations between profile attributes that are more meaningful for judging a profile's validity. That is, the CAGs identified in the learning phase are more

[8]Fake profiles are made by setting their attributes with values randomly selected from the available records and by ensuring that a fake profile does not have two values taken from the same original record.

useful to raters, who are uninformed about the learning process, than other combinations of attributes (i.e., CAG_{MB} and CAG_{none}) in the task of reliably estimating the trustworthiness of a target profile.

To sum-up, this first experiment proves both utility and value of our method in learning CAGs which cannot be determined by machine based techniques and which are more significant in reliably rating profiles. Our method successfully passes the test against machine learning under best conditions for this latter; i.e. a categorical sanitized dataset. Such conditions are not expected under a real OSN environment in which users insert free text using different semantics and extensively varied typing patterns.

Performance within Real Environment

In this second experiment, we study the feasibility and effectiveness of our CB method within an OSN environment. We choose Al Akhawayn University (AUI)[9] as our study group. In addition, we have opted for Facebook as one of the major, most popular, and widely used social networks. The choice of one community in this experiment is mainly aimed to maximize chances for knowledgeable feedback.

The Dataset:

With their consensus, we collected the Facebook profile data of 70 alumni students from the cohorts of 2011 and 2012 (see Table 4.5 for the collected profile attributes).[10] Our 70 profiles have 64% females and 36% males with approximated average age of 23.[11] These 70 profiles made our training dataset for the learning part of the model.

Some attributes have one dominant value in our training profiles set, such as the attribute Religious Views for which 99% of our training profiles had an equal value, and hence they have not been considered in the definition of candidate groups.[12] We obtained 14 possible combinations of size 2 over the experiment's profile schema to be our candidate groups (CA).

Settings and Design:

Trusted Users. We had 35 participants for the *Trusted Users* group[13] in our model. These are all current students of AUI attending different sections of the same course. 49% of them

[9] AUI is a Moroccan University operating under the American model for education. It offers all living facilities within its campus and its students live in there as a community: www.aui.ma

[10] The second column specifies whether the attribute accepts multiple values or not.

[11] The average age corresponds to the censused mean age of AUI graduating students in 2011 cohort.

[12] Considering one-value-dominant attribute in our candidate groups seemed meaningless as we cannot learn correlations given lack of diversity in values.

[13] The 35 participants are considered trusted because they come from the same community, they understood the objective of the experiment, and they engaged to take the survey with due care and attention to its questions.

Table 4.5 Adopted Profile Schema - OSN dataset

Attribute	**Multi-Value**
Gender	No
Religious Views	No
Work Place	Yes
Work Location	Yes
Country	No
Education (Ed. Major)	No
Sports	Yes
Pref. Music	Yes
Pref. Movies	Yes
Pref. Books	Yes
Likes (liked pages)	Yes
Groups (joined groups)	Yes

are females and 51% are males with an average age of 19.5. They all took an online survey in which they answered between 34 and 50 feedback questions on our 14 candidate groups.

Feedback Questions. We set feedback questions as specified under Section III.A. For those attributes accepting multi-values, such as Preferred Music, their different values have been considered one at a time when computing the support of groups containing them. For example, if a profile has two values for Preferred Music, *music1* and *music2*, and *female* as value for Gender attribute, then the support of the candidate group {Gender, Preferred Music} was computed considering feedback on [*female, music1*] and [*female, music2*] as two independent combinations.

Evaluation. We have considered the 35 trusted users as the set of target users to be evaluated. In order to make a sound testing dataset, and since these profiles correspond to real identities (i.e. most probably they are all coherent), we created 20 fake profiles out of the 35 real ones. This was done by assigning to each attribute in a fake profile a randomly selected value from a real one, ensuring that one fake profile will never have two attribute values taken from the same real profile. We labeled the testing profiles as RP for the real 35 and as FP for the 20 fake ones. We got 13 raters[14] to evaluate the 55 profiles in our testing dataset.

Achieved Results:

Defined Correlated Attribute Groups. Table 4.6 presents the supports received for all the 14 candidate correlated attribute groups (CAs) considered in the experiment. Considering

[14]These 13 raters come from the group of whom we collected the 70 profiles for the training dataset.

Table 4.6 Achieved support per candidate group

Candidate Group (CA)	Support
Gender, Movies	0.72
Ed. Major, Groups	0.67
Gender, Sports	0.65
Groups, Likes	0.61
Pref. Music, Pref. Movies	0.57
Pref. Movies, Pref. Books	0.44
Ed. Major, Likes	0.43
Sports, Likes	0.36
Pref. Music, Gender	0.31
Ed. Major, Sports	0.31
Pref. Music, Pref. Books	0.28
Gender, Pref. Books	0.23
Ed. Major, Books	0.20
Ed. Major, Movies	0.11

a support threshold of $sh = 0.30$, the 10 first groups in Table 4.6 are the correlated attribute groups (CAG) considered in the experiment.

Defined Coherence Relations. In order to detect the coherence relations in the considered CAGs, the confidence measures were computed. Figure 4.3 shows these values presenting a comparison between the two possible coherence relations out of each one of them. For example, for the CAG = {'Gender', 'Pref. Movies'}, the two possible coherence relations are ['Gender' $\Longrightarrow$ 'Pref. Movies'] and ['Pref. Movies' $\Longrightarrow$ 'Gender'].

A first remark from Figure 4.3 is that in all CAGs the two confidence measures are not equal. This implies that a direction in the coherence relation can be always detected. More

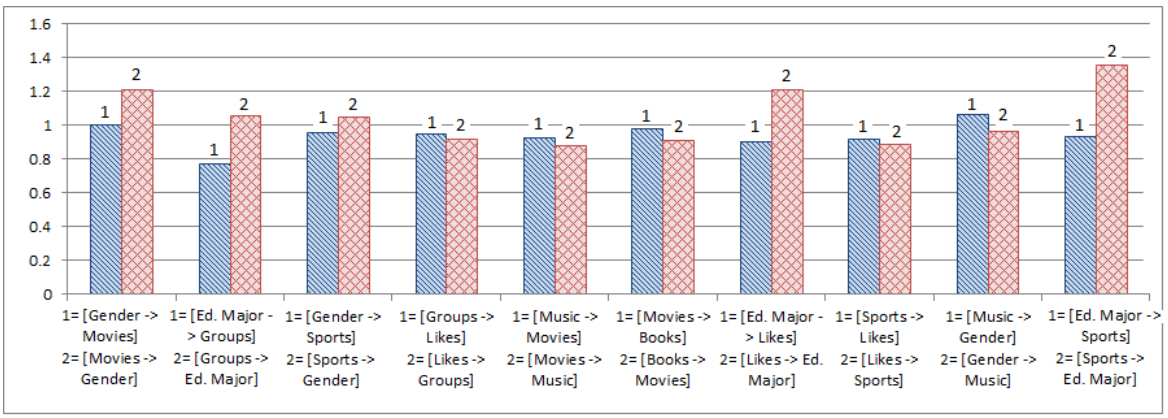


Figure 4.3 Coherence Relations for the 10 defined CAGs

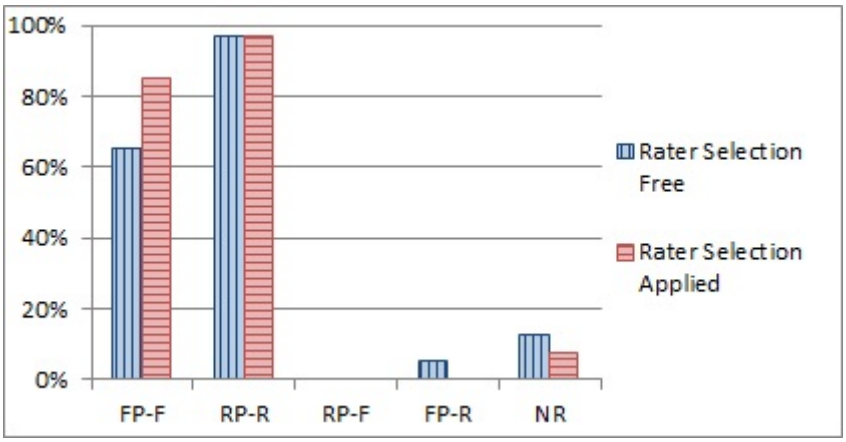


Figure 4.4 Evaluation of profiles with rater selection applied vs. rater selection free

importantly, this difference is more relevant in some CAGs than in others. In fact, on Figure 4.3, we can point to 4 specific CAGs for which the difference between the two coherence relations is considerable (i.e., this difference exceeds 0.2 for the 4 of them). These are {'Ed. Major', 'Groups'}, {'Ed. Major', 'Likes'}, {'Movies', 'Gender'}, and {'Ed. Major', 'Sports'}. What is flashing in these 4 CAGs is that 3 of them contain the attribute 'Ed. Major' and that this latter is in the right side of the dominant coherence relation over the 3 of them. We discuss this further under Section V.C.

Evaluation of Profiles. In order to evaluate the utility of raters selection, we performed profile evaluation with and without it.

Scenario 1 - Raters Selection Free. We dropped raters selection and we processed the rates provided by our 13 raters on each of the 55 testing profiles.

Scenario 2 - Raters Selection Applied. We apply rater selection with regard to the 4 most important coherence relations out of the ones mentioned in Figure 4.3. These are: ['Groups' $\Longrightarrow$ 'Ed. Major'], ['Likes' $\Longrightarrow$ 'Ed. Major'], ['Pref. Movies' $\Longrightarrow$ 'Gender'], and ['Sports' $\Longrightarrow$ 'Ed. Major']. Our 13 raters have been categorized by the determinant attribute (i.e., the left attribute) in each of the 4 considered coherence relations.

Under each of these two scenarios, an estimated ITL is computed for every profile in the testing dataset. Based on the computed ITL, the profile is classified as positively rated (positive ITL), negatively rated (negative ITL), or neutrally rated (ITL approximates 0). Recalling that all testing profiles are preliminary labeled as real (RP) or as fake (FP), we categorize our achieved estimations under 5 groups: 1. the FPs negatively rated (FP-F), 2. the RPs positively rated (RP-R), 3. the RPs negatively rated (RP-F), 4. the FPs positively rated (FP-R), and 5. the RPs and FPs neutrally rated (NR). Figure 4.4 provides the number of profiles (as percentages) under each of these five categories both when raters selection is applied and when it is not.

The first thing we learn from Figure 4.4 is the accuracy of our method in correctly rating profiles. For instance, more than 60% of fake profiles and more than 95% of real one are correctly estimated when raters selection was not applied. In addition to that, we clearly notice how raters selection improves the scores under all the five categories. Most importantly, raters selection considerably increased the number of profiles in the FP-F category and decreased the FP-R one. In the first case, raters selection correctly rated 85% of fake profiles against 65% only in the other scenario. In the second case, raters selection minimized the error in incorrectly estimating fake profiles by 5%. Moreover, raters selection minimized the NR category by over 7%.

4.1.4 Extended Discussions

Our two initial experiments on the CbIV method, each in a different context and from a different perspective, showed how our method draws from and harnesses the wisdom of the community to reliably estimate the trustworthiness of OSN claimed identities. In the first experiment, we learned that there are correlations between some profile attributes which cannot be detected by pure machine learning only, even when the training dataset is optimized for this latter. Relying on community-sourcing detects these correlations which we proved have higher effectiveness in reliably rating profiles. Overall, the results of the first experiment justify our method and approve its reliance on community feedback to achieve its objective. Still confirming the same, the second experiment proved that our method can efficiently and effectively stretch to the specificity of an OSN arena.

In parallel to confirming our method, the experiments provided other prominent lessons opening interesting opportunities for further exploration. For example, the coherence relations detected in the second experiment might be interpreted as specific to the experiment's community. In fact, the observation made earlier with regard to the Ed.Major attribute which exists in almost all the strong coherence relations tells us that correlated attribute groups and their coherence relations are bounded within social communities. This is an inherent result given that social norms and social configurations are commonly known to be community and social-groups specific (each community has its own culture and identity which differs from the other). This point brings us to understand that our method is community dependent and shall consider constructing its learning phase within identified communities and not over all the OSN community as one single entity. This last point is what makes the basis of our study of the method under a DOSN architecture, as we describe in Section 4.3.

Another crucial point to consider are quality guarantees of the system, especially in terms of privacy preservation of profiles values whilst being evaluated, accuracy in evaluations within the dynamical nature of OSNs, and operability guarantees of the model. This point triggered the extension of the model described in Section 4.2.

Table 4.7 Simplified OSN representation

Profile Schema = [Job, Education, Hobbies, City, Age]
$Q^S(u1)$={Student}, {Stanford}, {hiking, biking}, {NJ},{24}
$Q^S(u2)$={Student}, {Stanford}, {tennis}, {NY},{21}
$Q^S(u3)$={Detective}, {P.School}, {soccer, biking}, {TinyCity},{55}
$Q^S(u4)$={Nurse}, { }, {movies, swimming}, {TinyCity},{ }
$Q^S(u5)$={Student}, {H. School}, {basketball}, {TinyCity},{20}
$Q^S(u6)$={Student}, {Stanford}, {baseball, dance}, {Milan},{25}
$Q^S(u7)$={Student}, {P.School}, {hiking, biking}, {TinyCity},{30}

4.2 Continuous, Operable, Impartial, and Privacy aware Evaluation-COIP Model

We recall from Section 4.1, that the suggested CbIV model is designed over two phases, a *learning phase* and an *evaluation phase*. The focus in this section is on the *evaluation phase*. The goal is to incorporate in the design of this phase a set of quality properties that would be required for a practical solution for deployment in a real OSN environment.

We assume that the learning phase, as designed in the CbIV model, is executed and resulted in a superset of defined correlated attribute groups, to which we refer as CAG^*. We also assume that all coherence relations have been extracted. Without loss of generality, for each $CAG_i \in CAG^*$ we denote by CR_i its coherence relation. Moreover, given a user u and the coherence relation CR_i on CAG_i, $RS_i(u)$ refers to the set of potential selected raters to rate u on CAG_i.

During the evaluation phase, the profile of a target user is cut into chunks corresponding to each of the CAGs defined in the system. Each chunk is evaluated separately. For easier reference, we denote the average rate on a target user profile u w.r.t to a CAG $\overline{CAG}$ by, $f(Q^{\overline{CAG}}(u))$.

We consider the profile schema, the users, and the profiles in Table 4.7 as a simplified OSN representation that is referenced throughout this section.

We also provide the following reference examples that would help for illustration purposes throughout the flow of this section.

Example 4.2.1 *Let us assume that the learning process on the profile schema in Table 4.7 yielded the following CAGs and coherence relations. Note that for ease of presentation, in our examples, we limit the size of CAGs to two only. However, CAGs of greater size are possible as well.*

- CAG_1 = *{Job, City} with the coherence relation denoting that City implies Job,* CR_1 = $(\boldsymbol{City \longrightarrow Job})$

- CAG_2 = *{Education, Hobbies}, with* CR_2 = (***Hobbies*** $\longrightarrow$ ***Education***)
- CAG_3 = *{City, Education}, with* CR_3 = (***City*** $\longrightarrow$ ***Education***)
- CAG_4 = *{Job, Age}, with* CR_4 = (***Job*** $\longrightarrow$ ***Age***)
- CAG_5 = *{Education, Age}, with* CR_5 = (***Education*** $\longrightarrow$ ***Age***)

Example 4.2.2 *Let us consider that the profile of user u3 from Table 4.7 is going for evaluation. Assuming the CAGs and the coherence relations in Example 4.2.1, the system will select users u4, u5, and u7 to rate u3 on* CAG_1*, u1 and u7 to rate her on* CAG_2*, and so on for the remaining CAGs. Let us assume that the average rates that u3 received by the selected raters for every CAG are the following:*

- $f(Q^{CAG_1}(u)) = 0.7$
- $f(Q^{CAG_2}(u)) = 0.2$
- $f(Q^{CAG_3}(u)) = 0.4$
- $f(Q^{CAG_4}(u)) = 0.8$
- $f(Q^{CAG_5}(u)) = 0.9$

The ITL of u3, denoted by ITL_{u3}*, is the average of all these rates:* $ITL_{u3} = \frac{1}{5} * \sum_{j=1}^{j=5} f(Q^{CAG_j}(u)) = 0.6$.

4.2.1 Target Quality Requirements

OSNs are highly dynamic. OSN environments know high and uncontrolled changes with continuous movement in profiles coming in and coming out, and with constant modifications to their contents [59]. The CbIV model assumes an implicit static approach for the learning phase and it does not address the dynamic aspects in the evaluation phase. In fact, whilst the learning can be perceived as a static operation performed over a fixed period and considered final for some considerably long time, the evaluation phase has to be a constantly live one through which profiles continuously move to be rated. These profiles might be new ones to the OSN as they might be old but newly updated ones. Considering the rates of growth and contents evolution of OSNs, the loads of these movements are expected to be considerably high. Accounting for this continuous and massive flow of target profiles through the evaluation phase requires an adequate system design that effectively supports and efficiently handles this dynamic nature. For this purpose, we aim in this paper to provide a framework that assures a *continuous evaluation of profiles* without affecting the core of the system, i.e., the community based evaluation.

Reliance on community might threaten system operability and evaluations' impartiality. Both the system's learning and evaluation phases rely on community feedback. Whilst the learning phase might tolerate low or slow responsiveness from the community, the evaluation phase cannot sustain without its active and timely contribution. As a matter of fact, a profile cannot be endlessly pending in the evaluation phase waiting for raters responses. The system shall minimize such a risk or provide alternative solutions. One, or the most appealing solution to this issue, is to provide incentives and motivational rewards to encourage participation in the system. When we do not question the effectiveness of this solution, we consider it extrinsic to the design of the system. With that, our aim is to find other solutions by design, that ensure that *all profiles are evaluated within at most a worst evaluation time* that has to be guaranteed by the system.

In addition, reliance on the community might create bias in evaluations based on the available raters for every target profile. For example, two target profiles might be similar and receive different evaluations based on the group of raters selected and/or available to rate each one of them. The system shall account for such a bias and provide guarantees on its minimization. Therefore, to ensure operability and impartiality in evaluations, the system design shall first *guarantee some evaluation timeliness* by enjoying some level of independence and freedom from the assumption of infinite availability of the raters. Second, it shall *minimize the bias in evaluations* between similar users and maintain a level of impartiality among them and across all raters.

Reliance on community raises privacy concerns. The system creates interactions and data flows between raters and target users profiles that might contain sensitive information. As a matter of fact, the system reveals users' profiles values to raters. These values, given some other background knowledge, could be used by the raters to infer the identity of the target profile or to link sensitive information about it. The seriousness of this is accentuated by the fact that the system, by functional design, performs raters selection. This selection results in having the target user and the raters be from the same group as per the values of the attributes under evaluation. Consequently, the probability that the selected rater knows or is close to the target profile increases, and so does the probability that the rater has more background knowledge about the target user. Thus, a crucial concern of the system derives from *a need for privacy preservation and protection of sensitive profile information even if a group of users collude.* This requires the system to ensure some anonymity guarantees by which raters are not able to re-identify the users they are rating and/or are not able to infer their private information.

Example 4.2.3 *Consider CAG_1 and CR_1 from Example 4.2.1. Suppose that the profile of u3 from Table 4.7 is going under evaluation. By the system's logic, u4, u5, and u7 will be selected to rate u3 on CAG_1 (u3, u4, u5, and u7 have the same value for city). Under these conditions, these following scenarios are possible:*

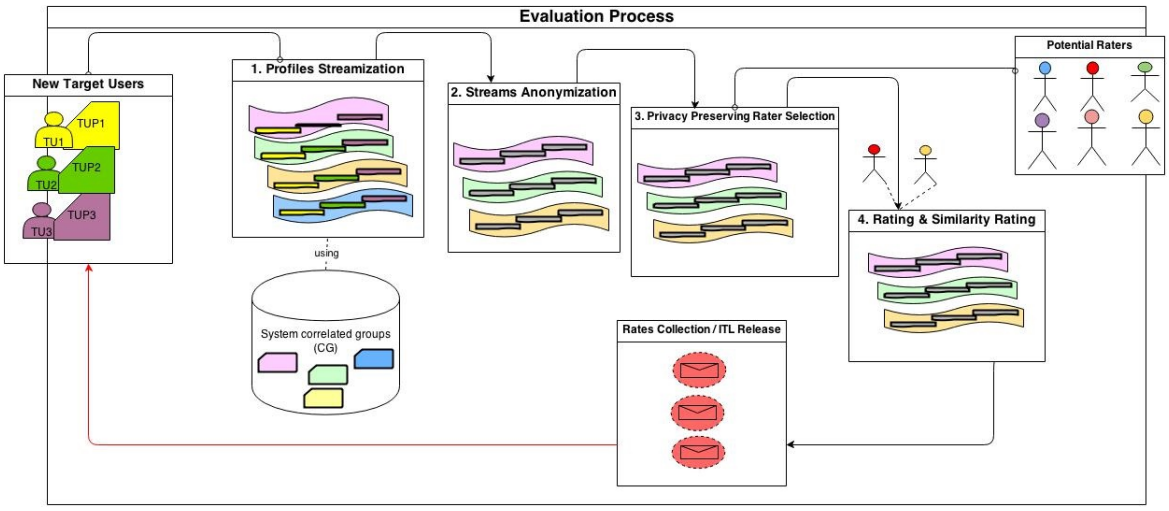


Figure 4.5 COIP's evaluation phase under the streams based design

1. u3 sets the job attribute as private, and one of the raters knows there are only four members from Tiny-City in the OSN, then she would link the job value presented to her for evaluation to the right identity with a probability of at least 1/3 (given this rater is not aware of the job value of any of the other 3 members of Tiny-City).
2. If u3's job attribute is public, and the raters know there are only four members from Tiny-City in the OSN, then the raters can identify who they are rating and can collude to provide positively or negatively biased evaluations.
3. If none of u4, u5, and u7 answer the evaluation request, u3 would be pending endlessly in the evaluation process.
4. Assuming that u7 was formerly evaluated by u5 on CAG_1 and u4 and that u7 responds to evaluate u3. The rate provided earlier by u5 to u7 might be so divergent from the rates that will be provided by u7 and u4 to u3. Hence, u7 or u3 who both are similar w.r.t to CAG_1 would be rated with a bias depending on the subjectivity of the users who rated each one of them.

Design Approach

We start by observing that the data dynamics in an OSN makes data flows in the system have similar characteristics with live data streams. In fact, we expect that user profiles move continuously through the evaluation process either as newly added profiles or as newly changed ones and come in big numbers changing through time. On this basis, we organize profile data of target users into separate data streams. Each stream, called *evaluation stream*, corresponds to a CAG identified during the learning phase. Evaluation streams go through the evaluation process by which their items get rated. Rates corresponding to a target user are then collected throughout the rated evaluation streams and aggregated to generate her

ITL. Figure 4.5 presents an overview of the architecture that we have devised. Target users profiles go through a *profile streamization* step, where they are decomposed in separate evaluation streams on the basis of the CAGs defined in the system. It is to be noted here that a target user profile can be both a new profile to the OSN or an already existing but newly changed one. Hereafter, the terms new profile or newly arriving profile refer interchangeably to both of them. Modified profiles will go through the evaluation as new profiles with the only difference that only CAGs affected by the modification will be evaluated.

To account for the model's requirements with regard to anonymity preservation, we anonymize evaluation streams before releasing them to get rated (i.e., step 2 in Figure 4.5). In particular, we adopt the k-anonymity principle, by which every record in a k-anonymized dataset is made indistinguishable from at least $k-1$ other records w.r.t some defined attributes [35]. We note that data streams have some specific characteristics making them different from other types of static data, and therefore, their k-anonymization requires a different treatment and introduces new requirements compared to static data.

Moreover, anonymizing CAG values (i.e., evaluation streams) might be not enough, since raters might have different levels of background knowledge making them able to infer private profile attributes of target users across the anonymized evaluation streams. Indeed, a well informed rater who is selected by the system to rate elements across multiple anonymized evaluation streams might end up collecting enough information to link to or to infer the identity of some target profiles.

Example 4.2.4 *Let us consider the CAGs CAG_1 and CAG_4 (see Example 4.2.1), which overlap on the Job attribute. Assume that u3 from Table 4.7 is going under evaluation and that a rater m has enough knowledge on how the validation system operates. Assume that u3 keeps his Age attribute private. In case m is selected to rate CAG_1 = {City, Job} and CAG_3 = {Job, Age} while u3's profile is still under evaluation, he might receive to rate these two pieces of information on u3: [Tiny City, Detective] and [Detective, 55]. Assuming that these two pieces of information are already k-anonymized each within its evaluation stream, m should not be able to differentiate them from at least $k-1$ other elements. However, intersecting these pieces on the attribute at which they overlap (i.e., the Job attribute), m would be able to infer that detectives in Tiny City are 55 years old, and hence uncover the age of the detective she knows from Tiny City though u3 keeps this piece of information private.*

To avoid such possible inference, we exploit a *privacy-preserving raters selection* strategy (see step 3 in Figure 4.5) by which the same rater is not selected for overlapping evaluation streams while the same profile is still under evaluation.

Finally, in order to maintain a level of impartiality across evaluations, we have to ensure that similar profiles are equally rated regardless of which raters the system has selected for their evaluation. For this, we introduce the *similarity rating technique*. This aims at

reusing rates already collected for a given CAG to judge that part of a target profile under evaluation. Since we exploit k-anonymity, this technique benefits from the fact that, by definition, elements in an anonymized evaluation stream should be similar. For example, rates collected for {Master, Engineer} can be extended for use to judge {Bachelor, Engineer}, as Master and Bachelor might converge to similar values after the anonymization (i.e., they belong to the same generalization hierarchy and they both converge to the same generalized value). The similarity rating technique not only assures the minimization of the bias in evaluations, but also minimizes the effort of the community and hence helps in offering guarantees on the worst evaluation time.

Before formalizing and detailing each step of the proposed design, we first discuss and formalize its grounding quality requirements.

4.2.2 Model Properties

Addressing the limitations discussed earlier requires considering some quality requirements. First, anonymity needs to be assured; however, it is commonly accepted that data anonymization goes in trade off with its utility for further analysis or usages [24]. For example, data anonymization incurs inevitable losses to data utility as some pieces of information have to be discarded (suppression) and as others have to be perturbed by the introduction of some kind of noise [24]. In our model, the anonymization of the evaluation streams might hinder the quality of the rates attributed to the CAGs they are part of. Accepting that this treatment might not be avoidable in meeting anonymization, we need its effects on the quality of rates to be kept within some tolerance boundaries. In addition to that, the system needs to ensure operability. That is guarantying that all target users can be evaluated effectively, with minimum bias, and in finite time using the existing resources (i.e., the available raters). The system quality requirements can be scoped in four main ones, described in the following. We provide proofs to how the suggested system design meets all these property requirements under **Appendix B**.

1. Anonymity assurance. The system shall meet an anonymity guarantee level by which raters are not able to re-identify the users they are rating. To measure this requirement, we suggest an anonymity guarantee property defined as follows:

Property - P1. (*Anonymity guarantee.*) Let u_t be an OSN user and let $Q^{CAG}(u_t)$ be the vector of profile attribute values of u_t corresponding to the CAG $CAG \in CAG^*$. Let SR be the raters selected for $Q^{CAG}(u_t)$. Let $\mathcal{B}_r$ be the event that a rater, $r \in$ SR, identifies u_t. The system's anonymity guarantee A_g is defined as a probabilistic metric such that $p(\neg\mathcal{B}_r) \geq A_g$, $\forall r \in$ SR.

2. Utility assurance. In meeting the first requirement, the system should still ensure a utility level by which the reliability and accuracy of evaluations are met within some

defined boundaries. We capture system utility by a *utility* metric which reflects the maximum tolerated loss in accuracy in the computation of an ITL for a target user. Formally, we define the following property for the system:

Property - P2. (*System utility.*) Let u_t be a target user whose profile is under evaluation. Let ITL_{max} be the value of ITL for u_t computed relaxing property P1 (i.e., no anonymization is applied) and let ITL_{anonym} be the one given property P1. Let *utility* be a minimum threshold for the acceptable system's utility. The utility of the system w.r.t u_t is given by: $utility_t = 1$ - (ITL_{max} - ITL_{anonym}). The system utility property ensures that, $utility_t \geq utility \; \forall u_t \in SN$.

3. Operability assurance. Whilst meeting the two previous requirements, and using the available resources, the system should ensure that all profiles in the OSN are evaluated with no exception and in finite time. That is, it has to ensure that all users in the OSN are evaluated within some given maximum tolerable waiting time. We formalize this property as follows:

Property - P3. (*Operability.*) Let t_{max} be a value representing the maximum waiting time for a profile to be evaluated. Let t_u^{start} be the arrival time of the profile of a user u. The operability property requires that, $\forall u \in SN$, $\exists \; t_u^{end}$ such that t_u^{end} - $t_u^{start} \leq t_{max}$, and at which an ITL for u can be computed by satisfying both properties P1 and P2.

4. Impartiality across evaluations. Given that profiles are rated by groups of selected raters that might vary across target profiles, either because of raters selection rules or because of raters availability, similar profiles might end up getting unequally evaluated based on the different groups of raters selected for each of them. Thus, one of the goals of the system should be to ensure a level of impartiality in evaluating profiles. We measure this impartiality between evaluations by a bias metric. A bias is the difference in final aggregated rates between two similar profiles resulting from the fact that two different groups of raters responded to evaluate each one of them. The goal of the system is to maintain this bias at a minimum acceptable level. We formalize this in the following property:

Property - P4. (*Impartiality.*) Let ∇_{max} be the maximum tolerated bias in evaluating a target profile. Let $\overline{CAG} \in CAG^*$ be a CAG. Let $Q^{\overline{CAG}}(u)$ be the values vector of a user u on $\overline{CAG}$ and let $f(\overline{CAG}_u)$ be the average of all rates provided for $Q^{\overline{CAG}}(u)$. Let $G(u) \subset$ SN be a group of SN users who have similar values for $Q^{\overline{CAG}}(u)$.[15] The impartiality property requires that: $\forall v \in G(u)$, $|f(\overline{CAG}_v) - f(\overline{CAG}_u)| \leq \nabla_{max}$.

4.2.3 Profiles Streamization

Since profiles are evaluated as sub parts corresponding to CAGs, we first need to split every user profile into separate tuples according to the CAGs identified in the learning phase.

[15] We define similarity as equal values or values that generalize to the same value after the anonymization process.

Moreover, as some attributes accept multi-values, their corresponding values-set may contain more than one value item. In this case, the user will have as many tuples corresponding to the CAG containing the multi-value attribute(s) as the different values provided by the user for that/these attribute(s), as the following definition clarifies.

Definition 4.2.1 *Correlated tuple of user u. Let* $\overline{CAG} = \{a_1, a_2, ..., a_z\}$ *be a CAG in* CAG^*, *where* $a_i \in S, \forall i \in \{1, \ldots, z\}$. *Let u be a user in the OSN and let* $Q^{\overline{CAG}}(u)$ *be her profile values set for the attributes in* $\overline{CAG}$. *The correlated tuple-set of u w.r.t* $\overline{CAG}$, *denoted by* $Tuple^{\overline{CAG}}(u)$, *is the set of all combinations of values in* $Q^{\overline{CAG}}(u)$:

$$Tuple^{\overline{CAG}}(u) = \{Q_1^{\overline{CAG}}(u) \times Q_2^{\overline{CAG}}(u) \times .... \times Q_z^{\overline{CAG}}(u)\}.$$

An element belonging to $Tuple^{\overline{CAG}}(u)$ *is a correlated tuple of user u.*

Example 4.2.5 *Let us consider user* $u1$ *from Table 4.7. Since she specifies two values for hobbies,* $u1$ *has two correlated tuples for* CAG_2: $\{Stanford, hiking\}$ *and* $\{Stanford, biking\}$. *As a result,* $Tuple^{CAG_2}(u1) = \{\{Stanford, hiking\}, \{Stanford, biking\}\}$.

A user profile is evaluated based on all its corresponding correlated tuples, but the tuples are evaluated independently; i.e., each correlated tuple of a target user profile is individually rated. For that, given a CAG, we organize the correlated tuples of all available target users into a data stream. We call this data stream, an *evaluation stream*.

As it will be explained later, our solution needs to keep track of the arrival times of tuples within streams. Hence, the sequenced tuples within an evaluation stream are enriched with a time stamp, *tp*, which stores information on the arrival time of the corresponding target user profile. For simplicity, we consider the position of the tuple in the stream as an abstract representation of its arrival time. Given a tuple l in an evaluation stream $\mathcal{T}$, $l.tp$ denotes the attribute of tuple l which stores its time position. Since correlated tuples of a same user u for the same CAG are logically arriving to the stream at the same time, we require that the time position is not incremented when the stream admits correlated tuples referring to the same user. That is, more than one tuple in a stream might have the same time position given they belong to the same profile.

Formally, we define an evaluation stream as:

Definition 4.2.2 *Evaluation stream. Let* $\overline{CAG} \in CAG^*$ *be a CAG. The evaluation stream for* $\overline{CAG}$ *is defined as a data stream* $\mathcal{T}^{\overline{CAG}}$ *with the following schema:* $(tp, uid, Tuple^{\overline{CAG}}(u))$, *where* $Tuple^{\overline{CAG}}(u)$ *is the correlated tuple-set of a user u going through evaluation w.r.t* $\overline{CAG}$, *tp is these tuples position within the stream, and uid is user u's identity.*

Based on Definition 4.2.2, there will be, in the evaluation process, as many evaluation streams as the number of the CAGs defined in the system. Data within each of these streams need to go through the evaluation process to get rated by selected raters. Towards meeting the anonymity requirement, we make the evaluation streams subject to an anonymization process before releasing their content for rating.

4.2.4 Evaluation Streams Anonymization

Our goal is to meet the anonymity quality requirement through anonymizing evaluation streams before releasing their items for rating. The literature on anonymization is rich, but most pieces of work are targeted for static collections of data. In fact, anonymizing data streams, that are continuous and short-lived, requires dealing with guarantees on some maximum delay in releasing the anonymized version of any incoming piece of data [28]. This delay constraint makes the anonymization process bounded to offering alternatives for those data items that are about to exceed the delay, but that cannot be anonymized yet within the available stream data. These alternatives might range from suppression of the item to its generalization at different levels. In general, traditional anonymization methods, such as k-anonymity [139], l-diversity [90], etc, are adapted to answer the requirements of data streams, especially related to freshness and utility. These methods follow a semantic approach to anonymization where the objective is to modify the input data at an individual item level to meet a given property. For example, items in the input data might be generalized, such as replacing the full date of birth by only year of birth, to generate an anonymized output where one cannot differentiate between its different records w.r.t the generalized attribute. Those semantic based models could be argued to be obsolete or less effective against re-identification compared to differential privacy techniques [35]. Differential privacy operates with a different approach in that it does not directly generate anonymized data at a micro level. Instead, it relies on adding some statistical perturbation to the result of a given query on the data, requiring that, given any two input datasets that are different only in one data item, the distribution of the perturbed output on these two datasets is almost the same [85]. This ensures a privacy guarantee on every data item in the form of an equal probability to be re-identified regardless of whether or not it belongs to the input dataset to be anonymized. Differential privacy might be offering stronger guarantees on anonymity and privacy preservation; however, given the noise it requires to add to the data, it might come with considerably high costs on data utility that change based on target tasks and on the nature of input data to be anonymized [45]. Such costs might be acceptable for some mining or statistical tasks as the output would be treated in mass; however, in our context, the anonymized values are needed at an individual level. That is, every value in a record is required with the minimum noise or change possible to offer reliable rating. Moreover,

differential privacy has been mostly adopted for static large collections of data and is still not fully deployed to continuous and real-time data, which is our target scenario [46].

In adopting any of the above mentioned anonymizing techniques we had to consider properties P2 and P3 by which anonymization shall not affect the quality of the rating, and by which it is required that all profiles are rated within some given time interval. Indeed, in our context, profile data arrives to the evaluation streams at quasi-stochastic times and with different load distribution over time. As such, an ideal anonymizing process needs to: ensure guarantees on anonymization time constraints, meet adaptability to the distribution of the incoming stream data, and prove effectiveness with regard to the utility of the data after it is anonymized. Given the high requirements on utility and timeliness, and given the discussion above, we have not considered differential privacy techniques. Instead, we have focused on adopting semantic techniques accepting the lower guarantees on privacy that they provide, especially since we couple anonymization with a privacy preserving raters selection to provide the final privacy guarantees of our system.

In studying available anonymization algorithms on semantic techniques for datastreams, one of the available work that seems to better fit our demands is the time constrained scheme for k-anonymizing data streams (CASTLE) presented in [28]. This scheme is developed for anonymizing stream data by considering its specificities especially those related to data freshness and to utility after anonymization.

CASTLE

CASTLE adopts the k-anonymity principle to anonymize data streams. This is done by intercepting arriving tuples in the stream and grouping them, based on their similarity, into clusters. The first cluster is built upon receiving the first tuple. Then, with every new tuple arrival, CASTLE selects the cluster which best fits it. This fitness depends on the similarity level of values between the new tuple and the already existing ones into the available clusters. CASTLE considers the possible generalization scenarios[16] required to fit the new tuple within each of the available clusters and selects the one that incurs the minimum information loss.[17] When no available cluster can enclose the new tuple within a given average information loss threshold τ, a new cluster is built around it. When the size of a cluster is at least k, CASTLE releases its contained tuples with its generalization value ensuring by this that the output tuples are k-anonymized. For example, a cluster which contains tuples (25, PhD), (30, PhD), and (22, PhD) will release them with the generalization ([22,30], PhD) by satisfying k-anonymity for $k = 3$.

[16] This implies substituting the original values of tuples by upper level ones from their domain generalization hierarchy.

[17] CASTLE quantifies the information loss incurred by generalizing a tuple by an infoLoss metric adopted from Iyengar as cited in [28].

A nice property of the CASTLE scheme is that the k-anonymization happens with respect to a data freshness constraint, by which intercepted tuples are ensured to be output within k-anonymized groups at worst within a fixed delay δ. CASTLE ensures respecting this δ time constraint by dedicating special treatment to the *expiring* tuples. A tuple l with position $l.tp$ is considered expiring when the newer tuple f with position $f.tp = l.tp + \delta$ gets into CASTLE. CASTLE addresses tuples expiry by means of clusters merging. Indeed, when a tuple within a non k-anonymized cluster is expiring, CASTLE looks for other nearby clusters with which it can be merged all while being under the information loss threshold τ. In case no such merge option is possible, the expiring tuple is allowed to be suppressed. To minimize suppression, CASTLE suggests keeping a copy of k-anonymized clusters in memory after releasing them. This copy helps in reusing this data to possibly contain some new tuple that else would need to be suppressed.

Another feature of CASTLE is its adaptability to the distribution of data streams w.r.t the definition of the information loss threshold τ. In fact, CASTLE does not assume a predefined fixed τ, but it dynamically defines it, based on the nature of the input stream data. To achieve this, it assumes an input parameter μ that reflects the expected variance of the incoming data distribution. Then, τ is set to the average information loss of the μ most recent k-anonymized and released clusters. The advantage of this dynamic setting of τ is to allow the algorithm generate small clusters with small information loss when the stream data is inherently well clustered, and to allow creating larger ones with larger information loss when the data is sparse. However, this might cause the algorithm to generate many small clusters when the incoming data is first dense and then changes to be sparse (starting with a small τ and facing a sudden change in incoming data density) causing an increase in the overhead of searching for best fitting cluster for every new coming tuple. To account for this, CASTLE suggests a parameter β that serves to limit the number of allowed created clusters. The authors suggest that β can be set based on the available computational and storage resources.

Adapting CASTLE to Evaluation Streams

We adapt and apply CASTLE to each of the evaluation streams independently, as illustrated on Figure 4.6. CASTLE comes with parameters and features that fit well within the conception of our system; however, some of these parameters require adjustment and remodeling to satisfy our requirements. As we show in what follows, this adaptation requires differentiating between CAGs based on their relevance. For this, we consider the support of a CAG to be an indication of its weight. We use the terms weight and support interchangeably and we refer to them by $Supp(CAG_i)$ for every $CAG_i \in CAG^*$.

In addition to this, and given that CASTLE leverages on k-anonymity by which it is required to set the sensitivity of the attributes in the data to be anonymized in terms of

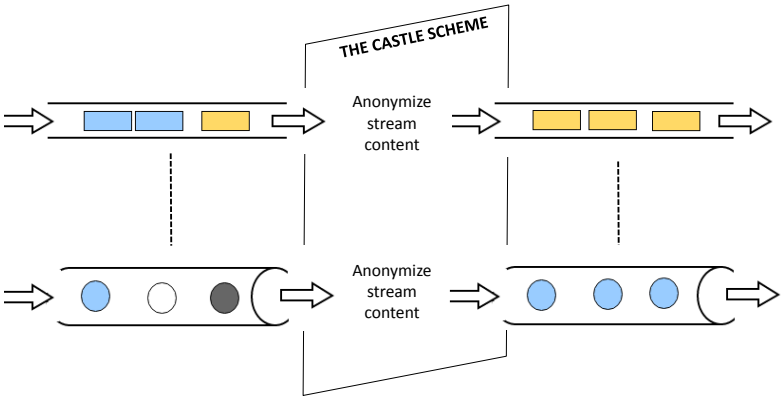


Figure 4.6 Anonymizing evaluation streams using CASTLE

identifiable and non-identifiable ones, we note here that we consider all attributes within a CAG to have the same sensitivity level and to be all identifiable attributes. In the terminology of anonymization, the term identifiable attribute, or personally identifiable attribute, or quasi-identifier, as sometimes referred to in the literature, is an attribute which, if joined with other pieces of information, would reidentify its holder [34]. For example, a full address field can be a quasi-identifier, as it would refer to a specific domicile and hence to a specific known person. As this sensitivity of attributes depends on the underlying data, and given we do not have static knowledge on the dynamic data in an OSN, we consider all attributes that are to be anonymized within an evaluation stream to be quasi-identifiers.

Time constraint on an evaluation stream

CASTLE's time constraint property fits well within our context as we shall ensure timeliness in evaluating target profiles. However, an equally important concern in our scenario is the utility of the anonymized data. Hence, δ has to be carefully set to allow meeting timeliness without incurring much damage to utility. Indeed, since the values in anonymized tuples are to be presented to raters for evaluation, these values need to be conserved as much as possible to allow sound evaluation. Moreover, tuples corresponding to a CAG with high support are expected to have more weight in the evaluation, and so the higher the support of a CAG is, the higher its utility should be. For this, we suggest setting δ on an evaluation stream based on the support of its corresponding CAG. At the same time, the higher the support of a CAG is, the more effect its rating has on the final ITL and hence the soonest it has to be released. This accentuates the trade-off between timeliness and utility w.r.t the high support CAG. Herein, we got two options: opt for utility and relax the time constraint, or

constrain the time and find alternative solutions for the utility. We adopt a hybrid approach, by relaxing the time constraint in favor of utility, but only to some limit at which workaround solutions are applied. With this, we make the δ constraint on an evaluation stream directly proportional to the support of the corresponding CAG. A CAG with a low support has lower effect on the evaluation and is meant not to delay this latter waiting to get anonymized; hence its time constraint would be more strict. Formally we define:

Definition 4.2.3 *Time constraint on an evaluation stream. Let $\mathcal{T}^{\overline{CAG}}$ be an evaluation stream. Let AR be a non-zero positive constant that represents the frequency of arrival of new tuples to the stream. Let $Supp(\overline{CAG})$ be the support of the CAG $\overline{CAG}$. The CASTLE's δ value for $\mathcal{T}^{\overline{CAG}}$ is given by: $\delta(\overline{CAG}) = AR * Supp(\overline{CAG})$.*

When the time constraint on an evaluation stream is fired, expiring tuples are not further generalized or merged as suggested in CASTLE. Rather, these expiring tuples are treated as special cases as described in what follows.

Dealing with expiring tuples

As discussed, the information loss is a major concern as over-generalized tuples might be misleading as of the rates they receive. To avoid having high information loss, these expiring tuples can be dealt with in two ways: (1) rate them outside of the anonymization framework by relying on the existence of a group of trusted users who are to be entrusted to rate the non-anonymized expiring tuples. (2) inject within the stream enough dummy tuples, similar to the expiring ones, with which CASTLE can cluster them for k-anonymization. The first way has two drawbacks: on the one hand, it relies on the trusted users assumption that puts the privacy of the tuple subject to its realization. On the other hand, trusted users might not be best placed or informed to rate the tuple in question. The second way addresses these two drawbacks but might result in flooding the system with dummy information and inefficiently increase the effort of the community. However, this might not be an issue as we expect the occurrence of such special case expiring tuples to be minimal. This is because the homophily effect is expected to increase chances for similar profiles to join the network at close times [59]. To not manipulate the logic of the system, we opt for option (2) and we experimentally study its overhead (see Experiments subsection).

Information loss threshold

CASTLE considers the allowed information loss when deciding on fitting tuples within clusters to be dynamically set based on the distribution of the data. In our context, utility remains always of much higher importance and hence the allowed information loss has to be preliminary known and fixed. We make the information loss threshold τ inversely proportional

to the support of the CAG corresponding to the evaluation stream. With this, a CAG with lower support will allow higher τ value to its stream; whereas, a CAG with higher support will tolerate lower τ value. Since we consider τ to be preliminary defined and fixed, we do not make use of the μ and the β parameters in the CASTLE algorithm.

Discussion

Defining and scoping privacy has been challenging across several fields and is as old as law and criminology disciplines [17]. Our aim in this work is not to re-invent or to define privacy and how to scope it. Our aim is to be aligned with common known standards, in the field of computer science, for protecting data from disclosure when target profiles are evaluated. Some might criticize our reliance on k-anonymity for ensuring privacy. This critique might be supported by works showing weaknesses of k-anonymity and of its variants, as cited and discussed in [35]. The point here is that the picture and the scale in our context is different. On the one hand, most of the de-anonymizing algorithms discussed over online social networks' data rely on the publishing of edges' information (i.e., the social graph) and on the merging of social links and released information on top of them across multiple social networks. This scenario does not apply to our scenario in which the system does not have to be aware of the social links and it is not using them. On the other hand, profile attributes' sensitivity and probability of identification based on them is already cut down by their streamization based on CAGs and by their temporal distribution based on their arrival time. That is, profile information is not published all at once and/or within more informative groupings. In addition to that, the release of information in our context is done not for mining purposes but for evaluation ones at an individual scale. Relying on semantic anonymization sounds best fitting within our context and its possible shortcomings are down-cut by the simple logic of the design that considers all elements of a tuple to make sensitive information. Moreover, in applying k-anonymity to the evaluation streams, we do not differentiate between personally identifiable and non-personally identifiable attributes. Indeed, as argued in [101], one of the challenges in successfully preserving the privacy of personal data using semantic anonymization is the pitfall of distinguishing between personally and non-personally identifiable attributes.

Having said all that, we do not claim, by any means, that we ensure full privacy preservation. Risks of disclosure cannot be completely mitigated and can only cut down to acceptable levels. That is, what we achieve is a guarantee on privacy within given boundaries. We meet this by the suggested anonymization technique coupled with a privacy preserving raters selection, as will be presented in the following section.

4.2.5 Privacy Preserving Raters Selection

Correlated tuples of the same profile are anonymized each within its corresponding evaluation stream independently of the others. This ensures the targeted anonymity of a piece of a profile, but it does not necessarily ensure the same level of anonymity for the corresponding user. More specifically, if two CAGs overlap, then the combination of their anonymized evaluation streams' data is not guaranteed for the same level of anonymity, as the following example shows.

Example 4.2.6 *With reference to Example 4.2.1, suppose that at some instant the evaluation stream for CAG_4 contains the correlated tuples of users $u1$, $u2$, and $u6$, whereas CAG_5's stream contains the tuples of users $u1$, $u7$, and $u5$. Assume that we are under 3-anonymity and so the two streams are anonymized with these generalizations:*

- *CAG_4's stream: {[Student], [21,24]}*
- *CAG_5's stream: {[Stanford], [20,30]}*

Assume a rater knows that $u1$ is a student at Stanford and that $u1$'s profile is under evaluation. This rater can minimize the age range from [20,30] as is in the anonymization of CAG_5 to [21,24] which is the anonymization range for the age attribute in CAG_4.

As illustrated by Example 4.2.6, the inference that a knowledgeable rater can make is only possible given that the two CAGs share at least one common attribute. We note that a knowledgeable rater is one who knows how the system's logic works and who might have been listening to the streams and collecting data thoroughly.

The possibility of inferring information from overlapping evaluation streams gets higher since the system performs raters selection by which raters share common attribute values' ranges with the tuples they rate. This increases the probability that raters come from the same community as the target user and hence have more background knowledge about the target profile. To overcome this problem, a first solution is to make the system anonymize the data in overlapping evaluation streams by the highest of their generalizations. For example, CAG_4's stream from Example 4.2.6 would be anonymized with the generalization {[Student], [20,30]} instead of {[Student], [21,24]} as it intersects with CAG_5 at the Age attribute and as CAG_5 got a higher generalization for the Age. However, this will result in undesirable lower utility of tuples' information. Therefore, we address this at the level of raters selection by introducing a constraint that ensures that one given rater is not selected to rate intersecting evaluation streams within a given time span (i.e., the maximum time tuples of the same profile from these intersecting evaluation streams can coexist). We call this, anonymity preserving raters selection, and we define it as follows:

Definition 4.2.4 *Anonymity preserving raters selection. Let CAG_h and $CAG_p \in CAG^*$ be two CAGs. Let u be a target user under evaluation and let $RS_h(u)$ and $RS_p(u) \in RS$ be the selected raters to rate the anonymized tuples of u in CAG_h's and CAG_p's evaluation streams, respectively. Let $\delta(CAG_h)$ and $\delta(CAG_p)$ be the δ anonymization constraint on CAG_h's and CAG_p's evaluation streams. $RS_h(u)$ and $RS_p(u)$ are anonymity preserving iff they satisfy this condition:*

$$\textit{If } CAG_h \cap CAG_p \neq \emptyset, \textit{ then } RS_h(u) \cap RS_p(u) = \emptyset \textit{ over an evaluation time span } \Delta t = max(\delta(CAG_h), \delta(CAG_p)).$$

As such, the system satisfies anonymity preserving raters selection iff $\forall RS_i(u)$ and $RS_j(u) \in RS$, $RS_i(u)$ and $RS_j(u)$ are anonymity preserving.

4.2.6 Rating Profiles

There are two main concerns to address regarding the rating of anonymized tuples. First, the collection of rates for an anonymized tuple has to account for the level of generalization it went through during the anonymization process. Second, the collection of rates has to ensure minimum bias between similar tuples.

Rates precision

The anonymization of evaluation streams results in an information loss in the utility of the anonymized tuples. Though we restrict this information loss to remain below a threshold τ, anonymized tuples do still undergo different levels of generalization. As such, an evaluation rate provided to an anonymized tuple is not expected to be as accurate as the one given to its un-anonymized value. To account for this and to reflect fairness in evaluating the tuple, we introduce a rate precision factor that adjusts the weight of the rate based on the amount of information still contained within the anonymized tuple w.r.t its un-anonymized value. To define this factor, we leverage on information theory [30][37]. We exploit the concept of self-information, also referred to as entropy, which is a measure of the amount of information contained in a variable X with probability X_i [37][10]. Self-information quantifies the uncertainty in a variable X and hence provides a measure of its delivered information [37]. In our scenario, we need to quantify the amount of information contained in an anonymized tuple given its information loss. Based on the information theory, this is the entropy of the anonymized tuple t given its associated information loss $InfoLoss_t$ and it is given by the logarithmic function: Entropy(t) = -ln($InfoLoss_t$). The logarithmic function can be used with different bases depending on the adopted unit of information measure [10]. We opt here for the natural logarithm, and thus the base is e. Given these elements, we formally define the rate precision factor as follows:

Definition 4.2.5 *Rate precision factor - PrecFactor. Let u be a target user and let* $tuple^{\overline{CAG}}(u)$ *be her tuple for* $\overline{CAG} \in CAG^*$. *Let InfoLoss be the information loss incurred by a generalization of one element e of this tuple,* $e \in tuple^{\overline{CAG}}(u)$. *The precision factor of the rates provided to the tuple* $tuple^{\overline{CAG}}(u)$ *after generalization of element e is:* $PrecFactor_e$ = *{1, if InfoLoss* $\leq \frac{1}{e}$*; -ln(InfoLoss), otherwise}. Since the elements within tuples are correlated, their precision factors are considered dependent. That is, the precision factor of a tuple* $tuple^{\overline{CAG}}(u)$, *denoted by* $PrecFactor(\overline{CAG}_u)$, *is:*

$$PrecFactor(\overline{CAG}_u) = \prod_{\forall e_i \in tuple^{\overline{CAG}}(u)} PrecFactor_{e_i}$$

Example 4.2.7 *With reference to Example 4.2.1, assume user u*1 *had her tuple for* $CAG_5 = \{Education, Age\}$ *anonymized by changing it from {Stanford, 24} to {A California state Univ, [20,25]}. Assume the information loss of generalizing her Education value is 0.4 and the one of generalizing the age is 0.1. The rate precision factor associated with this tuple is: 1 * -ln(0.4)* $\simeq$ *0.92.*

Rates by similarity

The k-anonymized output of an evaluation stream is a group of at least k equal tuples. These tuples are expected, given impartiality in ratings, to get equally evaluated by the community. For this, rating one of these tuples is enough to propagate the same rates to all its other equal fellows. Therefore, we pick one anonymized tuple out of every released generalization in an evaluation stream as a *representative* of its other equal tuples. The total average of all rates gathered for a representative is then propagated to all its other fellows in the same generalization. Moreover, this propagated rate is saved for reuse for future similar generalizations. That is, when the system gathers an aggregated rate for a representative tuple, it saves this rate associated to the corresponding generalization. When other tuples are anonymized with the same generalization, this saved rate is applied to them instead of making them go again through the cycle of feedback collection from raters.

Such propagation and reuse of rates benefit the system with the minimization of the effort of the community and also the assurance of fairness across similar tuples within the same anonymized stream data.

Rates aggregation

In CbIV, rates to CAGs of a target profile are averaged to compute its ITL (see Section 4.1.2). Here, we need to reconsider the ITL's definition given the new adapted parameters. We propose a weighted average in which the weight of a rate is defined by its precision factor and by the support of the CAG to which it is provided. We define the following:

Definition 4.2.6 *Rate's potency. Let $Supp(\overline{CAG})$ be the support of $\overline{CAG} \in CAG^*$. Let u be a target user and let $PrecFactor(\overline{CAG}_u)$ be the precision factor for the rates provided to her $tuple^{\overline{CAG}}(u)$. Let $f(\overline{CAG}_u)$ be the total average of all the rates gathered for $tuple^{\overline{CAG}}(u)$. The potency of $f(\overline{CAG}_u)$, denoted by $f(\overline{CAG}_u)^{pot}$ is given by:*

$$f(\overline{CAG}_u)^{pot} = Supp(\overline{CAG}) * PrecFactor(\overline{CAG}_u) * f(\overline{CAG}_u)$$

Based on Definition 4.2.6, the weighted ITL is computed as follows:

Definition 4.2.7 *Weighted ITL. Let u be a target user. Her weighted ITL is the average of the potency of all its gathered rates for all the available CAGs:*

$$WeightedITL(u) = \frac{1}{|\overline{CAG}^*|} * \sum_{\forall \overline{CAG} \in CAG^*} f(\overline{CAG}_u)^{pot}$$

The ITL measure is computed for each profile independently from the overall distribution of profile information across other profiles in the OSN community. That is, an ITL provides an estimation of the trustworthiness of a profile, but does not provide statistical guarantees normalized over the user population. Devising a statistical model that provides guarantees on the reliability of the ITL measure within the values distribution of a community would provide better insight on the truthfulness of a profile. However, this would require conducting empirical studies on annotated datasets that are representative of the different scenarios and data elements across different OSN communities. We acknowledge the importance and value of performing such a study and we believe that it represents enough research material for a different piece of work. Therefore, we consider this as a future work possibility.

4.2.7 Experiments and Results on COIP

The experiments reported hereafter are on the main objective of this work; that is, proving the feasibility of our suggested method within the dynamics of OSNs. In designing these experiments, we had two main goals: (A) to verify that, given the variance and the distribution of profiles' data in an OSNs, k-anonymization of evaluation streams can be performed within acceptable delay, all while preserving acceptable utility; and (B) to study the feasibility of the model w.r.t raters' availability for every target profile.

Feasibility of anonymization

As we failed to find a representative dataset from a real OSN that contains enough attribute values in its profiles, we opted for the UCI-Adults dataset[18] for this first experiment on the feasibility of anonymization. This dataset contains demographic information on a sample from the 1994 US Census database with 32561 records (30162 records after removal of records with

[18] Available at: https://archive.ics.uci.edu/ml/datasets/Adult

Table 4.8 Correlated attribute groups used to construct evaluation streams.

CAG_a	{Education, Hrsperweek}
CAG_b	{Occupation, Hrsperweek}
CAG_c	{Occupation, Educ-num}
CAG_d	{Occupation, Work-class}

missing values) consisting of 14 attributes (8 categorical attributes and 6 numerical attributes with varying ranges and hierarchy levels). It is a commonly used benchmark dataset for k-anonymization, and we believe that it represents meaningful data for the objective of this experiment (i.e., anonymizing profile values that describe people's identities in an OSN within limited time and quality thresholds). In addition to that, it is the dataset we used in CbIV to verify the existence of CAGs over a profile schema (see Section 4.1.3). To verify the feasibility of anonymizing evaluation streams we have considered four CAGs (presented in Table 4.8) out of the total nine previously identified over this dataset (refer to Table 4.4 for the full list of CAGs previously identified).

We implemented the CASTLE anonymization algorithm considering the adaptations of the delay constraint δ and of the average information loss threshold τ. Intuitively, within each evaluation stream, the two attributes of the corresponding CAG were set as quasi-identifiers. For the definition of generalization hierarchies, several proposals exist in the literature. For the attributes in our CAGs, we have adopted the taxonomy trees used and publicly shared in [54].

Our goal is to study the feasibility of reaching an acceptable average information loss (AIL) within acceptable delays. To approximate this within our experimental setting, we had to set assumptions w.r.t the arrival rate and to the variance of incoming profiles. These assumptions have to best represent the dynamics of a real OSN for the soundness of our study. We managed the arrival rate by calculating the average number of new profiles per day in Facebook based on the statistics collected and parsed in [52].We then projected this number as a percentage on the number of records in the Adults dataset; i.e., an average of approximately 0.3M new users per day for a total of 1184M users in Facebook as in [52] is projected to an average of 8 users per day for the Adults' dataset's size. As for the variance, we assume the random selection of arriving profiles. The motivation behind is that proving feasibility with a random variance, which reflects the worst case scenario, is a granted proof for a real-OSN's one where homophily is expected to decrease variance among joining profiles. Intuitively, a decreased variance suggests more density at the level of similar attributes and hence increases the feasibility of k-anonymizing within smaller delays and at lower AIL.

We ran the experiment varying values for the arrival rate (AR) of the new joining profiles around the estimated value of 8 new profiles per day. More precisely, we varied AR between

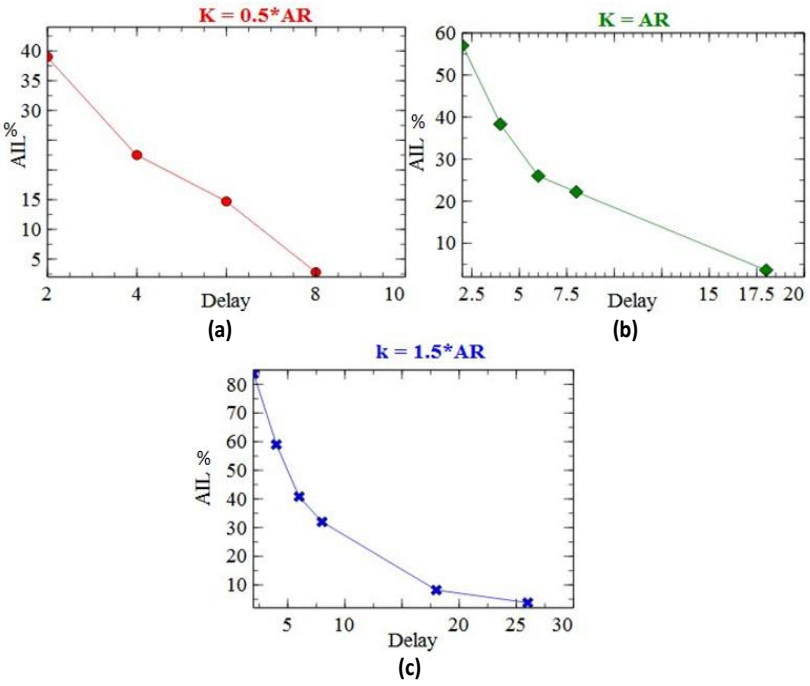


Figure 4.7 Average information loss vs. k-anonymization delay for different k.

4 and 16 new profiles per time unit with steps of 4. In setting k for the experiments we considered the arrival rate, i.e., AR. This is because the ideal scenario would be that for every group of tuples arriving to the stream at the same time, their anonymization happens without any delay and with the highest possible value for k. This means that an ideal case is to be able to k-anonymize the stream with $k = AR$. For this, we ran the experiment fixing k to AR, $0.5 * AR$, and $1.5 * AR$, and we computed the average information loss (AIL) across the four evaluation streams varying the anonymization waiting time of tuples, i.e., *Delay*. We report the achieved results in Figure 4.7.

In Figure 4.7, we can notice that the AIL decreases as the delay increases and that it drops to very low values given enough waiting time per the adopted value for k. This result is, in fact, expected as it has already been demonstrated in [28], but it is not the aim of our study. Indeed, our goal is to test the feasibility of achieving anonymization with acceptable information loss and within acceptable time delays in a real OSN setting. For this, we need to read the results in Figure 4.7 considering the arrival rate of profiles in an OSN. For example, from Figure 4.7, we can read that, for an arrival rate of 8 random new profiles per day, profiles have to wait for approximately an average of 17 days to get 8-anonymized with an average information loss lower than 3% (Figure 4.7.b). We consider this a very promising result if projected back to the growth numbers within a real OSN (i.e., our Facebook adopted statistics). Indeed, given the adopted Facebook statistics and the results on the Adults

dataset as reported in Figure 4.7, evaluation streams are expected to get 300-anonymized within only 1 hour waiting time. To verify the validity of such an approximation, we ran the anoymization experiment on a real OSN dataset with profile values for two attributes.

Anonymization with real OSN data

We used the Berkeley Google+ dataset used in [58] as the only one we could find that contains some profile information. This dataset was collected by first crawling Google+[19] on July 6, 2011. This first crawl made what is referred to as the dataset's first snapshot. After that, the dataset was augmented by adding daily crawled consecutive snapshots until October 11, 2011. The first snapshot of the dataset and three daily augmentations have been publicly released.[20] Each snapshot represents the friendship graph made of nodes (i.e., user profiles) and undirected edges between them. For each node, values for four attributes, namely, *employer*, *school*, *major*, and *places-lived*. The first snapshot, which also makes the base of the crawl, contains 4339311 nodes (a node represents a user profile) and 47130326 edges (an edge represents a friendship between two nodes). Analyzing the graph in this snapshot, we understand that it is a representative sample of an OSN's expected growth and expected characteristics. Therefore, we only make use of this snapshot and we refer to it, hereafter, by the *G-dataset*.[21]

We considered $\{employer, places\text{-}lived\}$ to be a CAG and we considered new profiles to be joining at an arrival rate AR = 482145 profiles/day.[22] We considered the arrival of nodes to be at random. We fixed $k = 100$ and we considered a node to be expiring if it cannot get k-anonymized within the day of its arrival.We did not allow any suppression and we opted for injection of similar dummy tuples instead. We plot in Figure 4.8 the number of expiring nodes against the needed number of dummy nodes to anonymize them for each of the 9 days.

What we can learn from Figure 4.8 is that, though the number of expiring nodes knows a quasi-linearity across the days, the number of needed dummy tuples decreases considerably and is less than the number of expiring tuples starting from day 4. This is because the number of expiring tuples in the first days correspond to multiple small in size clusters; whereas it corresponds to bigger and lesser clusters as the network develops. This provides considerable evidence to support opting for inserting dummy tuples as an efficient way for saving both utility of the tuples and the logic of the system. However, we also realize the restrictive scenario of the experiment with only one CAG (a restriction imposed by the available profile values in the dataset), and therefore, we cannot withdraw generalized conclusions.

[19] www.plus.google.com

[20] Available at: http://aerie.cs.berkeley.edu/release/gplus.tar.gz

[21] Analytical charts of the G-dataset that demonstrate its representativity of a real OSN's expected evolution are available at: $http://strict.dista.uninsubria.it/?page_id = 676$

[22] Google+ was launched on June 28, 2011 [58] and the G-dataset was collected on July 6, 2011. We considered a simplified fixed daily arrival rate dividing the number of users in the G-dataset by the number of days from the start of the OSN to its collection point.

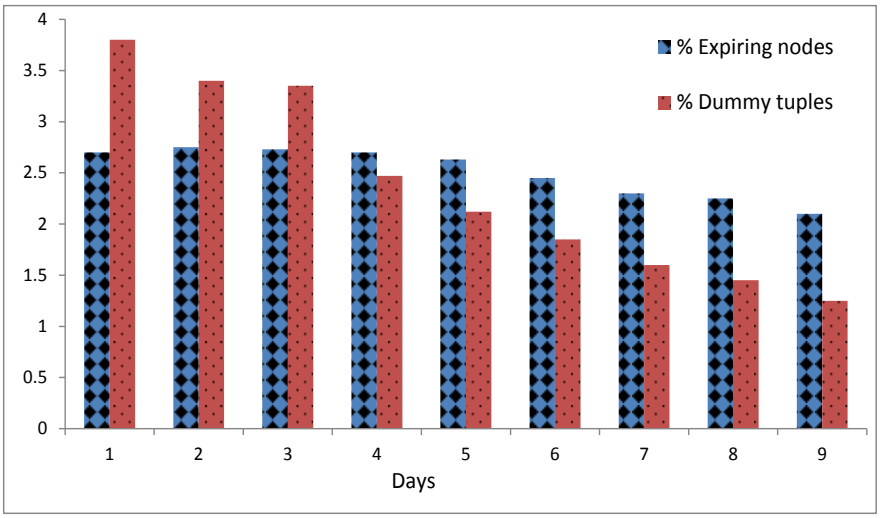


Figure 4.8 Expiring and dummy nodes for $k = 100$.

Raters' availability

The goal of this experiment is to study whether there are enough potential raters in the OSN for every target profile going through evaluation. For this purpose, we need to have a dataset which represents the evolution timeline of the OSN. As, to the best of our knowledge, there does not exist such a dataset, we have leveraged on the work in [77] where the authors suggest a framework for reconstructing the evolution timeline of a given OSN network graph, based on degree assortativity of nodes. Using the tool developed in [77], we could get an evolution timeline of the G-dataset.[23] We limited the number of reconstructed evolution snapshots of the G-dataset to 10 with snapshot 1 referring to the state of the graph at time 0, and snapshot 10 referring to its final state, i.e., the G-dataset's graph. Given that the G-dataset corresponds to 10 days of Google+ existence, we consider each of these snapshots to be corresponding to daily states of the Google+ graph represented in the G-dataset.

For testing purposes, we assumed that the attributes *employer* and *places-lived* form a CAG with coherence relation ($\{employer \longrightarrow places\text{-}lived\}$). Thus, raters selection on this CAG would require selecting users sharing the same value for the *employer* attribute. Then, in each snapshot n ($n \neq 1$) of the G-dataset, we consider the nodes in snapshot n-1 to be the pool of all potential raters and the new nodes appearing in snapshot n to be the target users. We then calculate the number of users from the raters' pool who are eligible by the assumed coherence relation (i.e., share the same value for *employer* as a target user). At every snapshot n we calculate the percentage of target nodes with less than 30 available selected raters. As a matter of fact, it is generally observed from work studying the cost vs. the accuracy of results in machine learning when leveraging on crowd-sourcing that, high accuracy levels

[23]The evolution timeline was constructed in 50 snapshots representing the intermediary states of the G-dataset's graph between time 0 and the time at which the graph in the dataset was achieved.

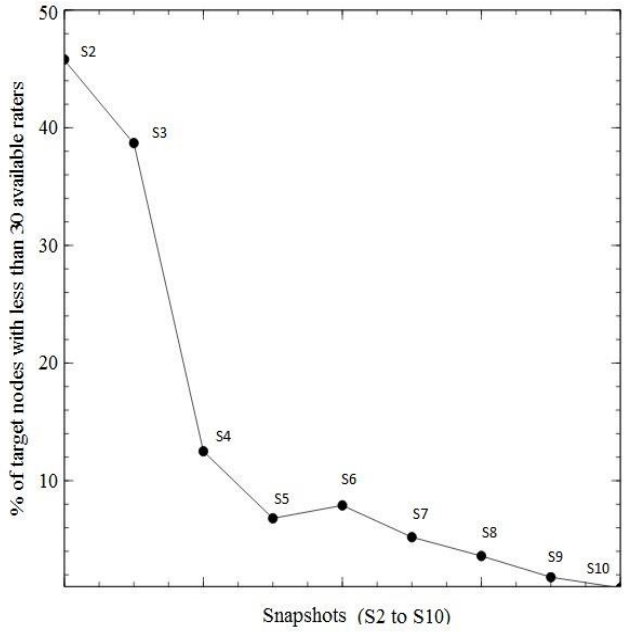


Figure 4.9 Percentage of users with less than 30 potential raters at every evolution snapshot.

and stability of results or convergence to consensus is empirically achieved with a number of workers (i.e., people performing the tasks) between 10 and 20 [84][88][127]. These numbers are observed for tasks performed by workers from the wide Internet community and within settings in which these workers get paid per answer with limited means to evaluating their credibility or work's quality [140]. As such, the accuracy or honesty of a worker, in such settings, makes the worst case scenario for our system in which raters are required to perform lighter tasks, they are selected from the pool of OSN users only, and they are expected to be more involved and motivated for their general good. However, we still run our study based on a minimum threshold of 30 raters as to account for the non-overlapping of raters that might be required by privacy preserving raters selection. We believe that assuring a minimum of 30 available raters per target user is large enough to safely claim that the raters' selection is feasible within the OSN.

We report the results in Figure 4.9, where we can clearly see that the percentage of users with less than 30 available selected raters is quite high in the first snapshot (45.8%). This is expected and understood as the first snapshot illustrates the very first instance of the existence of the OSN. However, this percentage quickly and drastically decreases especially starting from snapshot 4 (from 38.7% in snapshot 3 to 12.5% in snapshot 4). It then continues a general decline throughout the following snapshots until it reaches 0.88% only by snapshot 10. We consider this a good result as the few users below the 30 available raters can be easily addressed adopting one of the algorithms suggested for minimizing the number of

Table 4.9 Lowest number of available raters per snapshot.

Snapshot	S2	S3	S4	S5	S6	S7	S8	S9	S10
Lowest number of available raters	3	2	4	3	0	6	0	2	4

needed workers per task by creating a balance between number and quality, such as the work suggested in [84].

Another thing worth further analysis is the slight increase at the level of snapshot 6 compared to snapshot 5 (from 6.8% to 7.9%) that we can observe on Figure 4.9. This increase can only be explained by new nodes joining the network for which the pattern is not yet known or popular in the OSN (i.e., nodes with newly seen values for the *employer* attribute). To better understand this slight increase, we calculate, for each snapshot, the lowest number of available selected raters among all target users and we report the results in Table 4.9.

From Table 4.9 we can clearly see that, at every snapshot, there is at least a target node facing a low number of available selected raters. This number does not represent any trend and reflects the join of nodes with new values to the OSN. These might be the early seeds for the joining of a community (e.g., a person from Company A joins the network and then invites other colleagues to join), or anomalous nodes. In general, the nodes that are the first ones to join the network from their communities are always expected to experience this issue. Such a problem is commonly known in the domain of recommender systems as the cold start problem [86]. Number of works suggest workarounds and solutions to this problem, mostly by means of predicting and/or inferring the missing information based on social links, or by connecting to external information sources [81][149]. In our system, the cold start problem does not reflect lack of information about the node itself but lack of similar nodes to it in the network. An applicable solution is to confide those nodes to a group of experts in the OSN. This experts group might be the OSN's operators or a group of highly trusted and qualified users who can unlock cold start situations.

Finally, a thing worth mentioning in understanding and evaluating the results of this experiment is that the G-dataset corresponds to the period when Google+ was in its early days. That is, the G-dataset corresponds to the wider evolution phase of the network in which variance is expected to be the highest and hence our analysis is done on a worst case scenario basis.

4.3 Decentralized Identity Validation-DIVa Model

After designing an identity validation model (i.e., CbIV) that extracts correlations between profile attributes and deploys them to evaluate target profiles, our subsequent research

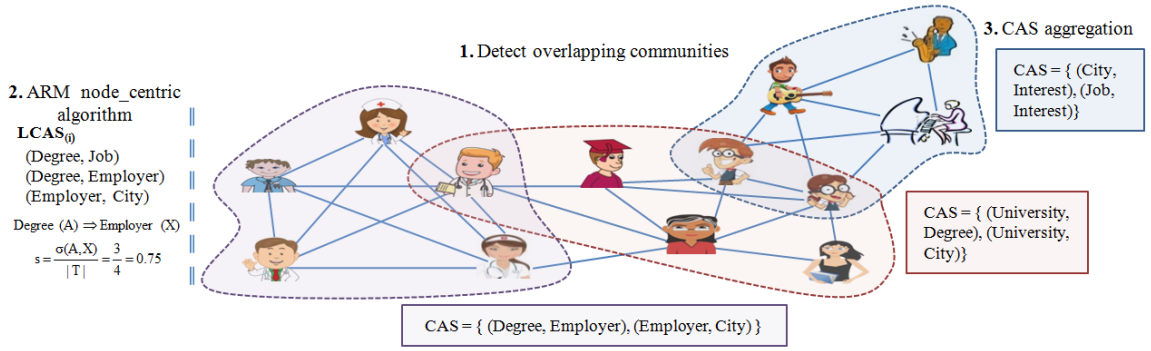


Figure 4.10 The three phases of the DIVa model.

concern was to redesign the system under a DOSN architecture. The motivations included: 1) benefiting from the power of decentralization is terms of increased performance and scalability; and 2) removing any potential privacy issues at the macro level by relying on local information at users level only.

We observed at the closing of Section 4.1 how the results of the experiments on the CbIV model unveiled a community dependence in the definition of the correlations between profile attribute groups. Starting from this observation, and based on the principle that OSNs exhibit a clustering phenomena as users topologically cluster into groups with intra-ties denser than inter-ties [78, 49], we designed a decentralized identity validation (DIVa) model.

In designing DIVa model, two characteristics of OSNs were based upon: (1) OSNs cluster into socially homogeneous communities that, (2) bind people with generally common identity trends and patterns [49]. Accordingly, DIVa operates in three phases (see Figure 4.10). Firstly, employing a state-of-the-art decentralized diffusion-based community detection algorithm ([111]), communities are defined such that every user (i.e., node) in the DOSN is aware of its community membership. Secondly, every node runs a node-centric algorithm to extract common trends of profile attribute correlations within its fiends closed circle. These correlations learned at a local level are referred to as local correlated attribute groups (LCAGs). Finally, all nodes within a community exchange their locally extracted knowledge without actually moving local data outside its ownership circle. The result of this knowledge sharing is modeled as a set of correlated attribute groups (CAGs) that are defined per each community in the network.

The resulting community specific CAGs can be used to evaluate the integrity of profile information of new nodes desiring to join a community. However, the focus of DIVa is on the extraction of those CAGs that can be then used in different ways for validation purposes, and not on the evaluation part itself.

We provide the details of each of DIVa's three phases in the following subsections.

4.3.1 Discovering Local CAGs

We base the distributed learning of the LCAGs on the association rule mining (ARM) that is generally used for the extraction of associations among different items in a shopping basket [2]. In our model, the items are the profile attributes and the association rules are the correlated attribute groups. For example, a node can learn that among her direct friends, users who are employed at company X also live in city Y. If a node observes that this is frequent enough in the profiles of its direct friends, the node deduces that attributes Employer and City are correlated. Before giving the formal definitions, we first introduce some notations.

We model an OSN as an undirected graph G = (V, E), where V is the set of nodes (or users) and E is the set of edges (or friendships), where $e_{ij} \in E$ denotes a relationship between nodes v_i and $v_j \in V$. We denote with S = {A_1, A_2, .., A_m }, the profile schema adopted in the OSN. Given a node $v_i \in V$, p_i denotes the set of its profile values: $p_i = \{p_i.a_1, p_i.a_2, ..p_i.a_m\}$, where $p_i.a_k$ is the value provided by v_i for $A_k \in S$.

In the decentralized learning of LCAG, every node stores the profile information of all its direct friends to form its *Local Profile Collection*. More precisely, given $v_i \in V$, we introduce the set $DF_i = \{v_j \in V | e_{ij} \in E\}$ as the set of v_i's direct friends, and $LPC_i = \{p_k | v_k \in DF_i\}$ as the collection of their profiles, referred to as v_i's local profile collection.

To minimize computational effort, a node v_i considers only those attributes having high value frequency in its LPC_i. As an example, a job value *musician* that is repeated in more than 30% of the profiles in LPC_i makes the attribute *Job* a frequent one; whereas, one satisfying less than this threshold is a non-frequent attribute. Therefore, we define:

Definition 4.3.1 *Local Frequent Attributes - LFA. Let $v_i \in V$ and LPC_i be its local profile collection. Let $A_k \in S$ be an attribute from the profile schema and let $P_k^{\vartheta} \subseteq LPC_i$ be the set of profiles in LPC_i having the same value ϑ for attribute A_k. That is, $\forall$ ϑ (ϑ is a given value), $P_k^{\vartheta} = \{p_m \in P_k^{\vartheta} \ | p_m.a_k = \vartheta\}$. Let $LFA_i \subseteq S$ be the set of attributes such that, $LFA_i = \{A_k \in LFA_i | \ \exists \vartheta \ s.t. \ \frac{|P_k^{\vartheta}|}{|LPC_i|} \geq \epsilon\}$, where ϵ is a global predefined threshold.*

Given LFA_i, v_i computes the support of each attributes pair in it. We limit to a pair as any combination larger than 2 can be decomposed into correlations between pairs. Assume CAG = (A,B,C). This implies there is a non-empty set of profiles where the values for A, B, and C co-occur. That is (A, B), (B, C), and (C, A) are correlated. The correlations (A, B), (B, C), and (C, A) are not enough for (A, B, C) to be correlated; however, the correlation (A, B, C) necessarily implicates the paired correlations.

We define the support of an attributes pair as follows:

Definition 4.3.2 *Support of an attributes pair. Let $v_i \in V$, LPC_i be its local profile collection, and LFA_i be its local frequent attributes set. Let (A_j, A_h) be a pair of attributes from LFA_i. The support of (A_j, A_h) defines the percentage of co-occurrence of the same paired values for the two attributes A_j and A_h to the total number of values in LPC_i:*

$$Support((A_j, A_h)) = \frac{values\text{-}co\text{-}occurrence(A_j,\ A_h)}{all\text{-}values(A_j,\ A_h,\ LPC_i)} \tag{4.1}$$

Where,

$values\text{-}co\text{-}occurrence(A_j,\ A_h)$=$|\{(p_e,p_m) \in LPC_i | p_e.a_j = p_m.a_j \wedge p_e.a_h = p_m.a_h\}|$.

and,

$all\text{-}values(A_j,\ A_h,\ LPC_i)$=$|\{\vartheta\ | \exists p \in LPC_i\ s.t.,\ p.a_j = \vartheta \vee p.a_h = \vartheta\}|$

By Equation 4.1, the support of an attributes pair is computed based on the values-co-occurrence of its elements to the total number of values in a local profile collection. For example, the values-co-occurrence between *Job* and *Education* is the number of profiles in which a specific pair of values, (Job = X, Education = Y), exist. However, the values of profile attributes in an OSN are mostly textual and are input as free-text by the users. Thus, two values might be syntactically different or worded differently but semantically the same. Therefore, computing the values-co-occurrence requires a technique for values comparison. We use for that a set of intersecting words comparison by which we compute the set of common words between two attributes' values. The size of this set reflects the similarity between these two values (see Algorithm 1).

Once the support has been calculated for all pairs in LFA_i, node v_i selects the ones for which the support is high enough to reflect they are correlated. Formally, we define a LCAG as:

Definition 4.3.3 *Local Correlated Attribute Group - LCAG. Let $v_i \in V$ and $LFA_i \subseteq S$ be its local frequent attributes set. Let (A_j, A_h) be a pair of attributes from LFA_i. The pair (A_j, A_h) is a local correlated attribute group, denoted as LCAG, if: $Support((A_j, A_h)) \geq \beta$, where β is a global predefined threshold.*

The threshold β can be set to whatever value that is most representative within the settings of the target network. That is, its value depends on the underlying characteristics of the OSN and on its nature. For example, an OSN targeting professional networking and focusing on professional profile information only might require higher values for the β variable. This is because the values co-occurrence in such a well scoped OSN are expected to be higher than in a general purpose OSN, for example. However, as a general guideline, we base the set up of the β threshold, as well as per the ϵ threshold as in Definition 4.3.1, based on the 20-to-80 rule, or what is commonly known in the statistics and economics literature as the Pareto rule [64]. The rule states that 80% of the outcomes come mostly from 20% of inputs only. This rule has been demonstrated by the Italian economist Pareto in his research and as such it was named after him. There are many economic conditions that demonstrate this rule, such as the distribution of wealth on earth that is roughly estimated as about 80% of the planet's resources being owned and/or controlled by only 20% of the population.

Besides, this rule holds also at different micro levels and is used in statistics and economics as the basis of number of theories and working real-life solutions. As such, we consider in our scenario that a correlation between attributes that is pronounced in at least 20% of the community population is reflecting an existing correlation between these attributes and is not only the result of chance or of some randomness in population distribution. That is, we set our thresholds, as will be mentioned in the experiments on DIVa to the value 0.2.

Algorithm 1 details how node v_i finds its LCAG list (i.e., $CLIST_i$). $CLIST_i$ is a list of attribute pairs (i.e., (A_1, A_2)) with their support contained in the variable $(A_1, A_2).c$. First, for every (A_1, A_2) from LFA_i, the pair is inserted in $CLIST_i$ with a support equal to 0 (line 3). Then, *tokenize()* retrieves all the words from all the values of A_1 and A_2 in all the profiles in LPC_i (lines 4 and 5) to form their respective word list, W_1 and W_2. Then, we calculate the number of pairs of profiles from LPC_i, such that their values for A_1 and A_2 share some words from their respective word lists and we update $(A_1, A_2).c$ accordingly (*getIndex()* locates the pair (A_1, A_2) in $CLIST_i$) (lines 6 - 12). Basically, for every two profiles from LPC_i, we get the set of words for which word lists of A_1 and A_2 intersect, denoted as X and Y respectively. Then, the normalized support of (A_1, A_2) is computed by dividing the length of minimum intersection between their word lists by all the words in W_1 and W_2 (line 10). The pairs in $CLIST_i$ with support higher than the predefined threshold β are the LCAGs of v_i.

Example 4.3.1 *Assume Jane is another OSN user belonging to two communities C_m and C_j. To learn her LCAG, Jane collects the available profile attributes from all her direct friends to construct LPC_{Jane}. Assume she finds that Job and City are in LFA_{Jane}; that is, their values are highly frequent in LPC_{Jane}. Jane computes the number of profile pairs in LPC_{Jane} for which these two attributes' pair have similar values. Assume in more than 40% of the profiles in LPC_{Jane}, this pair is co-occurring. Assuming that the LCAG threshold β = 0.3, the pair (Job, City) is an LCAG for Jane.*

4.3.2 Decentralized Community Detection

LCAGs learned locally at nodes need to be disseminated to construct community level CAGs. Thus, a community detection step is required. In general, a community is defined as a subgraph G' $\subset$ G = (V', E') representing a tightly-knit set of nodes that are sparsely connected to the rest of the nodes in G.

Many decentralized algorithms for community detection have been proposed [96, 111]. In particular, the work by Rahimian et al. [111] seems to comply with our decentralization requirement. In this work, every node starts as a community by itself using its node ID as its community ID. Then, every node chooses to quit its current community and join one of its neighbour's if this brings some modularity gains. Therefore, we define modularity gain in

Algorithm 1: LCAG Learning at node v_i

Input : LFA_i, LPC_i, and β
Output : List of LCAG with their support: $CLIST_i$

$CLIST_i = \emptyset$;
foreach $(A_1, A_2) \in LFA_i$ **do**
  $CLIST_i$ = `Insert(`(A_1, A_2)`, 0)`;
  W_1 = `Tokenize(`LPC_i, A_1`)`;
  W_2 = `Tokenize(`LPC_i, A_2`)`;
  foreach $p_k, p_j \in LPC_i$ **do**
    $X = \text{tokenize}(p_k.a_1) \cap \text{tokenize}(p_j.a_1)$;
    $Y = \text{tokenize}(p_k.a_2) \cap \text{tokenize}(p_j.a_2)$;
    if $X \neq \varnothing \wedge Y \neq \varnothing$ **then** $s = \frac{min(|X|,|Y|)}{|W_1|+|W_2|}$;
    $CLIST_i[getIndex(A_1, A_2)].c = CLIST_i[getIndex(A_1, A_2)].c + s$; ;
  end
end
foreach $(A_1, A_2) \in CLIST_i$ **do**
  if $CLIST_i[getIndex(A_1, A_2)].c < \beta$ **then** `Remove(`$CLIST_i$, (A_1, A_2)`)`; ;
end

terms of dominant ID by which every node changes its community ID to the dominant one in its neighbours. This step is iterated until no node wants to change its ID as it already represents the dominant one of all its neighbors. If a node does not find a dominant ID among its neighbours, it changes to the highest ID between its own and the ones of its neighbours.

However, since communities in social networks are majorly overlapping, and considering that DIVa aims to learn correlations that are more representative of a node's environment, we opt for a soft clustering approach. That is, a node can belong to more than one community and hence have more than one dominant community ID. More precisely, a node keeps track of the top dominant community IDs that it learns about when the community detection algorithm converges. As shown in Figure 4.11, DIVa instances have organized themselves into two overlapping communities. Further, node with the largest ID (i.e., node with darker shade) has been identified as the leader node of the detected community.

4.3.3 Community-level Aggregation

After step 2 of DIVa, every node becomes aware of its community ID(s). Furthermore, each community ID points to its leader node (i.e., its diva node) of the community; that is the node with the maximum nodeID. Particularly, this node is responsible for collecting all LCAGs found in its community to extract from them the community CAGs using a weighted voting mechanism. Thus, each node sends its LCAGs and their corresponding support values to its community's diva(s) and receives back CAGs with the maximum votes represented by maximum aggregated support. However, recall that in DOSNs nodes are aware only of their

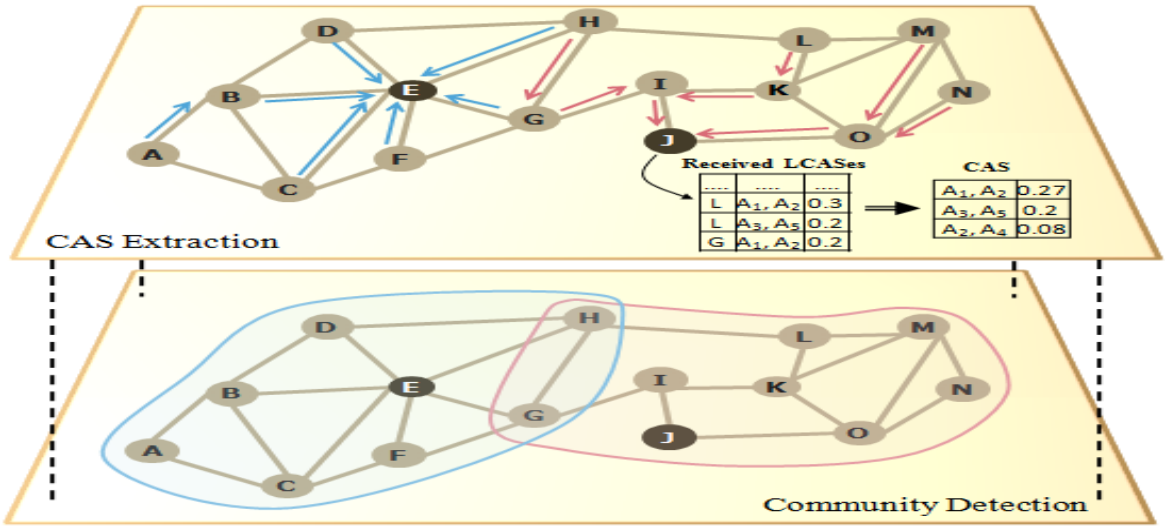


Figure 4.11 DIVa example: community level aggregation of CAGs.

direct friends. Therefore, nodes must be aware of the path to reach their communitie(s)' diva(s) by a hop-by-hop routing over their social ties. Accordingly, during the execution of the community detection part, nodes maintain paths leading to the diva of each community they belong to. Path construction is straightforward. First, a node checks if the diva is a direct friend. If it is not, the node creates the path by assembling the node IDs of intermediate nodes leading to the diva. Figure 4.11 depicts how nodes reach their communities diva nodes by following the constructed path towards them.

Given that all nodes know the path to each of their communitie(s)' diva node(s), messages containing their LCAG, along with the support of every LCAG attached as a parameter $LCAG.c$, are sent by every node to reach the destined diva(s) using a hop-by-hop consumption of the specified path. With every newly received message, each diva node inserts the ID of the sender into a *participants* list and the communicated LCAG into a list of received LCAG (i.e., $LIST_L$). A diva node is assumed to be preliminary aware of the size of its community after execution of community detection. When the number of received participants represents the majority of the community's nodes, the diva aggregates the support of received LCAG. The calculation of the aggregated support is straightforward: for each LCAG a diva node receives, it computes the average of its received supports from the different nodes communicating it. As shown in Figure 4.11, node J has received LCAGs form other nodes belonging to the community and has aggregated partial results to reach community CAG. The last step is to inform the rest of nodes with the final consensus in CAG representing the community.

Example 4.3.2 *Following up with Example 4.3.1, Jane communicates to the diva nodes of C_j and C_m her LCAG $(Job, City)$ with its computed support. Assuming that the diva of C_j (i.e., $diva_j$) receives this LCAG from the majority of nodes in C_j, $(Job, City)$ is a CAG in*

C_j. However, let us assume that in C_m, (Job,City) is under represented and hence it is not a CAG in C_m. Given this, Jane will have the choice of using the CAG (Job,City) to evaluate the trustworthiness of her new contacts based on which community, C_j or C_m, she knows them from. For example, a new contact who claims to be a colleague is more highly expected to meet this CAG, when another one who Jane knows from the GYM is less likely to be judged based on it.

4.3.4 Security and Complexity Analyses

Security Analysis

We analyze the security of DIVa assuming a malicious adversary model. The goal of an attacker would be to convince the community to use a corrupted CAG so as to confirm the trustworthiness of his/her identity or to compromise the ones of honest nodes. Since CAG values depend on the co-occurrence frequency of similar attribute values within a community, an attacker can have an effect only by introducing enough fake profiles (i.e., sybil nodes [147]) within the target community. Sybil nodes in OSNs have been studied both from a graph and a profile perspectives [23]. Since we detect communities based on topology, our focus will be on graph based sybil detection (GSD). Most work on GSD agrees that Sybils often manifest in proper topological structures making them distinguishable from honest nodes [23]. More precisely, sybil nodes exhibit proportionally smaller degree centrality and tend to group as outliers in the graph, compared to honest nodes [23]. Therefore, based on the logic of DIVa, sybil nodes would be detected in separate communities with their own CAGs unless they trick enough honest users into establishing friendship links with them so that they join their honest communities. This last possibility is discussed in the following sub-sections. Proofs to security theorems are presented in **Appendix B**.

Introducing a fake CAG

To introduce a new CAG, CAG_{new}, to a community, an attacker has to successfully integrate in it enough fake nodes, say z nodes, that carry profile information confirming CAG_{new}. Inserting a node x into a community C requires the creation of a number of edges (i.e., friendships) with enough other members of C so that x is included into it by the community detection algorithm. We emphasize on the amount of effort needed to introduce one new fake node into a community and we deterministically define z, the number of such fake nodes required to introduce a fake CAG_{new}.

Theorem 4.3.1 *Let $\overline{C} \subset G$, $\overline{C} = (\overline{C}.V, \overline{C}.E)$, be a community of size n ($|\overline{C}.V| = n$). Let sup_{lowest} be the lowest support by which a CAG is accepted in $\overline{C}$. For a new CAG, CAG_{new} to appear in $\overline{C}$, it must be inserted a group of fake nodes C_f that successfully join $\overline{C}$ and that show profile information confirming CAG_{new} such that:*

$$z = |C_f| \geq \frac{sup_{lowest}}{(1-sup_{lowest})} * n.$$

By Theorem 4.3.1, we can see that the required adversary effort to introduce a fake CAG is directly proportional to the size of the target community and to the lowest support by which it accepts a new CAG. Given that, communities tend to be more vulnerable to adversaries when they are small in size. However, for the adversary to succeed, fake nodes need to successfully establish enough links with the members of the target community. We believe that small communities, that are supposedly composed of nodes knowing and trusting each other, would tend to be more selective and hence more resilient to accepting unknowns compared to larger communities. In addition to that, a smaller community is expected to be more homogeneous (a small group of people knowing each other closely would be more homogeneous than a larger group) and so could allow high threshold values for the computations of its CAGs. By Theorem 4.3.1, it is clear that the higher the support threshold is, the higher the number of fake nodes are needed for an attack to succeed.

Corrupting a valid CAG

Additionally, an adversary might be interested in removing a valid CAG from a community. This can be achieved by introducing profiles that are not compliant with this CAG in a number big enough to disturb its valid support.

Theorem 4.3.2 *Let $\overline{C} \subset G$, $\overline{C} = (\overline{C}.V, \overline{C}.E)$, be a community of size n ($|\overline{C}.V| = n$). Let sup_{lowest} be the lowest support by which a CAG is accepted in $\overline{C}$. For a valid CAG, CAG_{valid} with support S_v, to disappear from $\overline{C}$, it must be inserted in $\overline{C}$ a group of fake nodes, C_f, that does not have profile information confirming CAG_{valid} such that:*

$$z = |C_f| > \frac{S_v * n}{sup_{lowest}} - n.$$

By Theorem 4.3.2, the vulnerability of a valid CAG is inversely proportional to the percentage of nodes in the community that contributed in making it arise. In fact, the lower the support of a CAG is, the smaller the number of nodes in the community represent it, and as such, the most vulnerable CAG to such an attack is the one having the lowest support. However, the success of such an attack, even for the weakest CAG, still requires the insertion of a considerable number of Sybil nodes in the target community. As we discussed before, this in itself remains a challenge for attackers.

Complexity Analysis

The model's cost is an aggregation of its steps. First, every node computes its LCAG. The complexity of this is a function of the number of node's friends (i.e., its degree d) and of the number of profile attributes in the profile schema. Indeed, the LCAG learning requires

computing for every pair of attributes (a profile schema of m attributes results in $p = \binom{m}{2} = \frac{m^2-m}{2}$ number of pairs), its value-co-occurrence among all the node's direct friends. Therefore, the number of performed checks per attributes pair is, $c = \binom{d}{2} = \frac{d^2-d}{2}$. Accordingly, the LCAG learning's complexity is $\mathcal{O}(c*p)$. By this, the nodes with higher degree would be the bottlenecks in the LCAG learning step; however, this step is node dependent and does not require the simultaneous online availability of all the nodes. Moreover, it is executed only upon significant changes to the network.

Regarding the community detection step, it costs in terms of communication traffic between all the nodes in the OSN. By our adopted work for decentralized community detection, this step's cost is $\mathcal{O}(N*D*R)$, where N is the total number of nodes in the OSN graph, D is the average node degree, and R is the total number of rounds needed for the algorithm to converge. R depends on the topological properties of the underlying graph [111]. This can be very costly for large graphs, but this step is a one time process only, as is in this proposed model.

4.3.5 Experiments and Results on DIVa

We have implemented DIVa using GraphLab [89] with two different distributed execution modules. The first module executes our adopted community detection algorithm until it converges so that every node knows the divas of its communities and the paths toward them. Thereafter, the control is moved to the second module that extracts CAGs for every detected community. We compare DIVa discovered CAGs with global CAGs generated by learning from all profiles in one central repository. That is, global CAGs are obtained by executing DIVa's LCAG learning algorithm, but over all the available profiles in the OSN all at once.

As aforementioned, the LCAG learning considers a node's LFA to prune the attributes passing the required values-frequency threshold ϵ (Definition 2.2). We set $\epsilon = 0.2$, as already discussed under Section 4.3.1.

Datasets Description

We conducted experiments to validate the effectiveness of DIVa using a real profile dataset from Facebook that contains 23332 nodes and 28972 edges. This Facebook dataset was collected and used in [5]. The profile schema in the dataset contains: First Name, Gender, Home County, Education, Job, Current Country, and Interests. The dataset was collected using a Facebook app that gathered the friendship links and profiles of the users who launched it. As a result, the collected data makes a graph of connected hubs with no overlapping communities.

Therefore, every node belongs to a single community, and communities divas are reached directly without any message routing overhead. The community detection algorithm converged after three rounds detecting 64 communities.

Table 4.10 Diva extracted CAG vs. Global CAG for Facebook dataset.

DIVa CAG Community1		DIVa CAG Community2		Global CAG	
Attribute Pair	**Sup.**	**Attribute Pair**	**Sup.**	**Attribute Pair**	**Sup.**
1:(gender, f.name)	0.14	1:(education, employer)	0.168	1:(ob, interest)	0.335
2:(f.name, h.country)	0.105	2:(employer, interest)	0.164	2:(gender, interest)	0.179
3:(education, job)	0.103	3:(job, h.country)	0.108	3:(education, interest)	0.138
4:(job, employer)	0.102	4:(gender, f.name)	0.105	4:(job, h.country)	0.137
5:(education, interest)	0.085	5:(f.name, h.country)	0.104	5:(education, job)	0.126

Achieved Results

Table 4.10 lists CAGs extracted by DIVa and their equivalent support values for two communities comparing them to the global extracted CAGs. For the two communities we list the top five attribute pairs in terms of support value. In fact, DIVa extracts 15 attribute pairs on average, but we show the top 5 due to space limitation. As a first observation, CAGs extracted by DIVa differ between the two communities. This proves that communities have different identity patterns and that DIVa succeeds in revealing them. This is further confirmed by the common CAGs in the two communities as their support values hugely differ in each one of them. For example, the pair (education, interest) has different support values 0.085, and 0.168 in Community1, and Community2, respectively. This emphasizes that the importance of an attribute pair differs from a community to another depending on its social and identity characteristics.

Our second observation is related to the nature of the CAGs discovered by DIVa compared to the ones retrieved from global learning, especially with regard to the type (single or multi-value) of attributes they contain. Indeed, the pairs extracted universally capture the common global trends and mainly consist of multi-value attributes. Meanwhile, most DIVa CAGs are made of single-value attributes. This is an important factor to consider and a real strength of DIVa as single-value attributes cover more specific identity dimensions that could not be captured from the global learning. Indeed, this latter failed to extract fine-grained and community-aware rules that disappear when considering the whole network as one giant component.

Figure 4.12 depicts a comparison of total support values of CAGs extracted by DIVa and those extracted globally. For every detected community we compute DIVa total support by summing up the support of its extracted CAGs. Meanwhile, the second total support (global) is the sum of the global support values associated with attribute pairs of global CAGs if those attribute pairs are correlated inside the detected communities. Figure 4.12 shows that DIVa provides stronger validation than the global approach, as the total support of DIVa generated CAGs is higher than the total support of global CAGs. In general, DIVa achieves average improvement over global CAGs that is up to 52.5% in the Facebook dataset.

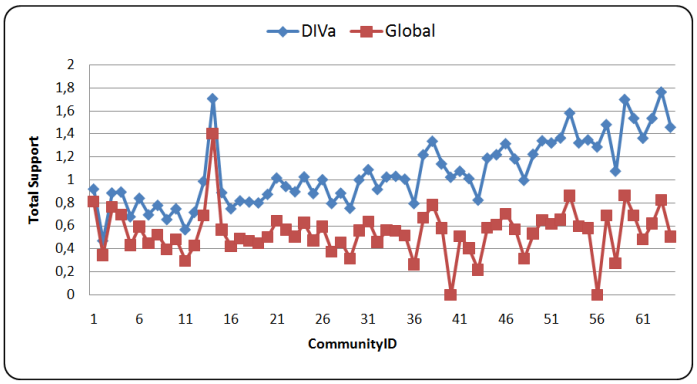


Figure 4.12 Comparing total support of DIVa CAG to globally extracted CAG for Facebook dataset.

4.3.6 Extended Notes

We must note that one of the limitations of DIVa is its static nature in the sense that it cannot adapt to changes in the network. Indeed, the community detection and the learning are done on a graph as a static instance; whereas graphs representing real OSNs environments are dynamic and continuously changing with new nodes joining, or other leaving the network. This suggests that both the community detection and the learning, both local and aggregate at the level of diva nodes, should be performed in a continuous manner adapting to the dynamics of the underlying graph evolution. Moreover, the system thus far relies on diva nodes within each community to perform the aggregations of community level CAGs and their dissemination to the other nodes in the network. This might be both a bottleneck, in terms of performance, and also a single point of failure.

To address these issues, we have extended the collaboration with the team at KTH to suggest a continuous and adaptive version of the DIVa model (CADIVa). As of the submission of this thesis, this work is under review for publication.

Chapter 5

New approaches to access control in social networking sites

As discussed under Chapter 3, Section 3.2.3, we have identified two main lines of action corresponding to two of the major challenges and limitations still facing the access control problem in social networking sites. These are, offering more *flexibility in specifying richer access policies*, and supporting more *efficiency and scalability in enforcing them* in social networking sites, especially under the decentralized architecture. Therefore, in this chapter, we dedicate our effort to the exploration of alternative access control methods that could be adopted to provide richer controls to the users on the one hand, and more efficiency and scalability in enforcing these controls, on the other hand.

The results of our research are two access control models for OSNs that we have defined, formalized, and tested.

For the first model, the main target issue was the efficiency and scalability of enforcing access control in DOSNs. The corresponding literature focuses mainly on reviewing cryptgraphic based enforcement mechanisms with the aim of offering better efficiency and scalability within the dynamic and fine-grain requirements of DOSNs (refer to Chapter 3, Section 3.2.2 for an amplified discussion on this). However, the natural rigidness of encryption mechanisms, as they are thus far, seems to be restrictive to practical progress along this path.

The other possible alternative is to rely on trust based mechanisms instead, as is already adopted under the centralized model of OSNs (see Chapter 3, Section 3.2.1). However, relying on trust requires the interposition of a central entity that manages, observes, and enforces the trust based controls. To solve this duality, we suggest to take a completely different approach; i.e., aposteriori access control enforcement. Therefore, our suggested model bases on a collaborative aposteriori control and report data sharing (CARDS), that bases on trust and that is post controlled by an auditing mechanism, to provide flexible, scalable, and accountably controlled data sharing in DOSNs. Objects under the CARDS model have associated access policies, formulated based on compound trust rules, that express the sharing

requirements imposed by their owners; however, no control is enforced apriori. The DOSN users are expected to share objects in alignment with their associated rules, and a recording mechanism is installed to detect those who do not. CARDS relies on trust, reputation, transparency, and accountability and deploys auditing techniques to ensure the cleanliness and correctness of the system. We further explain the motivation of CARDS and we present its details under **Section 5.1**.

Concerning our second target scenario, we explore a labeling strategy that would allow OSN users to express a richer set of privacy and access control preferences for their objects, at atomic ownership levels. Stepping aside from the relationship based access control model, traditionally adopted for social networking sites (see Chapter 3, Section 3.2), we exploit a mandatory access control (MAC) alternative that is designed in a user-centric fashion. This combination has already been explored in the literature, under what is known as label based access control (LBAC), to benefit from the formalism of data flow control provided by MAC all while allowing a flexibility in who defines the controls. We demonstrate how our suggested method provides a richer set of privacy settings and simplifies the problem of managing multiple access control to co-owned objects.

In our model, LAMPS (label-based access-control for more privacy settings), users of the social networking site are considered the ultimate owners and administrators of the information they generate or that is generated about them by other users (e.g., a tag). As such, a user is the central authority that defines the control labels to be associated with her/his data (objects) and friends (subjects) in the network. Access control is then enforced using a set of principles and properties that we carefully design following the types of interactions and the resulting information flows in social networks. The novelty of the LAMPS model is that it also allows the modeling of access authorization in terms of relationships between objects, allowing the specification and enforcement of individual privacy settings on uniquely owned *dependent* objects (e.g., likes, comments) that make part of an aggregate co-owned object (e.g., a photo annotated with likes, comments, tags, etc.). Therefore, the proposed LAMPS model would offer more flexibility and a richer privacy preferences repository for OSN users, compared to what is currently enabled by the relationship based model adopted by major social networking sites, such as Facebook. We details LAMPS under **Section 5.2**.

Chapter's List of Abbreviations

CARDS	Collaborative Aposteriori control and Report Data Sharing
AC	Access Control
MAC	Mandatory Access Control
LBAC	Label based Access Control
LAMPS	Label-based Access-control for More Privacy Settings
FCL	Friend Clearance Label
FSP	Fundamental Security Property
SHP	Share Higher Property
WHP	White Higher Property
OSL	Object Sensitivity Label
SR	Share and Report
LR	Listen and React
IP	Inform and Prevent
ME	Monitor and Evaluate
TReMa	Trusted Register Manager

5.1 Audit-based Aposteriori Access Control - CARDS Model

As we have discussed in Chapter 3, the access control problem has been essentially approached with the goal of keeping sensitive resources securely locked in, by ensuring that only authorized entities can unlock their way to them. This approach ensures the protection of data as much as the locks are stronger than the will of intruders, and fails short when the shields are broken. Moreover, it works under the assumption that all the approved accesses are preliminary known and static, and thus, they can be implicitly or explicitly coded into a set of authorizations.

In today's web-based collaborative environments, that are characterized by dynamicity, denseness, and diversity in data creation, sharing, and ownership, the traditional view to access control may not be enough. Indeed, information security advocates have already started calling for an alternative approach to data protection, that should be based on accountability, operated by transparency, and regulated by adequate and enforced laws and systems [142][114].

DOSNs, with their decentralized architecture, and with the personal nature and granularity of data shared in them, make one of the scenarios where the standard a-priori access control paradigm better shows its limits. The huge user population, the fact that access to personal or sensitive information is not always under the control of the data owner but can be influenced by the actions of other users in the network (e.g., a user tagging another user in a picture), and the absence of a centralized repository for data management and observance are all factors that make the formulation and the enforcement of access authorization more complex than it has ever been.

With the aim of finding more flexible alternatives to the rigidness of cryptography based solutions (see Chapter 3, Section 3.2.2 for more details on that), and with a strong belief in the power of accountability and transparency in ensuring better informed data security management and in increasing awareness towards privacy issues, we studied the potential of deploying a deterrent a-posteriori access control mechanism for DOSNs. Instead of deploying hard preventive mechanisms, we suggest an open sharing environment based on collaboration and on trust, where an auditing mechanism coupled with a reputation management system is put in place to encourage good behavior and to deter bad actions. Moreover, a proactive activity continuously takes place to prevent continuity of damage in cases of detected bad actions.

5.1.1 The underlying Framework

The CARDS model assumes that each object in the system is solely owned by the user that created it (i.e., the owner), and is under the protection of any user that gets access to it (i.e., custodians). Owners specify access requirements for their objects, and the system needs to ensure the objects are disseminated in alignment with these requirements. The CARDS

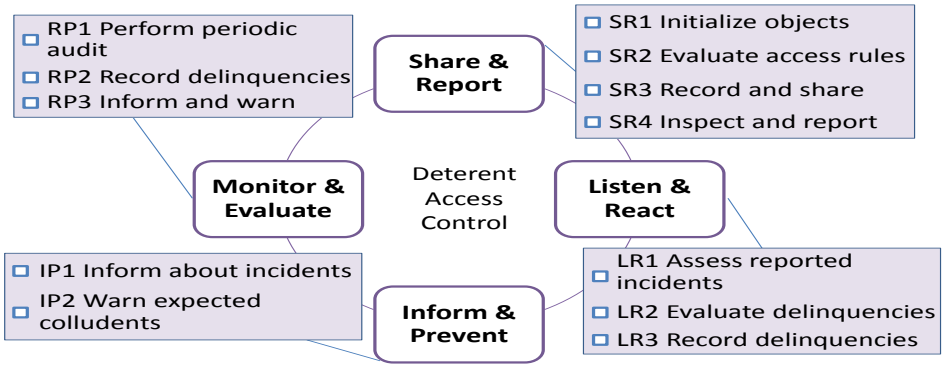


Figure 5.1 The four steps of the proposed CARDS framework

model is developed based on an a-posteriori paradigm that follows a proposed framework of four connected processes, as depicted in Figure 5.1. These processes are described as follows:

- **Share & Report:** this process aims at recording data sharing transactions. Data owners specify access rules for their objects by activity SR1 and objects are shared based on evaluating their access rules (activity SR2). Custodians might share an object against its access rule, thus activity SR3 records any sharing transaction taking place. Finally, activity SR4 reports illegitimate sharing transactions and can be performed by custodians, in case they have access to the object's sharing log, or by any entity safekeeping it.

- **Listen & React:** with three activities, LR1, LR2, and LR3, this process concerns the reaction to illegitimate sharing (i.e., *delinquencies*) communicated using activity SR4. It is recommended to have LR1 and LR2 taken by different entities than the one performing SR4, since such a separation of tasks would ensure dual control that provides stronger security guarantees. In fact, dual control requires actions from more than one entity to grant access, basing on the premise that for a breach to happen, all entities need to collude. It is also recommended that these entities do not share a common interest so as to avoid collusion [135].

- **Inform & Prevent:** this process aims at deterring suspected entities from performing further illegitimate sharing of a victimized object. The main challenge is the timeliness to have a preventive action and also the ability to predict future behavior that is to be prevented.

- **Monitor & Evaluate:** the motive of this process is to ensure the cleanliness of the environment under the assumption that not all entities will willingly abide by the reporting mechanism of the first process. This is ensured by three activities that should be carried out periodically on all the nodes in the system.

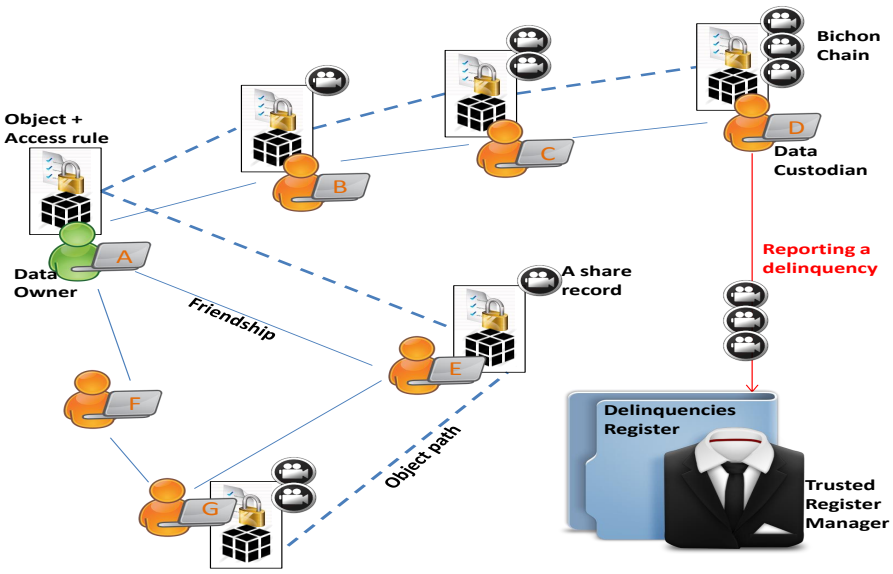


Figure 5.2 The CARDS architecture for a decentralized social network with a trusted monitor

Based on these processes, we model the CARDS model by considering the DOSN users as its major active players. As exemplified on Figure 5.2, DOSN users establish friendship links with each other such that they are aware of and can only contact their direct friends, create objects, and share them with their fellows based on their privacy preferences.

As we have discussed in Chapter 3, privacy preferences in OSNs and DOSNs are generally defined following a relationship-based model [50] in the sense that users gain access to information based on the links they establish with its owners. Similarly, in designing CARDS, we assume a rule-based model for the formulation of relationship-based privacy settings for an object. That is, upon creating an object, its *owner* assigns to it a relationship-based access rule encoding its privacy requirements and then shares the object, in plain format, with direct eligible friends as allowed by the rule. Users that receive an object (i.e., *custodians*) can share it further with their contacts. To allow for activity SR3 from the CARDS framework (Figure 5.1), we attach to every object a log that traces its *sharing trajectory* (*Bichon Chain* as on Figure 5.2). A *Bicon Chain* keeps record of every sharing transaction that the object has been subject to through a given path since leaving its owner. The first sharing record is added by the owner when first releasing the object, then records are added to the chain by every custodian who shares the object. The chain should allow an object's receiver to check: 1) whether the sharing transaction by which it received the object is legitimate or is a *delinquency*, and 2) which are the valid sharing transactions that it can still perform on the object.

When a delinquency is detected, the node needs to report it and appropriate actions against the delinquent node should be taken. To manage delinquencies, we suggest the exploitation

of a central trusted register manager (i.e., *TReMa*) that manages a central *delinquencies register*. The *TReMa* listens to all the nodes in the system, receives reported susceptible Bichon chains, evaluates them, and records in the *delinquencies register* any delinquency they contain (Figure 5.2). The *delinquencies register* contains records of confirmed delinquencies as a pair *[actor, delinquency's severity]* (refer to Algorithm 3 for the severity computation). The *TReMa* also performs the activities in the *Inform& Prevent* and *Monitor& Evaluate* processes from the CARDS framework (cfr. Figure 5.1), to ensure the sanity of the system and to catch unreported collusion, if any. Since we target collaborative DOSNs, we limit the activities of the *TReMa* to managing the reported delinquencies and to performing periodic audits, ensuring that it does not get hold of the objects themselves. In this work, the trustworthiness of the *TReMa* is assumed.

The CARDS framework is to be deployed with a reputation management system that uses the recorded delinquencies in computing entities' reputation. This system has also to provide a punish/reward mechanism that provides incentives for good actions and deters from bad ones and has to be aligned with the architecture and the requirements of the system instantiating the CARDS framework. Typically, a reputation management system should provide a mechanism for the collection of information and computation of reputation scores, a mechanism for the dissemination of these scores, and security guarantees against manipulation of scores. Reputation management is a due discipline in itself that is being subject to a number of research work [73] with proposals for both decentralized [42] and centralized solutions [6]. As such, we consider, in this work, that a reputation management system is put in place keeping its study outside of the model's scope.

5.1.2 The CARDS Model Formalization

In this section, we detail our proposed CARDS model for DOSNs. We start by defining its basic building blocks.

Basic Definitions

We note that we differentiate between the reference monitor of a node, represented by the software, and the end-user (the person) who manipulates it. Hereafter, we refer to the software as the *node* and to the person using it as the *end-user*. We formally define a DOSN as follows:

Definition 5.1.1 *DOSN. A DOSN is a directed graph,*
G=(V, E, R, T), where:

- *V is the set of nodes (or users), where each node $v_i \in V$ has a unique network identity (denoted $v_i.id$), a digital identity signature expressed by the ownership of an identity key-pair (idk_i, $sidk_i$), and a reputation score denoted as $v_i.rep$.*

- *E is the set of relationship edges such that $e^r_{ij} \in E$ denotes a relationship of type $r \in R$ from node $v_i \in V$ to node $v_j \in V$.*
- *R is the set of available relationship types in the DOSN. Example of relationship types can be: partner, colleague, family.*
- *T is a function that assigns to each edge $e \in E$ a trust value, denoted by e.trust, (e.trust $\in$ [0, 1]).*

The *TReMa* is the only entity in the system aware of all the available nodes' identity signatures and it is the only entity that can verify them. The identity key-pair can be implemented using any cryptographic signature scheme.

The reputation of a node is public in the DOSN and can be computed based on the number and severity of delinquencies recorded against it in the *delinquencies register* and in alignment with the adopted reputation management system. Given that the *TReMa* is responsible of computing, safekeeping, and disseminating reputation scores, a centralized reputation system could be deployed. However, and for performance purposes, the dissemination of reputation scores should also be enabled at a peer to peer level such that a node could retrieve this information from its neighbors, if they have it locally, instead of contacting the *TReMa*. Solutions such as those suggested in [42][6] could be considered. We plan tackling this issue in more details in future extensions of this work.

Regarding edges' trust values, and given they will be used for evaluating access to personal data, we consider them to be assigned by the users involved in the direct relationship. Users might consider the reputation of a node when assigning their trust values to their links with it. Finally, and for computational purposes, we consider that every relationship type in R corresponds to an integer in $[1, \ldots, |R|]$.

Nodes generate content in the DOSN in the form of objects and are considered their owners. An object can be of different types depending on what is supported in the DOSN. Examples of object types are *text*, *picture*, *video*, etc.

Owners set the privacy requirements for their objects and encode them into access rules. We formally define an object as follows:

Definition 5.1.2 *Object. Let $v_i \in V$ be a node in the DOSN. An object owned by v_i is denoted by the tuple, $Ob^i_k = (OID, t, val, ow, AcsR)$, where:*

- *$OID = v_i.id\#t\#k$ is a unique ID in the DOSN referring to the object Ob^i_k, where k is a sequence number computed locally at node v_i.*
- *t is an integer indicating the type of the object.*
- *val is the bit-value of the object.*
- *$ow = v_i.id$ is the owner of the object.*

- *AcsR is the access rule encoding the privacy requirements of the object.*

For access rules modeling, we adopt the model in [29]. By this model, every resource is linked to an access rule that expresses conditions on the type, the depth (distance between owner and requestor) and the trust level that a relationship needs to satisfy so that its members are eligible to access the resource. That is, given an object Ob, its access rule (denoted as $Ob.AcsR$) is a list of access conditions among which at least one has to be met for a legitimate access to the object. An access condition is denoted by $acsc = (ow, RType, MaxD, MinT)$, where ow refers to the owner of the object $RType$, $MaxD$, and $MinT$ respectively refer to the type, the maximum distance, and the minimum trust of the relationship between the owner and a legitimate receiver. In what follows, we use the dot notation to denote an object components.

In a DOSN, no node would be able to trace paths of distance higher than 1. For that, we need to consider an indirect trust value for indirect relationships. The literature offers several ways for computing indirect trust values [87]. In our model, and since the access rules refer to the existence of specific paths, we only consider the path in question for the computation of indirect trust. That is, given the trust values, $e_{ij}.trust$ and $e_{jk}.trust$, between directly connected nodes v_i and v_j, and v_j and v_k, the trust value of the path $p_{ijk} = (e_{ij}, e_{jk})$ is expressed by: $p_{ijk}.trust = e_{ij}.trust * e_{jk}.trust$.

Objects are shared between nodes exploiting the edges established between them. To log an object's sharing activity, a *Bichon chain* is attached to the object since it first leaves its owner. A Bichon chain is a list of chained sharing records, each signed by the node performing it. A sharing record is defined as follows:

Definition 5.1.3 *A sharing record. Let $v_o, v_i, v_{i+1} \in V$ be three consecutively connected nodes in the DOSN via relation types $r, r_1 \in R$ respectively, i.e., $e_o = e^r_{oi} \in E$ and $e_1 = e^{r_1}_{i(i+1)} \in E$. Let Ob^o be a resource owned by v_o. The sharing records of object Ob^o are generated as follows:*

- *When node v_o shares Ob^o with its direct neighbor v_i using the edge e_o, it creates the initialization tuple $ShR_0 =< Ob^o.OID, tr, dist, ty >$ s.t. $Ob^o.OID$ is the object's ID, $tr = e_o.trust$, $dist = 1$, and $ty = r$. ShR_0 is called the reference sharing record.*
- *The sharing record of Ob^o from v_i to v_{i+1} is the tuple: $ShR_1 =< Ob^o.OID, tr_1, dist_1, ty_1 >$ s.t. $tr_1 = ShR_0.tr * e_1.trust$, $dist_1 = ShR_0.dist + 1$, and $ty_1 = r_1$.*

As by Definition 5.1.3, a sharing record is created, by the sharing node, for every passing of the object from one node to another. It contains aggregate information on the path traversed by the object so far and it is meant to publish the criteria based on which the sharing happened.

The sharing records of an object are all chained to make its sharing trajectory comprised in the *Bichon chain*. This chain is a list of blocks (i.e., *Bichon rings*) each containing a

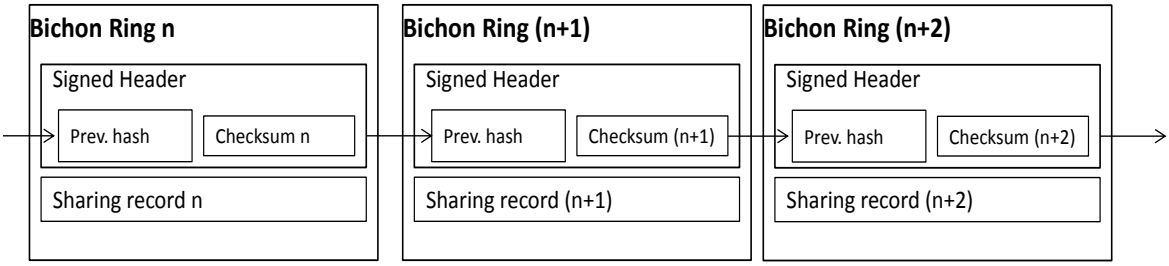


Figure 5.3 The structure of a Bichon chain for an object's share trajectory

sharing record for a given object's trajectory and a corresponding *Bichon header*. This header serves for the verification of both the correctness and the integrity of the chain, and for the identification of the block's creator by the *TReMa* when performing the audit. We formalize a *Bichon header* as follows:

Definition 5.1.4 *A Bichon header. Let* $v_o, v_i, v_{i+1} \in V$ *be three consecutively connected nodes in the DOSN and let* $ShR_0 =< Ob^o.ID, tr, dist, ty >$ *and* $ShR_1 =< Ob^o.ID, tr_1, dist_1, ty_1 >$ *be the sharing records of object* Ob^o*, owned by node* v_o*, from* v_o *to* v_i *and from* v_i *to* v_{i+1}*.*

- *The reference Bichon header for* ShR_0 *is denoted by the signed element:* $header_0 = sign_{sidk_o}(checksum_0)$*; where* $sign_{sidk_o}(X)$ *is a function that signs message* X *with the identity secret-key* $sidk_o$ *of node* v_o *and* $checksum_0 = tr + dist + ty$*.*
- *The Bichon header for* ShR_1 *is:* $header_1 = sign_{sidk_i}(concat(checksum_1, H(header_0)))$ *where* $sidk_i$ *is the identity secret-key of node* v_i*,* $concat(f, g)$ *is the concatenation of strings* f *and* g*, and* $H()$ *is a hash function.* $checksum_1 = tr_1 + dist_1 + ty_1$*.*

Every Bichon ring contains one sharing record plus its corresponding Bichon header. As per Definition 5.1.4, every header comprises a hash of the previous Bichon ring's header. This has the effect of creating a chain of sharing records making the trajectory of the traced object. Sharing records (rings) are guaranteed to be chronologically ordered in the Bichon chain because the previous Bichon ring's header would otherwise not be known. The Bichon rings can be verified for integrity using the checksum value included in the signed header, and the signed headers serve for accountability recording and for audit tracking by the *TReMa*. Figure 5.3 depicts a simplified sketch of a Bichon chain. Formally, a Bichon chain is defined as follows:

Definition 5.1.5 *Bichon chain. The Bichon chain of an object Ob, denoted as BC_Ob, is a list of BC_Ob.length Bichon rings: BC_Ob* $= \{ring_i \mid ring_i = [header_i, ShR_i],\ i \in [0, BC\_Ob.length]\}$*, where* ShR_i *is a sharing record and* $header_i$ *is its corresponding Bichon header.* $ring_0 = [header_0, ShR_0]$ *is the reference ring of BC_Ob.*

Sharing and Reporting

In this section, we detail the sharing and reporting processes.

Sharing objects

When a node receives an object, it also receives its corresponding Bichon chain. Upon viewing the object, the end-user might choose to share it with one of its friends. At such event, the node uses the object's access rule along with the last Bichon ring in the Bichon chain to evaluate the legitimacy of the intended sharing transaction. We clarify that a legitimate sharing path for an object must abide by the object's access rule, and hence is required to contain edges of the same type only. Let us start by defining the following:

Definition 5.1.6 *Legitimate sharing record. Let* Ob^o *be an object and let* $ShR_n = < Ob^o.ID, tr_n, dist_n, ty_n >$ *be the sharing record in the n-th Bichon ring of its Bicon chain. Given the access rule of the object, encoded as a list of access conditions,* $Ob^o.AcsR = \{acsc \mid acsc = (Ob^o.ow, RType, MaxD, MinT)\}$*, the sharing record* ShR_n *is legitimate if and only if:* $\exists\ acsc_h = (ow, RType_h, MaxD_h, MinT_h) \in Ob^o.AcsR$*, such that:* $tr_n \geq MinT_h \wedge dist_n \leq MaxD_h \wedge ty_n = RType_h$.

The evaluation of the legitimacy of a sharing transaction is performed by Algorithm 2, that takes as input the access rule ($Ob.AcsR$), the last sharing record in the Bichon chain of the target object (ShR_n), and the trust and the type of the edge along which the sharing is to be evaluated ($e^r.trust$ and r). The algorithm computes the sharing record to be evaluated by making a call to the *CreateShR* function (line 1). Then, it evaluates its legitimacy by the *MatchRule* function (line 2). If the sharing record is legitimate, the algorithm returns *Legitimate* (line 4); otherwise it returns a *Delinquency* message (line 7).

Once the node has the answer to the legitimacy of the intended sharing, it prompts the end-user for the last decision. If the end-user chooses to share the object, regardless of the legitimacy of the transaction, the node creates the Bichon ring corresponding to this sharing and appends it to the object's Bichon chain. Afterwards, the node initiates a secure communication channel with the target node (i.e., the next receiver of the object). There are two possible scenarios, that are exemplified by Example 5.1.1: 1) the sharing transaction is legitimate, or 2) the sharing transaction is a delinquency.

Example 5.1.1 *Let us assume Kate and Jane are DOSN friends and that the relationship edge from Kate to Jane has type colleague and trust value 1. Assume that Kate receives a video*

Algorithm 2: Evaluation of the legitimacy of a sharing

Input : The access rule of the object to be shared: $Ob.AcsR$
Input : The last sharing record in the Bichon chain of the object to be shared: $ShR_n =< Ob.ID, tr_n, dist_n, ty_n >$
Input : The edge along which the object is to be shared: e^r
Output : Message *legitimate* or *delinquency*.

```
1  new_ShR = CreateShR(e^r.trust, r, ShR_n);
2  match = MatchRule(new_ShR, Ob.AcsR);
3  if match then
4  |   return Legitimate;
5  else
6  |   return Delinquency;
7  end
8  Function CreateShR(e^r.trust, r, ShR_n)
9  |   SR_tr = ShR_n.tr_n * e^r.trust;
10 |   SR_ty = r;
11 |   SR_dist = ShR_n.dist_n + 1;
12 |   SR = < Ob.ID, SR_tr, SR_dist, SR_ty >;
13 |   return SR;
14 end
15 Function MatchRule(new_ShR, Ob.AcsR)
16 |   foreach acsc in Ob.AcsR do
17 |   |   if new_ShR.ty = acsc.RType ∧ new_ShR.tr ≥ acsc.MinT ∧ new_ShR.dist ≤ acsc.MaxD
   |   |   then return True;
18 |   end
19 |   return False;
20 end
```

(denoted as V) for which an access condition allows sharing it with colleague, with minimum trust= 0, and maximum distance= 10. Assume that the value of the distance parameter in the last ring of the video's Bichon chain is 5 and that Kate wants to send the video to Jane. Figure 5.4(a) depicts the communication flow between Kate's and Jane's nodes to perform this sharing. First, the Bichon chain of the video, BC_V, is updated at Kate's node to add the ring for the sharing transaction to be performed and it is then encapsulated, with the video's access rule (V.AcsR), in a newShare() request to Jane's node (see Figure 5.4(a)). The received Bichon chain is verified at Jane's node by extracting the last ring's sharing record from the Bichon chain and evaluating it using function MatchRule from Algorithm 2. In this case, the sharing is legitimate, so V is accepted and the transaction is fulfilled.

Assume now that Kate received a photo (denoted as P) owned by her cousin Ryan, with whom she is friend on the DOSN with a family relationship type. Ryan wants the photo to be viewed by family members only. Kate decides to share the photo with Jane though this sharing is a delinquency (Kate does not have a family relationship with Jane). Figure 5.4(b) presents the communication flow between Kate's and Jane's nodes for this delinquent sharing. This starts by Kate's node sending a newShare() request to Jane's node encapsulating the picture's Bichon chain (BC_P) and its access rule (P.AcsR). The last ring in the chain is verified at Jane's node and is found to be delinquent. Jane's node prompts Jane to take the

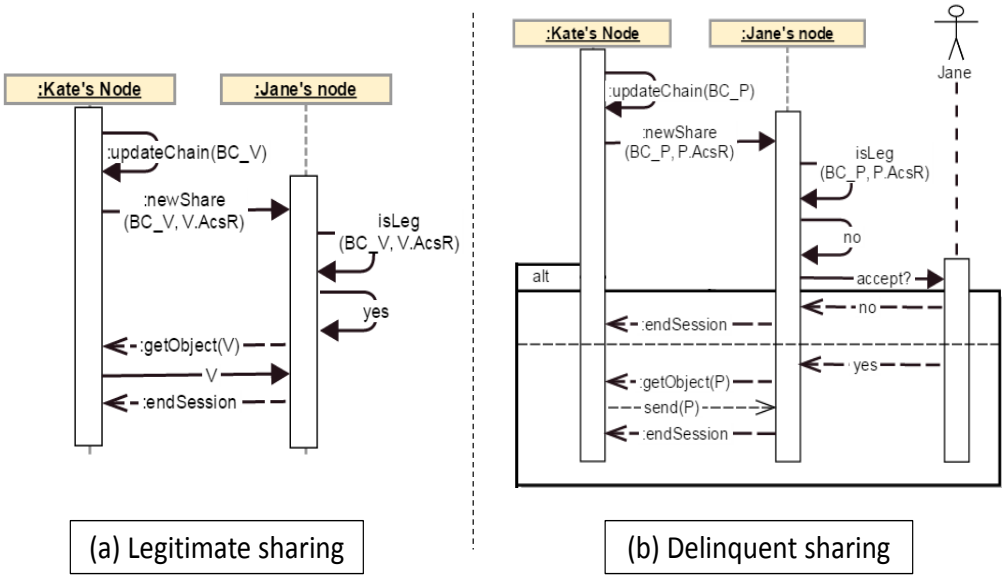


Figure 5.4 Communication flow for a legitimate and a delinquent sharing

decision. If Jane accepts, the picture is sent and the operation is terminated. If she rejects, the operation terminates without transferring the picture.

Reporting a detected delinquency

When a node receives a request for a delinquent sharing, it can report it to the *TReMa*, regardless of whether it accepted receiving the object or not. In fact, even in the event of accepting a delinquent sharing, the receiver might want to redeem for having colluded in the delinquency by reporting it. This would be an illustration of a honest-but-curious user. To report the detected delinquency, the node needs to send to the *TReMa* the Bichon chain in question along with the concerned object's access rule. To demonstrate a reporting operation, we continue with the scenario in Example 5.1.1. In fact, when Jane's node finds that the sharing request for object P is a delinquency, it prompts Jane for whether it has to report it to the *TReMa* or not. If Jane chooses to report it, a *report delinquency message - rdm* is sent to the *TReMa*. A report delinquency message is defined as follows:

Definition 5.1.7 *Report delinquency message (rdm). Let $v_i \in V$ be a node in the DOSN, Ob be an object, and BC_Ob its Bichon chain. An rdm on BC_Ob communicated by node v_i to the TReMA is defined as follows:*

$$rdm = sign_{sidk_i}(Ob.AcsR, BC\_Ob); \text{ where } sidk_i \text{ is the identity secret-key of node } v_i.$$

The TReMa's Operations

The *TReMa* reacts to reported delinquencies by executing the processes *Listen& React* and *Inform& Prevent* (cfr. Figure 5.1). It verifies the reported message, records the confirmed delinquencies, informs the faulty nodes, and runs targeted auditing on other nodes that could have possibly received the victim object. Before detailing the *TReMa*'s operations, we first define a delinquency record.

Definition 5.1.8 *Delinquency record (dr). A delinquency record (dr) is a pair [actorID, severity], where actorID = v.id ($v \in V$) and severity* $\in \mathbb{N}^*$.

Listen and React

The *TReMa* evaluates a received *rdm* from node v_i by running Algorithm 3. The algorithm takes as input the message *rdm* and the identity-key idk_i of the reporting node v_i. It initializes an empty *log* list (its output parameter) and checks the validity of *rdm* by verifying its signature (by calling function *CheckSignature* in line 1). If the message is valid, function *CheckChainCorrectness* is called (line 3) to verify the integrity of the reported chain by checking the conformity of each ring's header to its corresponding sharing record (via the checksum value) and to the hash of the previous ring's header, as from Definition 5.1.4 (lines 13-22). If the verification of the chain or of the *rdm*'s signature fails, the algorithm returns a null value. Otherwise, a call to the *AuditChain* function is made (line 5). This latter evaluates the legitimacy of the rings of the target chain (i.e., *BC_Ob*) starting from the last one (lines 24-36). It then continues evaluating the previous rings as long as delinquencies are detected (lines 26-32). To evaluate a ring, its enclosed sharing record is extracted (line 27) and it is checked for legitimacy using function *MatchRule* from Algorithm 2 (line 28). If the sharing record is not legitimate, the actor of that sharing transaction is extracted from the corresponding ring's header using function *GetSignerID* (line 30) and a delinquency record is added to the log with a severity level equal to 1, by using function *RecordDelinq* (line 31). When the algorithm finds a legitimate ring in the chain, it stops its evaluation as rings previous to it are also legitimate (the break statement at line 33). If the number of delinquent rings is more than 1 (line 35), the severity levels of the logged delinquency records need to be adjusted. This is what is achieved by invoking function *AdjustSeverityLevels* at line 35. This function increases the severity level of each delinquency record based on its position in the series of delinquent rings (lines 42-46). That is, the actor of the first delinquent sharing is punished with a severity equal to 1, the second colludent with a severity of 2, and so on. This adjustment ensures that nodes that collude by receiving a delinquent sharing without reporting it, get a higher punishment than the first initiator of the collusion. Finally, the algorithm returns the list of found delinquency records.

If the reported delinquency is a false alarm (i.e., the output Log is empty), the *TReMa* notifies the reporter node v_i and logs the false alarm. Moreover, a log of all received

notifications with the result of their evaluation is kept by the *TReMa*. This information can be used to reward the nodes reporting valid delinquencies by increasing their reputation.

Inform and Prevent

After recording a delinquency, the *TReMa* informs the object's owner about the incident. The owner can use this information to reconsider the assigmnent of access rules to its future shared objects. Besides, it also informs all the nodes against whom a delinquency record has been logged, as returned by Algorithm 3. Afterwards, the *TReMa* takes a protective procedure to prevent eventual delinquencies to happen with the same object by starting a targeted auditing on all these nodes's direct contacts (next potential colludents). It then continues auditing on any other of their contacts' contacts found to have received the object until the sharing of the object is found to be stopped.

Algorithm 4 presents the steps to perform the targeted auditing based on an identified object. The algorithm takes as input the log of delinquency records found for an object (i.e., the output of Algorithm 3) and the id of the object in question ($Ob.OID$). Using these two parameters, it goes through all the delinquency records, *dr* in the input list. For each *dr*, it retrieves all the friends of its actor node (line 3). For each of these friends, it checks whether they have received the target object from the actor node or not (line 5). If a friend is found to have received the object from the actor node, a delinquency record is created against it using function *RecordDelinq* from Algorithm 3 by passing to it a severity input parameter incremented by 1 compared to the severity by which the previous *dr* has been recorded (line 6). This new recorded delinquency is appended to the initial input *colludents* list so that the friends of its actor are audited as well (line 10). Finally, the algorithm returns the log of all new found delinquency records.

The aim of the targeted auditing on an identified object is to track all other possible colludents after the first delinquency report received on it. It is admitted that the prevention is not strong as it does not guarantee full protection of the victimized object. However, it works on minimizing the risk of it being subject to further mis-shares by notifying possible future colludents. Given that the system is based on a reputation spirit, we believe that warning nodes and/or punishing them through detecting and recording their delinquencies would have a deterrence preventive outcome.

Monitor and Evaluate

Besides reacting to reported delinquencies, the *TReMa* takes proactive actions by executing the activities under the *Monitor& Evaluate* process (cfr. Figure 5.1). Periodically, the *TReMa* starts general auditing on three different groups of nodes, of fixed sizes, selected randomly and uniformly from their respective pools. The first group is selected from the pool of nodes that have no delinquency record registered yet. The second one is selected from nodes having

Algorithm 3: Processing an *rdm* by the *TReMa*

Input : reported delinquency message from node v_i: $rdm = sign_{sidk_i}(Ob.AcsR, BC\_Ob)$.
Input : identity-key of node v_i: idk_i.
Output: A list of delinquency records for delinquencies in the chain: *Log*.

```
1  Log = ∅; validMsg = CheckSignature(rdm, idk_i);
2  if validMsg then
3  |  validChain = CheckChainCorrectness(BC_Ob);
4  |  if validChain then
5  |  |  Log = AuditChain(BC_Ob, Ob.AcsR);
6  |  |  return Log;
7  |  else
8  |  |  return null;
9  |  end
10 else
11 |  return null;
12 end
13 Function CheckChainCorrectness(BC_Ob)
14 |  foreach ring_i in BC_Ob do
15 |  |  header_i = UnsignMsg(ring_i.header_i);
16 |  |  if is ring_0 then
17 |  |  |  if header_0.checksum_0 is not equal ShR_0.tr + ShR_0.dist + ShR_0.ty then return
   |  |  |  false;
18 |  |  else
19 |  |  |  if header_i is not equal concatenate(header_i.checksum_i, H(header_{i-1})) then
   |  |  |  return false;
20 |  |  end
21 |  end
22 |  return true;
23 end
24 Function AuditChain(BC_Ob, Ob.AcsR)
25 |  Log = ∅; pos = 0; len = BC_Ob.length - 1;
26 |  for ring_len in BC_Ob do
27 |  |  ShR = ShR_len;
28 |  |  legit = MatchRule(ShR, Ob.AcsR);
29 |  |  if not legit then
30 |  |  actor = GetSignerID(header_len);
31 |  |  RecordDelinq(Log, pos, actor, 1);
32 |  |  pos = pos + 1; len = len - 1;
33 |  |  else break;
34 |  end
35 |  if pos ≥ 2 then Log = AdjustSeverityLevels(Log, pos) ;
36 |  return Log;
37 end
38 Function RecordDelinq(Log, n, actor, severity)
39 |  dr = [actor, severity]; Log.Add(dr, n);
40 |  return Log;
41 end
42 Function AdjustSeverityLevels(Log, pos)
43 |  foreach dr_i in Log (i starts from 0) do
44 |  |  dr_i.severity = pos; pos = pos - 1;
45 |  end
46 |  return Log;
47 end
```

Algorithm 4: Targeted auditing based on identified objects

Input : list of delinquency records of colluding nodes on mis-sharing an object: $colludents = \{dr_1, .., dr_n\}$.
Input : the target object id: $Ob.OID$.
Output : A list of delinquency records for delinquencies detected on the object: Log.

```
1  Log = ∅; pos = 0;
2  foreach dr in colludents do
3  |   targets = GetFriendsOf(dr.actor);
4  |   foreach friend in targets do
5  |   |   if HasReceivedObject(Ob.OID) then
6  |   |   |   RecordDelinq(Log, pos, friend, dr.severity+1);
7  |   |   |   pos = pos + 1;
8  |   |   end
9  |   end
10 |   colludents = colludents ∪ Log;
11 end
12 return Log;
```

at least one record in the delinquencies register. As for the third group, it is selected from the nodes for which the number, the severity level, and the frequency of recorded delinquencies is the highest since the last performed periodic audit. The rationale behind running periodic auditing on these three groups is to ensure the cleanness of the DOSN assuming that not all the nodes will abide by the reporting mechanism. For the first group, the aim is to double check that there are no malicious nodes hidden uncaught among good ones; whereas targeting the second group aims at checking how suspicious nodes behaved since the last recorded delinquency against them. As for the third group, the aim is to insist on deterring them considering that they do not seem to be compliant with the system yet.

A general periodic audit is performed by requesting all the Bichon chains in custody of the target nodes. Each of these chains is then verified, as by function *AuditChain* from Algorithm 3, to check and record any delinquencies it contains. We rely on the underlying software to ensure the communication of all the Bichon chains in a nodes' custody. However, if the node is broken through and the *TReMa* cannot get hold of its Bichon chains, it immediately informs all the network about the invalid node.

5.1.3 Security and Complexity Analyses

In this section, we provide and prove the model's security properties and we discuss the complexity of the system. We must mention that we assume that all the nodes abide by the process of creating Bichon rings for every share they make, as this is enforced by the software.

Security Properties

We consider a malicious adversary model with three possible attacks. First, malicious nodes can collude to propagate a delinquent sharing and never report it to the *TReMa*. Second, a malicious node can create fake valid Bichon chains to compromise the reputation of other honest nodes. Finally, a malicious node can alter valid Bichon chains either by inserting new fake rings or by changing the information already contained in the chains.

Let $v_m \in V$ be a malicious node with the aim of propagating an illegitimate sharing on an object *Ob*. Let S be the number of nodes randomly selected from V for the *TReMa*'s periodic general audit. CARDS satisfies three security properties:

- **P1:** v_m cannot remain uncaught given T general audit cycles, with T following a geometric distribution of $p = \frac{S}{|V|}$.
- **P2:** Fake but valid Bichon chains cannot be created.
- **P3:** Compromising the integrity of a valid Bichon chain will result in an invalid chain.

For property **P1**, let us assume an object *Ob* which has known an illegitimate sharing made by node v_m in collusion with the receiving node v_{m+1}. There are two possibilities for this collusion to be discovered: 1) a subsequent node, v_{m+2}, is a honest node that reports to the *TReMa*, or 2) v_m or v_{m+1} is selected for a periodic general audit. Let us assume the worst case by canceling the first possibility and study the probability that the collusion will be detected from a general audit. Assuming that the *TReMa* selects S nodes per general audit cycle, we get that $\forall v \in V, Ps(v) = \frac{S}{|V|}$ and $PNs(v) = 1 - Ps(v)$ with $Ps(v)$ being the probability that node v is selected in an audit cycle and $PNs(v)$ the probability that it is not. Clearly, the event of a node being selected in an audited cycle is a Bernoulli trial. Now, let T be a random variable referring to the number of audit cycles needed before a specific node is selected for the audit. Since the audit cycles are independent events and given that T supports values in $\mathbb{N}*$, T follows a geometric distribution with probability of success $p = Ps(v)$. Therefore, the expected mean of T is $\mu_T = \frac{1}{Ps(v)} = \frac{|V|}{S}$ [108]. This suggests that $\forall v \in V$, there exists a value of T during which v is selected for the general audit. Moreover, this value for T can be deterministically upper bounded by setting appropriate values for S. More precisely, for T to be upper bounded by k, S can be set to $S = \frac{|V|}{k}$ so that $\mu_T = k$; thus, having T upper bounded by k.

Regarding **P2**, let us assume that a malicious node wants to create a fake Bichon chain. Since Bichon rings contain identity signed Bichon headers, the node can only make fake rings with its identity signature or with bogus ones. For the first case, this is not logical as no node would fake a ring against its own reputation; whereas for the second case, the bogus signature will not be known by the *TReMa* and hence the chain will not pass the validity test of function *CheckChainCorrectness* (cfr. Algorithm 3).

Now consider that a malicious node wants to modify the data in a valid Bichon chain. The node can change the values in the chain's sharing records but will not have the means to accordingly update their corresponding Bichon headers as these latters are identity signed by their creators. This means that a valid Bichon chain for which the integrity has been compromised will never be considered valid by the *TReMa* as it will be evaluated as invalid by function *CheckChainCorrectness* that will detect disagreement between the sharing records and their Bichon headers in the compromised rings. Hence, given that the nodes that created all the rings in the chain do not all collude, property **P3** holds. If every and all nodes who signed a Bichon header in a Bichon chain collude, then **P3** will not hold; however, this is not realistic as, at least the signer of the reference Bichon ring (i.e., the object owner) will logically not collude.

Complexity Analysis

For Algorithm 2, its complexity is $\mathcal{O}(\textit{Rule-size})$, with $\textit{Rule-Size} = |Ob.LAcsR|$ being the number of the conditions in the longest object access rule in the system. Given that access rules are set by users to express their privacy requirements on an object, the length of a rule is logically not expected to be big.

For Algorithm 3, its complexity is $\mathcal{O}(|V| * \textit{LBC\_Ob.length})$, where *LBC_Ob.length* is the length of the longest Bichon chain in the system, and $|V|$ is the total number of nodes in the DOSN. *LBC_Ob.length* is determined by the maximum hops an object has to go through before it reaches all the nodes. Given the principle of six-degrees of separation, which has been demonstrated by experimental results to exist in today's online social networks with even a shorter scale (i.e., four-degrees) [40], *LBC_Ob.length* can be estimated to be less than 6. Concerning the $|V|$ parameter, it comes from the fact that the algorithm performs a search for users' identity keys, to validate rings, among all available keys. This might be considered costly, but can also be easily addressed by adopting a distributed hierarchical paradigm in designing the *TReMa*, similar to the architecture of the DNS (Domain Name System) protocol, for example. We further discuss this under the following section where we also present results on the scalability and performance of Algorithm 3.

Finally, Algorithm 4's complexity is $\mathcal{O}(\textit{LBC\_Ob.length} \times |C|)$, where $|C|$ is the number of detected colludents by the targeted audit.

5.1.4 Experiments and Results on CARDS

As the *TReMa*'s operations might seem quite demanding (analysis of Algorithms 3 and 4), we perform experiments simulating these operations to study the scalability of these two algorithms.

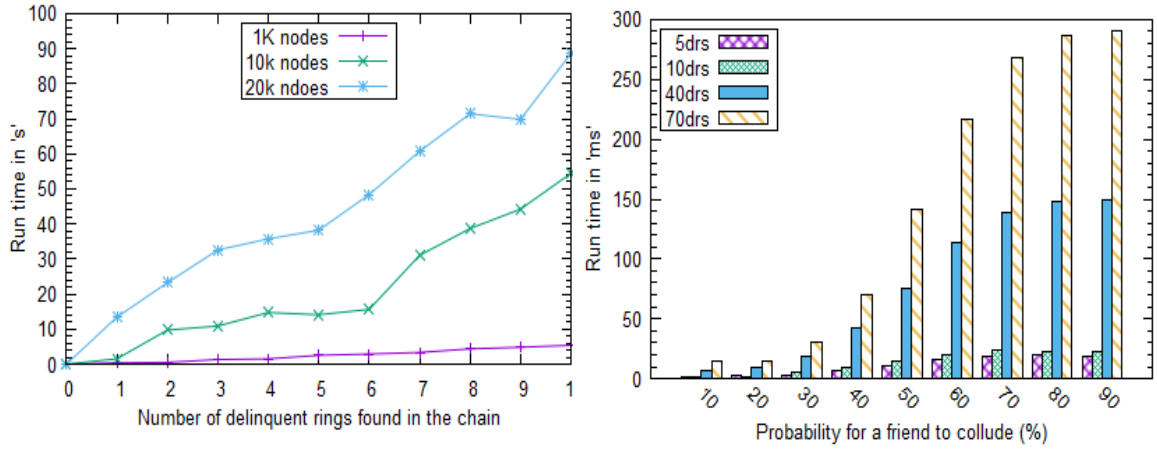
We run our experiments using the OSN's Pokec public dataset. Pokec is a Slovakian OSN and the related dataset is available at the Stanford datasets library the could be retrieved from: http://snap.stanford.edu/data/soc-pokec.html.

The dataset comprises 1.63 M nodes and 30.6 M edges. Its diameter, longest shortest path, is equal to 11, and so we set the maximum length of a Bichon chain to 11. We implemented both Algorithms 3 and 4 using the Java language. All the experiments have been conducted on a dual core PC of 3.4 GHZ each and 4 MB of RAM.

Scalability of Algorithm 3

Algorithm 3 processes a reported Bichon chain against the access rule of its object. We have simulated this reporting scenario by creating 10 different objects (i.e., 10 different access rules) and 10 Bichon chains, each of length 11, per object. For a given object, each of the 10 Bichon chains contains from 0 (false alarm) to 10 (max possible delinquencies in the chain) delinquent rings. We run Algorithm 3 20 times for each of the simulated objects against its 10 simulated Bichon chains. Since the algorithm performs a linear search for a ring's signer ID, in the case of a delinquent ring, we have set the pool of IDs over which this search is performed to 1K nodes, 10K nodes, and 20K nodes for three independent executions using the same settings w.r.t Bichon chains and objects. The sizes of the pools of IDs have been set based on the assumption of having a distributed *TReMa* architecture by blocks of communities. In fact, based on analyses of famous OSN datasets in the literature, we find that the average community size is in the order of hundreds of nodes [144]. As such, we run our simulations under the assumption of one *TReMa* node for every block of ten, hundred, and two hundred communities, respectively.

Figure 5.5a presents the results of this experiment. The x-axis refers to the number of found delinquencies in the reported chain, whereas the y-axis refers to the average run time in seconds of the algorithm. The figure depicts three lines each corresponding to one of the set pool sizes for the search on signer IDs. The first thing we read on the figure is that the higher the size of the pool is, the higher the run time is. This goes in line with the complexity of the algorithm (i.e., $\mathcal{O}(|V| * BC\_Ob.length)$). $|V|$ here represents the pool size. The second thing we can notice is that the higher the number of delinquencies is, the higher is the run time (the lines follow a linear sloped function). This is also expected as the search operation is performed per every delinquency found. However, the run time with the number of delinquencies under 5 (50% of the chain) is very low (35s as the highest recorded point for a pool of 20K nodes). Finally, even under the worst case of all the chains being delinquent (which is also not realistic) and with a pool size of 20K nodes, the algorithm converges in less than 90 seconds on a standard PC.



(a) Run time (in *s*) of processing an *rdm* by the *TReMA* Algorithm

(b) Run time (in *ms*) of targeted auditing by the *TReMA* Algorithm

Figure 5.5 Performance of CARDS Algorithms run by the *TReMA*

Number of audited nodes and scalability of Algorithm 4

Algorithm 4 takes a list of delinquency records (*drs*) found w.r.t a target object and performs an audit on the friends of their actors and subsequently on their friends as long as they are found to have received the object in question. The length of the input *drs* list is expected to be the same as the number of delinquent rings found in the reported chain that fired Algorithm 4. For experimental purposes, we have considered 4 scenarios with this length being of 5, 10, 40, and 70 drs, respectively. 5 drs corresponds to 50% of the reported chain being delinquent, and 10 drs to 100% of it. The other 40 drs and 70 drs have been set to stretch the system and study it's performance under more exhaustive conditions. Since the algorithm performs the targeted audit on a nodes' contacts as long as they are found to have received the object, we have simulated this by considering a varying probability for this to happen. That is, when a node is audited, a biased coin is flipped first. Depending on the flip result, the node is considered as has received the object or not. The bias in the coin is represented by a probability for the node to have received the object. We vary this probability from 10% to 90%. We simulate the algorithm for 20 times and report the average run time results in Figure 5.5b.

From Figure 5.5b, we can notice that the run time of the algorithm is in terms of milliseconds only and that the worst recorded time of 291ms is achieved with 70 initial drs (value used just to stretch the simulation) and a probability for friends to collude of 90%. We can also read on the same figure that the run time of the algorithm remains under 20ms for 5 drs and 10 drs with the probability of a friend to collude being lower than 60%. We think that this is a very good result as the worst case scenario for Algorithm 4 is when all the

Table 5.1 Number of audited nodes by Algorithm 4

	5 drs	**10 drs**	**40 drs**	**70 drs**
1 hop($90 \geq PCol \geq 60$)	79	192	1360	1441
2 hops($60 \geq PCol \geq 20$)	143	326	2316	2595
3 hops($20 \geq PCol \geq 10$)	7	18	82	134
4 hops($PCol \leq 60$)	0	0	1	3

reported chain is delinquent (i.e., 10 drs) and a non majority of colludents (i.e., probability to collude is less than 50%).

In addition to studying the performance of Algorithm 4 in terms of its running time, we also computed the number of audited nodes. Since the algorithm keeps auditing all the contacts of a node as long as they have received the object, we run our simulation by considering a probability of collision ($PCol$) of $90 \geq PCol \geq 60$ for the first hop friends, $60 \geq PCol \geq 20$ for the second hop, $20 \geq PCol \geq 10$ for the third hop, and $PCol \leq 60$ for the fourth hop. Table 5.1 presents the results on the number of audited friends by hop and by considering the same number of drs as before (5drs, 10drs, 40 drs, and 70drs). As we can read on the table, the number of audited nodes converges to 0 by the fourth hop. Besides, this number scores 326 audited nodes as the highest value with 10 initial drs. Recalling that the maximum initial drs the algorithm can have as input is 10 (the other values used to stretch the system), our experiment shows that the worst case scenario results in 326 audited nodes only.

5.2 Label-based Access Control - LAMPS Model

Privacy settings in current OSNs do not allow users to have control over all types of their generated data. In fact, users do not have the ability to set controls over likes or comments, for example, that they generate as an interaction on their friends' objects. We refer to such types of interaction generated data as *dependent objects*. Moreover, the related literature addresses this issue from a multiple data ownership perspective, by which aggregate objects (i.e., a parent object with its dependent objects such as comments, likes, etc.) are treated as one element over which multiple access policies could be specified, but only one would be enforced [131, 67, 133, 68]. This makes it complex for users to manage their privacy and does not guarantee the enforcement of their preferred policies over the specific data elements they have contributed with, hence that they uniquely own (see Chapter 3, Section 3.2.1 for more details on related work). The LAMPS model offers an access control mechanism that considers relationships and dependencies between these data elements that make aggregate co-owned objects. The purpose is to offer more flexible and richer privacy settings to OSN users. This is achieved by adapting label-based access control, LBAC (see Chapter 3, Section 3.1 for more details on LBAC) to the scenario and requirements of OSNs.

To better understand the limitations our work intends to address, we start by getting more insight on current OSNs common functionality and available privacy settings.

5.2.1 Privacy Settings and Common Functionality in Current OSNs

Typically, general-purpose OSNs would allow users to establish different types of relationships with their contacts and share different types of information. There are number of general-purpose OSNs in the online market nowadays, but it remains safe to say that the two worldwide biggest ones currently in the box are Facebook and Google+.

In these OSNs, users are associated with an information space typically made of two main blocks: a profile and a wall or timeline. The profile contains personal information, like full name, contact address, gender, age, education etc; whereas the wall organizes, in a graphically timelined manner, the data objects the user shared with her friends or those that her friends have shared with her.

Table 5.2 Types of information shared in OSNs.

Type	**Dependency**
Text (TX) [text posts & profile data]	Independent
Photo (P)	Independent
Video (V)	Independent
Like (L)	Dependent
Comment (C)	Dependent
Tag (TG)	Dependent
Geo-Location (GL)	Dependent
Friend-Post (FP)	Independent

Users can upload content of different types into their information space, or interact on their friends' shared information by liking it, commenting on it, tagging one or more other of their friends in it, etc. Such interactions result in the creation of other data elements that are related to each other, and that might also be owned by different users. Figure 5.6 illustrates an example of a published post, the data elements resulting from interactions on it, and the relationships between them.

Clearly, some types of information shared on OSNs can stand as *independent* by itself, while other types are always *dependent* on another information. For example, a comment cannot exist unless attached to a previously published post. Table 5.2 lists the different types of information that users can share and manage in nowadays OSNs specifying whether they can be independent or not. We note that the type termed as *Friend-post (FP)*, refers to an object posted on a user's wall by one of her friends.

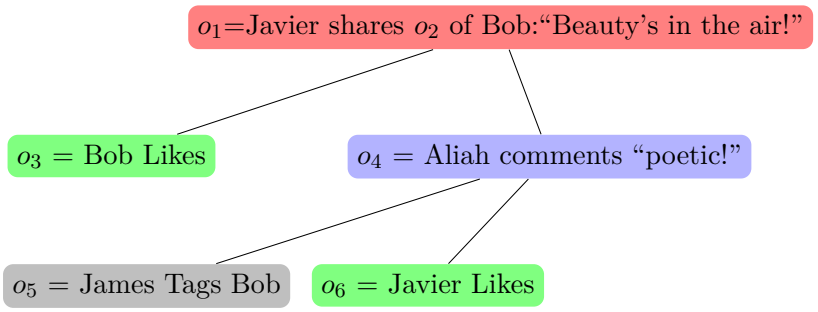


Figure 5.6 Typical objects' structure in OSNs.

Table 5.3 Users interactions in current OSNs and their associated possible privacy settings.

Privilege	**Privacy settings**
Read	Manageable for all types except C and L
Add-Comment	No available privacy settings
Add-Like	No available privacy settings
Add-Tag	Manageable both for who can add a tag and who can view added tags
Share	The audience specified by the original owner is automatically enforced on the shared copy
Write	Manageable at limiting who can post on one's wall and on who can see these posts but as an on/off setting by friend

Privacy Preferences Expression Tools and Related Limitations

In addition to publishing content, OSN users can also specify their privacy preferences, through a privacy settings interface, regarding who can access what information. Generally, a user can group her friends into groups and put some interaction limitations on some groups (e.g., cannot share content with me, or cannot tag me in photos). Additionally, at the moment of publishing a new *independent* information, the user can choose the friends or the groups to whom this content is addressed.

Without loss of generality, Table 5.3 provides a summary of the actions users can perform on OSNs along with the available privacy settings, as currently available either in Google+ or in Facebook. We note that the *write* privilege refers to a user posting on her friends' walls and not to writing on one's own wall.

As we can read on Table 5.3, the *read* privilege can be customized only for independent types of information, such as text, photo, etc. Dependent information, however, with the exception of photo tags, is automatically available to whoever has a *read* privilege on the information they depend on. This might be a real limitation, as a user might want to customize who can see her comments on others' posts. Moreover, users can only control the *read*, and *add-tag* privileges, whereas all the other privileges are available to whoever got a *read* access to an information. For example, if a user allows a friend X to read a post Y, he cannot prevent X from commenting on Y. However, users might need not only to control the

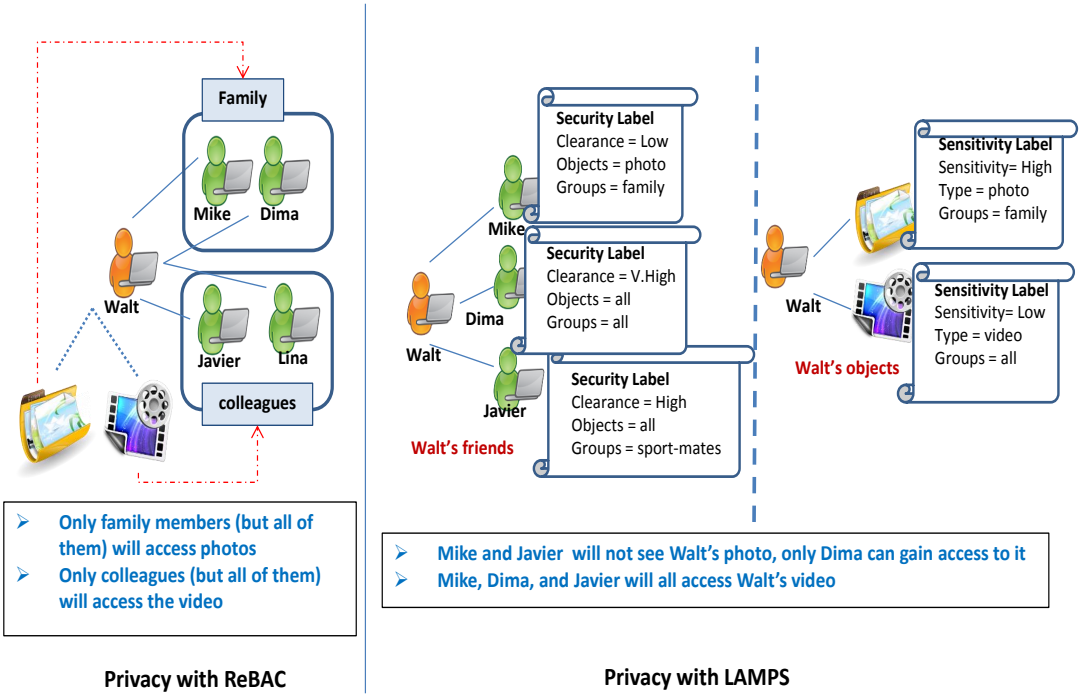


Figure 5.7 Privacy management with ReBAC vs. LAMPS

visibility of their information but also the interactions on it. In addition to this, users cannot control the *share* privilege. More interestingly, as on Facebook for instance, any friend who got a *read* privilege on a post can *share* it; however, this shared copy will only be available to the initial audience allowed by the original owner of the post. This might constitute a major limitation to the functionality of the *share* action in itself. For example, I might make a post available to only three of my friends, but I might also want them to share it with their trusted friends all while ensuring that non else of my friends will have access to it. Finally, users can control who can create content in their information space, but only in a binary manner. That is, it is either that a friend is allowed to post on a user's wall or not. However, users might want to control the levels of visibility of this content published by others in their space, instead of such basic on/off settings.

Moreover, the current tools available for privacy preferences expression are restricted to the notion of access groups. For example, assume a user Walt desires to apply similar privacy settings on his family members, Mike and Dima, and different controls on his colleagues Javier and Lina. The available option is to group Mike and Dima under a *family* list and Javier and Lina under a *colleagues* list, and apply the desired controls on these respective lists. At such an instance, all the information Walt shares with the *family* list will be equally accessible by both Mike and Dima and the same goes for information shared with the *colleagues* list (see Figure 5.7(a)). However, Walt might not always want to share his information based on this categorization. For example, if he wants to share an item only with Mike and Javier then the

only available way is to create a new list. In addition to this clear inflexibility, both Mike and Dima, for instance, will enjoy the same interaction rights on Walt's objects specified for the *family* list. That is, both of them can equally comment on, like, or share a photo that Walt made available for the *family* list. However, it might be that Walt does not want one of them to be able to comment on the photo, for example.

To overcome these limitations, we exploit in LAMPS a label-based access control strategy. As we have seen in Chapter 3, label-based access control is a more flexible adaptation of mandatory access control (MAC) that allows the establishment of security access orders between subjects and objects by means of labels that can be assigned by object owners.

Considering every user as the ultimate administrator of her/his data objects, with LAMPS, users assign *sensitivity labels* to their objects, and *security labels* to their friends. To account for the different aspects based on which users might choose their privacy settings w.r.t. their friends and to their data objects, we design the labels considering additional relevant features, other than the security and sensitivity levels as it is in standard MAC. More precisely, labels in LAMPS are expressed in terms of security or sensitivity levels, types of relationships, and types of objects. Access decisions are taken following a set of properties that we carefully define based on the specificity and types of activities taking place in current OSNs.

As a basic illustration, Figure 5.7(b) depicts the security and sensitivity labels that Walt assigns to each of his friends and of his objects, respectively. As it can be seen from the figure, access to an object is granted only to those friends having an equal or higher label. However, the richness of interaction types and access privileges that OSNs allow (e.g., tag, share, comment, etc) makes the picture more complex and requires careful design of different properties to account for all possible privilege request scenarios. This is what we dutifully explain, detail, and formalize in this work.

We formally describe the model in what follows.

5.2.2 The LAMPS Model Formalization

We start by presenting basic definitions, then we define how privacy requirements of users can be expressed using labels. Finally, we introduce the mechanism governing access decisions.

Basic Definitions

Without loss of generality, we model an OSN as an undirected graph $FG = (U, FR)$, where U is the set of users, and FR is the set of edges. We say that two users $a, b \in U$ are friends in the OSN if and only if there exists one direct edge connecting them, that is, $\exists e_{a,b} \in FR$.

OSN users create and share different types of information (see Table 5.2) leveraging on the relationships they establish with others. We define the set of information types in the OSN as $OTS = \{TX, P, V, L, C, TG, GL, FP\}$, and we refer to all pieces of information as an *object* that we formally define as follows:

Definition 5.2.1 *OSN object. An OSN object o is represented by a tuple (type, owner, parent, children, copyof), where: (We use the dot notation to refer to the parameters in an object tuple)*

(i) $type \in OTS$ *denotes the type of the object.*

(ii) $owner \in U$ *denotes the owner of the object. If* $o.type \notin \{TG, FP\}$*, the owner is the user who generated the object (dependent or independent). Otherwise, it is the user who is tagged or the user on whose wall the friend post is made.*

(iii) parent: if o is a dependent object, this refers to the object on which it depends. It is set to null, otherwise.

(iv) children is the set of objects that depend on o, as generated by interactions on it, if any. It is set to null, otherwise.

(v) copyof: if o is a shared copy of another object $\overline{o}$*, this is equal to* $\overline{o}$*, otherwise it is set to null.*

In addition to the objects that users can create, we consider the existence of another special object type, that we refer to as the *root* object, and that models user walls. We consider that every OSN user owns exactly one *root* object that is created by default upon her subscription to the OSN. Root objects serve for controlling the function of *writing* on users' walls by their friends as we will detail later (see Algorithm 5). A *root* object associated with user a is modelled as $root_a$ = *(root, a, null, null, null).*

Example 5.2.1 *Consider the OSN objects in Figure 5.6. By Definition 5.2.1, object* o_1 *is modeled as (TX, Javier, null,* $\{o_3, o_4\}$*,* o_2*), object* o_4 *as (C, Aliah,* o_1*,* $\{o_5, o_6\}$*, null), whereas the tag object* o_5 *is modeled as (TG, Bob,* o_4*, null, null).*

Besides creating and sharing objects, there are different types of actions that users can perform on them (see Table 5.3). We accordingly define the set of controllable privileges in LAMPS as PS = *{read, add-comment, add-like, add-tag, share, write}*. We recall that the *write* privilege in our context refers to users posting on their friends' walls.

Users should be able to express access requirements on their objects w.r.t. which privileges from PS can be performed on an object and by whom. As mentioned earlier, access requirements in LAMPS are expressed by means of labels, formally defined in the next section.

Privacy Requirements Formulation

To express privacy requirements, users assign security labels to each of their owned objects and to each of their friends. By labeling friends, users express the limitations they want to put on them, whereas object labeling serves for expressing the sensitivity of the information and its accessibility criteria. Before formally defining these two types of labels, we first mention that

we assume a set of possible groups that users can organize their friends into. These groups are created by users depending on their organizational preferences and can be viewed as similar to users' lists or users' circles as available on Facebook and Google+, respectively. A possible example of a set of user groups might be {friends, colleagues, teammates, schoolmates, family, acquaintances}. This set is created at user's side and is only used for the purpose of creating their friends and objects security labels. In addition, we also assume in the system a totally ordered set of levels *LOS = {Unclassified (UC), Very Low (VL), Low (L), Medium (M), High (H), Very High (VH)}*, with UC < VL < L < M < H < VH. Furthermore, we consider that users have a set of predefined groups from which they can choose, as they also can create personalized ones tailored to their relationships with their friends.

We refer to a friend label as *Friend Clearance Label (FCL)* and we formally define it as follows:

Definition 5.2.2 *Friend Clearance Label - FCL. Let $a,b \in U$ be two friends in the OSN (i.e., $\exists e_{a,b} \in FR$). The label $FCL_{a,b}$, assigned by user a to user b, is denoted by the tuple $(CL_{a,b}, TS_{a,b}, GS_{a,b})$, where: (i) $CL_{a,b} \in LOS$ is the clearance level that a grants to b; (ii) $TS_{a,b} \subseteq OTS$ is the set of object types that a wants b to get access to; (iii) $GS_{a,b}$ is the set of groups that a assigns to b. We consider in the system a defined and comprehensive set of groups from which users can choose.*

We refer to object labels as *Object Sensitivity Labels (OSL)*, and we formally define them as follows:

Definition 5.2.3 *Object Sensitivity Label - OSL. Let $a \in U$ be a user in the OSN and let o be an OSN object such that $o.owner = a$. The label OSL_o of object o is denoted by the tuple (SL_o, TY_o, GS_o), where: (i) $SL_o \in LOS$ is the sensitivity level of o; (ii) $TY_o = o.type$ is the type of o; (iii) GS_o is the set of groups for which object o should be available.*

Example 5.2.2 *Let Walt be an OSN user who uploads a new photo (GP) from his graduation ceremony assigning to it OSL_{GP} =(L, P, {colleagues, family, university}). This means that GP has a low sensitivity and it concerns Walt's colleagues, family, or university-mates. Suppose now that Walt has assigned the following label to his friend Javier $FCL_{w,j}$ =(H, {P, T, V}, {colleagues, university}). This means that Walt grants to Javier a high clearance level, allows him to see his photos, texts, and videos, and he considers him a member of the colleagues and university groups.*

Access Control Decisions

As we have discussed earlier, access to objects in an OSN can be related to one of the privileges enclosed in the privileges set (PS). To reflect this broader range of privileges, compared to traditional scenarios formalizing only read and write privileges, we refer to access requests as

interaction requests - (IR). An IR can be of two types: 1) performed on an object (e.g., read or comment on an object), or 2) made to create an object related to another user (i.e., tag a user or write on her wall). We define an *interaction request* as follows:

Definition 5.2.4 *Interaction Request - IR. Let $a, b \in U$ be two friends in the OSN (i.e., $\exists e_{a,b} \in FR$). An IR by user b on user a is denoted by the tuple $ir_{b,x} = (x,\ p,\ b)$, where $p \in PS$ is the requested privilege, and x has one of the following forms:*

- *$x = o$, where o is an object s.t. o.owner = a;* ***if*** *$p \in$ {read, add-like, add-comment}.*
- *$x = o$, where o is an object s.t. o.owner = a and o.parent = null;* ***if*** *$p =$ share.*
- *$x = a$, where a is the target user;* ***if*** *$p =$ write.*
- *$x = (a, o)$, where a is the target user and o is an object;* ***if*** *p =add-tag.*

Like in standard MAC, the evaluation of an IR is performed based on defined properties that establish orders between subject and object labels. Generally, in standard MAC, there are two properties that govern the system. These are related to the *read* and to the *write* requests, respectively. However, LAMPS requires not only the redefinition of these two properties to account for all the features in the FCLs and the OSLs, but it also requires the introduction of new axioms to take into account tha interactions in an OSNs are not limited to simple *read* and *write* privileges.

The fundamental security property

Order relations among labels must be defined to have the basis on which they can be compared. In LAMPS, we refer to this relation as the *dominance relationship*, formally defined as follows:

Definition 5.2.5 *Dominance relationship. Let $a, b \in U$ be two friends in the OSN (i.e., $\exists e_{a,b} \in FR$) and let $FCL_{a,b}$=($CL_{a,b}$, $TS_{a,b}$, $GS_{a,b}$) be the label a assigned to b. Let o be an object s.t. o.owner = a, and let $OSL_o = (SL_o, TY_o, GS_o)$ be its OSL. $FCL_{a,b}$ dominates OSL_o, denoted as $FCL_{a,b} \gg OSL_o$, if an only if,*

$$CL_{a,b} > SL_o \wedge TY_o \in TS_{a,b} \wedge GS_o \cap GS_{a,b} \neq \emptyset$$

Example 5.2.3 *Consider Example 5.2.2. The FCL that Walt assigned to Javier dominates the OSL that he assigned to his graduation photo because: 1) the clearance level assigned to Javier (High) is higher than the sensitivity level of the photo (Low); 2) object type photo is within the types set in Javier's label; and 3) the group sets in Javier's and in the photo's labels have groups in common (university, colleague).*

We are now ready to introduce the first property in LAMPS, referred to as the *Fundamental Security Property (FSP)*:

Definition 5.2.6 *Fundamental Security Property. Let $a, b \in U$ be two friends in the OSN (i.e., $\exists e_{a,b} \in FR$) and let $FCL_{a,b}$ be the label a assigned to b. Let o be an object s.t. $o.owner = a$, and let OSL_o be its label. Let $ir_{b,o} = (o, p, b)$ be an IR made by b on o s.t $p \in$ {read, share, add-like, add-comment}. $ir_{b,o}$ satisfies FSP, if and only if:*

$$FCL_{a,b} \gg OSL_o \ .$$

Informally, the FSP dictates that to grant a *read*, *share*, *add-like*, or *add-comment* IR on an object, the FCL of the requestor must dominate the OSL of the object it targets. A requirement of the FSP is that the requestor and the object owner are friends in the OSN as otherwise the friend label to be compared would not exist. As a consequence, in LAMPS, only friends of a user can have access to and interact on her information space. This is not fully aligned with the classical approach of OSNs where there is the possibility of making information available to friends-of-friends or public to all the OSN users. Regarding the case of information desired to be made public (i.e., accessible to all users in the OSN), it can be easily enabled by making the object unclassified (i.e., the object has a sensitivity level $UC \in LOS$) and by letting the system assign a default label $FCL_{default}$=(UC, GS, OTS) to all OSN users who are not direct friends with the object owner, where GS, and OTS are the sets of all groups and of all object types in the OSN, respectively. As for other information that is not made publicly available, our approach is to provide users with an environment that facilitates the understanding and control of their objects audiences. This is ensured by the suggested labels assignment, as users can be fully aware of (and track) who might gain access to what, contrary to the approach of allowing the unlimited and uncontrollable friend-of-friend access. However, LAMPS can still be extended to cover such a scenario by complementing it with a mechanism for automatic FCLs assignment to friends of friends.

Another important note regarding the FSP is that a *read* IR that satisfies this property grants a *read* access to the target object, that, according to Definition 5.2.4, is an independent object. Normally, when a user requests access to an independent object, a photo for example, she also implicitly requests *read* access to all its dependent information. However, we recall that in LAMPS access to dependent objects is restricted by their respective OSLs assigned by their corresponding owners. For this, when a *read* IR on an object o is granted, the system issues a similar IR on all the children of o to ensure their evaluation independently of their parent object and only based on their proper OSLs. If one of these system IRs is granted, the same propagation is performed on its second level children, and so on.

The FSP is sufficient for all the IRs it targets, except from the *share* one. This is what we address in the following sub-section.

The *share* privilege and the share up property

When a *share* IR is granted, it results in the creation of a copy of the original object, that is owned by the sharing user. The problem here is on what should be the label of this shared

copy. The simplest design idea is to make the copy inherit the same label as the original object, ensuring by this that only the audience of the original object is allowed to view the shared copy. This will map to what is being the standard in current OSNs, such as Facebook. However, this design choice results in two limitations that could not be desirable: 1) the user who performs the *share* interaction would have no way to influence the privacy settings of the shared copy; and 2) this strategy limits the intended purpose of the *share* functionality, since it does not result in an enlargement of the intended audience. Indeed, when a user allows a friend to share an object, this logically means that the user wants to delegate to her/him the dissemination of that object in order to make it available to a wider audience from the friend's side. Otherwise, if the user wants to limit the visibility of an object strictly to her allowed audience, then what could be the purpose from allowing a *share* interaction on it? Therefore, an alternative design decision, that would offer more flexibility and give meaning to the power of a *share* operation, is to have the shared copy have its distinct object label. However, the copy's label should still respect the privacy of the original object in such a way that the copy might be available to a wider audience without violating the privacy of the original object. Herein, the focus is on two concerns: 1) the shared copy *so* of an object *o* should not be made available to the friends of *o.owner* who are initially not allowed to access *o*; 2) the delegation offered by *o.owner* to the friend who creates the shared copy *so* should respect the sensitivity of *o*, such that *so* is shared only with those friends of *so.owner* who are at least as trusted as the sensitivity of *o*. We present Example 5.2.4 for a better illustration of these two concerns.

Example 5.2.4 *Consider Example 5.2.2, and assume that Walt and Javier have Mina as a common friend in the OSN. Let Walt's FCL for Mina be* $FCL_{w,m}$ *=(VL, {T}, {university}). Recall that the OSL of Walt's photo is,* OSL_{GP} *=(L, P, {colleagues, family, university}). Clearly, Mina does not have access to Walt's photo as* $FCL_{w,m}$ *does not dominate* OSL_{GP}*. Assume now that Walt allows Javier to share the photo; hence delegates to him its broader dissemination. This delegation means that Walt entrusts Javier to disseminate the picture to those of his friends that he trusts at least as high as the sensitivity of the photo (i.e., Walt's picture has a Low sensitivity, therefore it should be disseminated to those friends of Javier whom he trusts with at least a Low value). Furthermore, the shared photo by Javier should not be made available to Mina, no matter how Javier trust's in Mina could be different from Walt's.*

To address these two concerns, we first introduce an additional axiom, called the *Share Higher Property*. Before formally defining it, we introduce a new required concept that governs the relationship between the label of an object and that of its copies made from a granted *share* IR. We refer to this as the *Not-declassify relationship*:

Definition 5.2.7 (Not-declassify relationship) *Let* $OSL_o = (SL_o, TY_o, GS_o)$ *be the label of an object o. Let* $OSL_{\overline{o}} = (SL_{\overline{o}}, TY_{\overline{o}}, GS_{\overline{o}})$ *be the label assigned to a copy* $\overline{o}$ *of o (i.e.,*

$\overline{o}.copyof = o$). Label $OSL_{\overline{o}}$ does not declassify label OSL_o, and we write $OSL_{\overline{o}} \not\preceq OSL_o$, if and only if: $SL_{\overline{o}} \geq SL_o$.

The Not-declassify relationship requires that the sensitivity level of an object's copy is at least equal to that of the original object. This would ensure that a share action does not declassify the shared information, with the assumption of a delegated trust from the original object's owner to the sharing friend. It is also to be noted that no restriction is made on the groups set parameter of the label. This is because the organization of friends into groups is dependent on each user without necessarily mapping to the groupings adopted by their friends. For example, those who might be family to a user, might be colleagues to another. However, it is worth mentioning that the purpose of the share action is to disseminate objects to wider and different audiences, as long as this does not result in explicitly making the information available to a non-desired friend.

Based on the Not-declassify relationship, the second axiom of LAMPS, the *Share Higher Property*, is defined as follows:

Definition 5.2.8 (Share Higher Property - SHP) *Let $a, b \in U$ be two friends in the OSN (i.e., $\exists e_{a,b} \in FR$), and let o be an object s.t. $o.owner = a$. Let OSL_o be the label a assigned to o. Let $ir_{b,o} = (o, share, b)$ be a share IR made by b on o, and let $OSL_{\overline{o}}$ be the object label that b intends to assign to the copy of o. We say that $ir_{b,o}$ satisfies SHP, if and only if: $OSL_{\overline{o}} \not\preceq OSL_o$.*

Informally, the SHP enforces that users can share their friends' objects only if they assign to the shared copy an OSL that does not declassify the one of the original object. We recall that a *share* IR should first satisfy the FSP. Therefore, only those friends who have a clearance level at least equal to the sensitivity level of the object can perform a *share* IR. This ensures respecting the delegation of objects dissemination to only those friends who are at least as trusted as the sensitivity of the object they are entrusted to share.

The SHP addresses the delegation of dissemination concern, however it does not solve the first concern we discussed earlier related to when an object owner and the sharing friend have a friend in common who is not initially allowed to access the object by the owner (the case of Mina in Example 5.2.4). To prevent such *horizontal disclosures*, we enforce the preferences of objects' owners by considering their labels and not the copy's when the requestor, the sharer, and the owner are mutual friends. For the scenario in Example 5.2.4, for instance, if Mina issues a *read* IR on the shared copy of Walt's picture from the common friend Javier, the labels assigned by Walt for the original picture and for Mina are used to evaluate the IR, and not the ones assigned by Javier to the shared copy and to Mina. This is enforced by Algorithm 5 as it will be presented later.

The *add-tag* & *write* privileges and the write higher property

Both the *add-tag* and the *write* interactions, if granted, result in the creation of new objects, and therefore they bring to the problem of what the labels for these resulting objects should be. For instance, assume Bob has a very high level of trust in Alice and allows her to post on his wall. Alice might hold sensitive information about Bob and, as such, her posts on his wall might reflect some of it. Thus, Bob would want that Alice's posts on his wall be managed with a label that reflects their expected sensitivity; that is, a label that is at least as high as the one he assigned to Alice. On the other hand, if Bob's trust on another user, say Aliah, is low, then he might need to impose more control on her posts on his wall as it could be that she holds some sensitive information about him that she might post to embarrass him, for example. The same applies to the *add-tag* interaction as well. To cope with this, we suggest enforcing two conditions on the labels that requestors of granted *write* or *add-tag* IRs can assign to the resulting objects. The first condition imposes a sensitivity level for the created object that is at least as high as the clearance level that the affected user assigned to the requestor, if this latter is higher than the medium level. For example, if Bob assigns a high clearance level to Alice, Alice's posts on his wall will have at least a high sensitivity level. However, if the clearance level assigned by the affected user to the requestor is lower than the medium level, its inverse is set as the least requirement for the sensitivity level of the resulting object. For instance, if Bob assigns to Aliah a low clearance level, Aliah's posts on his wall will have a sensitivity level at least equal to the inverse of Aliah's clearance level (i.e., a high sensitivity level).

As mentioned before, the rationale behind this is that if Bob highly trusts Alice, then most probably the content she will post about him would be sensitive and should be available only to other friends he trusts at least as much as he trusts Alice. However, if Bob's trust in Aliah is very low, he might want to enforce more control over what she posts about him, and so the inverse of his trust in her is used as a least requirement for Aliah's posts on Bob's wall.

The second condition imposes that the group set of the resulting object's label is exactly the group set that the affected user specified in her label for the requestor. For example, if Bob assigned the group "colleagues" to Alice, then it is likely that Alice's posts about him will also concern the "colleagues" group. To formalize these conditions, we introduce the third axiom of LAMP, referred to as the *Write Higher Property - (WHT)*:

Definition 5.2.9 (Write Higher Property - WHT) *Let $a,b \in U$ be two friends in the OSN (i.e., $\exists e_{a,b} \in FR$) and let $FCL_{a,b}$=($CL_{a,b}$, $TS_{a,b}$, $GS_{a,b}$) be the label a assigned to b. Let $ir_{b,x} = (x,p,b)$ be an IR such that $p \in$ {write, add-tag} and $x = a$, if $p =$ write, $x = (a, ob)$, if $p =$ add-tag, with ob the object to be tagged. Let o be the object resulting from granting $ir_{b,x}$ and OSL_o be its assigned label by b. $ir_{b,x}$ satisfies WHT if and only if: o.owner $= a \wedge$ $OSL_o = (SL_o, TY_o, GS_o)$ s.t., $GS_o = GS_{a,b}$,*

$$TY_o = \begin{cases} TG & \text{if } p\text{=}\textit{add-tag}. \\ FP & \text{if } p\text{=}\textit{write}. \end{cases}$$

and

$$SL_o \geq \begin{cases} CL_{a,b} & \text{if } CL_{a,b} \geq M. \\ f^{-1}(CL_{a,b}) & \text{otherwise}. \end{cases}$$

With $f^{-1}(y), y \in LOS$ *being the complement function for security levels. For example* $f^{-1}(VH) = VL$, $f^{-1}(L) = H$, *etc.*

Example 5.2.5 *Consider Example 5.2.2, and assume Javier has the right to post on Walt's wall and that he posts on it a video of Walt's graduation ceremony. We recall that Walt assigned to Javier (H, {P, T, V}, {colleagues, university}). By the WHT property, the video's owner is Walt, and a possible granted label that Javier can assign to it is: (H, FP, {colleagues, university}). This means that the video will be available only to the friends of Walt to whom he assigned a high or a very high clearance level, who belong to the colleagues or to the university groups, and who are allowed to see Walt's friends' posts.*

Access control enforcement

Putting it up all together, Algorithm 5 defines how access control is enforced in the system based on the privacy preferences of users as expressed in FCLs and OSLs and on the properties defined above.

Algorithm 5 takes as input an IR to be evaluated and produces as output a *granted* or *denied* message. The algorithm switches the privilege requested by the IR and ensures the enforcement of the corresponding properties. More precisely, for a *read* IR, the algorithm starts by checking if the target object is original or a copy. If it is a copy, it makes a call to the *NoHorDisclosure* function that ensures the rewriting of the IR, as long as the involved parties are mutual friends (the *IsFriend* call in line 37), to enforce the labels of the copied object and of its owner and not of the copy's (lines 3,4). After making the IR immune to horizontal disclosures, it is evaluated for FSP satisfaction. In case of a positive result, a similar IR request should be propagated to all the children of the target object in the algorithm's input IR. This is ensured by the call to the *propagate* procedure, in line 6, that makes recursive calls throughout all the offspring branches of the object as long as the IR on the parent object node satisfies the FSP. Procedure *propagate* makes read IRs on each sub branch of the parent object, evaluates these IRs against the FSP, and sends a *granted* message for each positively evaluated one (lines 42-48). In case the input IR does not satisfy the FSP, a *denied* message is returned (line 8).

For a *write* IR, the algorithm first checks if the requesting user has a *write* privilege on the targeted user's wall. This is achieved by making a *read* IR on the *root* object of the targeted user (lines 10,11). If the new generated *read* IR satisfies the FSP and the input *write* IR satisfies the WHP, this latter is *granted*, otherwise a *denied* message is returned (lines 12-15).

Algorithm 5: Access control enforcement in LAMPS

Input : An IR, $ir = (x, p, b)$.
Output : *granted* or *denied*.

```
1  switch p do
2      case read
3          //x is an object;
4          if x.copyof ≠ null then
5              ir = NoHorDisclosure(ir);
6          end
7          if SatisfyFSP(ir) then
8              Propagate(ir);
9          else
10             Send(denied);
11         end
12     end
13     case write
14         //x is a target user;
15         root_x = GetRootOf(x);
16         ir_allowed = makeIR(root_x, read, b);
17         if SatisfyFSP(ir_allowed) ∧ SatisfyWHP(ir) then
18             Send(granted);
19         else
20             Send(denied);
21         end
22     end
23     case share
24         if SatisfyFSP(ir) ∧ SatisfySHP(ir) then
25             Send(granted);
26         else
27             Send(denied);
28         end
29     end
30     case add − tag
31         //x is a pair (usr, obj);
32         ir_allowed = makeIR(obj, read, b);
33         ir = makeIR(usr, p, b);
34         if SatisfyFSP(ir_allowed) ∧ SatisfyWHP(ir) then
35             Send(granted);
36         else
37             Send(denied);
38         end
39     end
40     case add − like ∨ add-comment
41         //x is an object;
42         ir_read = makeIR(x, read, b);
43         ir = makeIR(usr, p, b);
44         if SatisfyFSP(ir_read) ∧ SatisfyFSP(ir) then
45             Send(granted);
46         else
47             Send(denied);
48         end
49     end
50     otherwise
51         Send(denied)
52     end
53 endsw
```

Procedure Propagate($ir_{a,o}$)

```
1 Send(granted);
2 while ch = o.nextChild ≠ null do
3 |   IR = makeIR(ch,read,a);
4 |   if SatisfyFSP(IR) then
5 |   |   Propagate(IR);
6 |   end
7 end
```

Function NoHorDisclosure($ir_{a,o}$)

```
1 IR = ir_{a,o};
2 while o.copyof ≠ null do
3 |   if areFriends(a, o.copyof.owner) then
4 |   |   IR = ir_{a,o.copyof};
5 |   end
6 |   ir_{a,o} = ir_{a,o.copyof};
7 end
8 return IR;
```

In case the input IR is a *share* one (line 16), it is checked if it satisfies both the FSP and the SHP sending a *granted* message if it does, or a *denied* one in the failing case.

As for *add-tag* IRs, a *read* IR on the object to be tagged is formulated by the system (line 22). Then it is checked if this system issued IR satisfies the FSP and if the input *add-tag* IR satisfies the WHP. If the two conditions are satisfied, the input IR is *granted*, otherwise it is *denied*.

Finally, for *add-like* and *add-comment* IRs, they are granted only if they satisfy the FSP and if a system issued *read* IR on the object they target (line 29) is granted (i.e., it satisfies the FSP too) (lines 30-33).

5.2.3 Correctness and Complexity Analyses

In this section, we present the security properties of the LAMPS model and we discuss the complexity of its enforcement Algorithm 5. Detailed proof of the theorem is presented in Appendix B.

Correctness property

Access control in LAMPS is enforced by Algorithm 5 that ensures that all IRs are granted only if they satisfy the model's properties applying to them. To formalize the correctness property of the system, we define the concept of its *state* as the set, $SIR = \{ir_1, ir_2, ..ir_n\}$, of interaction requests currently granted in the system. We say that the state of the system is

correct, and we refer to it as the *correct state* if and only if $\forall ir_i \in SIR, ir_i$ satisfies all the model's properties. The system changes its state only when a new IR processed by Algorithm 5 returns a granted message. Given a *correct state*, the following holds (related proofs are reported in **Appendix B**):

Theorem 5.2.1 *Let SIR be the current state of the system. Let ir_{new} be a new interaction request issued to the system. Algorithm 5 issues a granted message if and only if $SIR_{new} = SIR \cup \{ir_{new}\}$ is a correct state.*

Complexity analysis

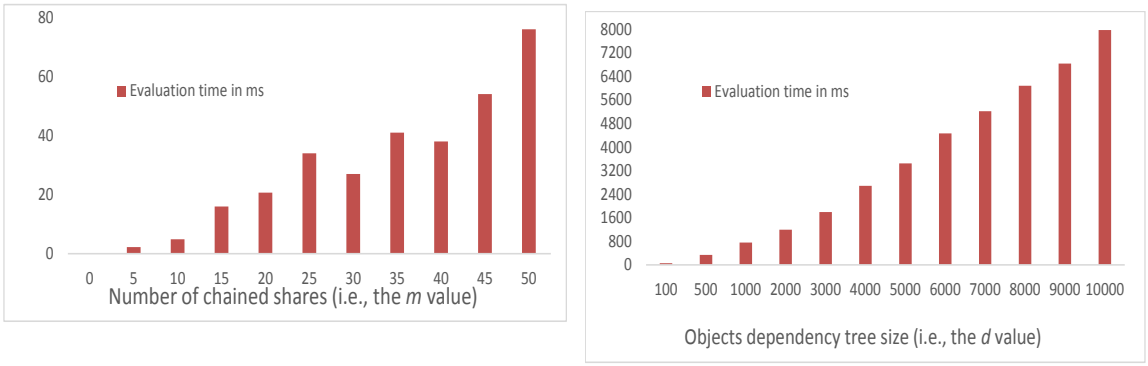
The complexity of Algorithm 5 depends on the evaluation of requests with a *read* privilege as this has the highest cost, with the calls to the *NoHorDiscolusre* function and to the *propagate* procedure. The *NoHorDiscolusre* function is $\mathcal{O}(m)$, where m is the maximum number of chained shares an OSN object is subject to, whereas procedure *propagate* is $\mathcal{O}(d)$, with d being the number of children in an object dependency tree. m is expected to be small. Research works demonstrate the principle of six-degrees of separation in today's OSNs with even a shorter scale (i.e., four-degrees) [40]. Therefore, it can be estimated that at most $m = 6$. Regarding d, it is reasonably low except for famous content (e.g., celebrity's pictures or status updates) that would typically receive a very high number of likes and/or comments. However, such instances of famous content would generally make a small portion of all content generated on an OSN and would, most probably, have a completely public privacy setting. Therefore, access to such content could be set as granted by default in the system.

5.2.4 Experiments and Results on LAMPS

We have designed two sets of experiments for the LAMPS model. In the first, we study the performance of LAMPS under access scenarios simulated on a real OSN graph. In the second, we conduct a preliminary usability evaluation to study the friendliness of our solution. We have implemented a prototype of LAMPS as a web application (hereafter referred to as *LAMPS app*) that we interfaced with Facebook graph API. The prototype has been implemented using PHP and the MySQL DBMS. Both the application and the corresponding database have been mounted on a standard PC with a dual core processor of 2.99 GHZ each and 8 GB of RAM.

Performance experiments

For this experiment, we exploited the public Pokec OSN's dataset previously exploited for experiments on CARDS model (see Section 5.1.4). We recall that the dataset's graph is made of 1.63 million nodes and 30.6 million edges with a diameter equal to 11.



(a) Evaluation time (ms) for access requests on shared copies

(b) Evaluation time (ms) by objects dependency tree size

Figure 5.8 Performance of LAMPS on shared and dependent objects

The performance of the LAMPS model in evaluating an access request is regulated by the number of children in the target object's dependency tree (i.e., d), and by the number of chained shares the requested object is subject to (i.e., m). For this, we performed two experiments, by varying both m and d independently. As such, we first simulated share operations along friendship paths of different lengths, and we computed the time required to evaluate an access request on each of the shared copies. Figure 5.8a reports the achieved results. The x-axis refers to the number of chained shares an object has been subject to, whereas the y-axis refers to the average time in milliseconds required to evaluate an access request on the last shared copy. All the reported numbers are the result of averaging 40 individual instances. As we can see on the figure, the time required to evaluate an access request on a shared copy is directly proportional to the distance between the target shared copy and the original object it refers to (i.e., the number of chained shares). However, as it can be read on the figure, the case of a chained share of length 50 requires no more than 80 ms. We can also see that the evaluation time does not follow a perfect linear increase with the number of chained shares. This is because the friendship sub-graph surrounding the sharing path also affects the convergence of the evaluation, as the *NoHorDisclosure* function in Algorithm 5 makes recursive calls as long as a common friendship is found to be between the requestor and the owner of the copied object.

We have also simulated object dependency trees of different sizes (d) varying both their breadth and depth. We considered the worst case scenario whereby the access evaluation on all children of a tree is positive. Figure 5.8b reports the achieved results by plotting the time in milliseconds (the y-axis) for tree sizes varying from 100 to 10K children. This range has been set based on a study of Facebook usage as of February 2015, that revealed that the average number of likes for personal content is below 100; whereas it is in the order of 100K for business and fans pages. For each reported tree size, the evaluation time has been

averaged across 10 different trees by varying their breadth and depth sizes. For all considered trees, the breadth made at least 80% of the tree's size. This is because, according to the same study mentioned above, most of the interactions recorded on content on Facebook are one level likes or one level comments.

As it can be seen on the figure, the amount of time required to evaluate access requests on all the children in an objects dependency tree increases, as expected, almost linearly with the size of the tree. We can see that the evaluation time still remains below 8k ms for trees of 10k children in size. We underline the fact that we are running our experiments on a PC with a standard configuration.

Preliminary usability experiments

We exploited summative testing to evaluate the usability of LAMPS [44]. Summative evaluation is a benchmark technique for testing the usability of products or processes, that is defined in terms of effectiveness, efficiency and satisfaction. A minimum of 12 participants representative of the users base is often considered to allow statistical analysis [44]. We conduct both a summative evaluation of LAMPS and a comparative study between it and Facebook privacy settings. We describe our experiments and report the results in what follows.

Summative evaluation of LAMPS

Two main tasks were considered: 1) assign FCLs to friends, and 2) create a post, assign to it an OSL, and publish it. Participants were asked to login to LAMPS app using their Facebook credentials and to accept to grant to it access to their wall both for *read* and *write* privileges. With Facebook's new graph policy of 30-05-2015, the request of these privileges needed consent from Facebook. As such, the app was deployed in a testing environment and participants were explicitly declared as testers.

After login, the participants were guided through the two tasks of the experiment. More precisely, users could view, from LAMPS app, a sub-list of their friends retrieved from their Facebook friends list, and use the app's interface to assign FCLs to them. After labeling some of their friends, participants were guided to publish a text post from the app's interface and assign it an OSL. Upon publishing a labeled text post, users could view the list of friends, selected by LAMPS app from the ones they have labeled already, who should be able to view the post. After each performed step, participants were asked to answer survey questions designed to get their feedback on the perceived usability of the task they have just performed. Users could exit the app at any time and finish the experiment by answering a post-usage survey that contained all the questions that were asked while using the plus some others on how they perceive the method compared to other privacy settings they have used before.

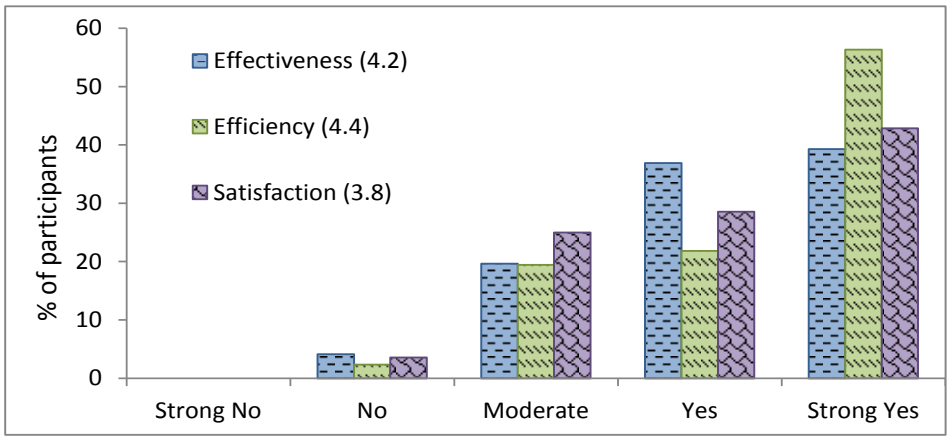


Figure 5.9 Participants feedback on LAMPS's usability, in percentage of participants

26 people, all approximately aged between 18 and 36 (16% Italian students, 10% non-students from France, US, Spain and Greece, and 74% non-students from Morocco) consented to the terms of and used LAMPS app. The average number of Facebook friends our participants have is 363, and about 68% of them reported, in a pre-experiment background survey, that they highly care about their privacy on social networks. About 20% of them answered that they do not have big concerns about their privacy.

All the survey questions (reported in **Appendix B**) were multiple choice with answers on a scale of five ordered items (i.e., Strong No, No, Moderate, Yes, Strong Yes). When processing the survey results, these items were orderly mapped to numerical values in the range [1,5].

As mentioned earlier, participants were asked to answer survey questions both while using the app and upon exiting. The survey contained a total of 10 questions that were designed and grouped to target each of the usability attributes commonly known in the literature [44]. These are *effectiveness*, *efficiency*, and *satisfaction*. The questions were distributed over the experiment to appear each time a task is completed. Each of the 26 participants in the experiment published at least one labeled post and labeled at least 4 friends. The average of published posts and the average labeled friends per participants were 4 and 10, respectively.

Figure 5.9 summarizes the obtained results by presenting the distribution, in percentage of participants on the y-axis, for each usability attribute over the five possible answers. The numerical total average is also presented for each of these attributes (the number between parentheses in front of the labels on Figure 5.9). As we can see on the figure, for the *effectiveness* attribute, that concerns the strength of LAMPS in correctly modeling users'

privacy requirements, about 77% of the participants answered with a yes or a strong yes, and more than 95% answered with at least a moderate level. Moreover, about 80% of the participants expressed at least a high level (*yes*) regarding the *efficiency* of LAMPS, reflecting by this its ease of use. Finally, More than 55% of the participants were highly satisfied with LAMPS and more than 83% of them answered that they would like to see such a method deployed in their Facebook accounts.

LAMPS vs. Facebook privacy settings

The tasks designed for this comparative study relate to the basic privacy operations in OSNs, that is, setting privacy limitations on friends, and setting privacy controls on objects. LAMPS differentiates between dependent and independent objects and we allow the specification of privacy preferences for both. However, Facebook privacy settings concern independent objects only. We recall that a dependent object (such as a comment or a like) is one that results from interactions on an independent one. For this, our comparison concerns the specification of privacy limitations on friends (task T1), and of privacy controls on independent objects only (task T2). Hereafter, we refer to the Facebook privacy settings as *FPS*. We conducted the comparative evaluation using a *within-subjects* design [44]. In contrast to the *between-subjects* design, that uses different groups of participants for each of the compared products, *within-subjects* exploits the same group of users to experiment with all the products under comparison. Its advantage is that it does not require dealing with nuisance variables, such as those related to differences in previous experiences with one of the products, to background knowledge, etc. However, this design requires counterbalancing the tasks across the compared products to avoid learning effects that might result from the order of tasks in the experiment [44]. For example, a participant might give higher evaluation to the second product only because experimenting with the first one made her more acquainted with the experiment. To go around this issue, we have designed the experiment by alternating the survey by product in an A-B-A format. That is, participants had to answer survey questions about *FPS* on both tasks T1 and T2, then about *LAMPS*, then again answer the same questions about *FPS*. The usability focus of our comparison was on *efficiency* and *satisfaction* and the participants group was the same as in the previous experiment. Details on the survey questions used for this experiment are provided in **Appendix B**.

We start by providing, in Table 5.4, a summary of the number of operations required to perform each of the two tasks considered for the comparison in each of *LAMPS*and *FSP*. As it can be read on the table, *LAMPS* ensures both tasks T1 and T2 in one operation only corresponding to creating an FCL or an OSL respectively; whereas in *FPS*, the number of operations for task T1 depends on the number of groups to which the target friend needs to belong to. Whilst for task T2, the number of operations depends on the number of friends or groups that should be allowed to view the object.

Table 5.4 *FPS* vs. *LAMPS*: complexity in terms of number of operations required to perform a privacy setting task.

T1: privacy on a friend		**T2: privacy on an object**	
FPS	*LAMPS*	*FPS*	*LAMPS*
1 operation per friend per group	1 operation per friend	1 operation per object per allowed friend/group	1 operation per object

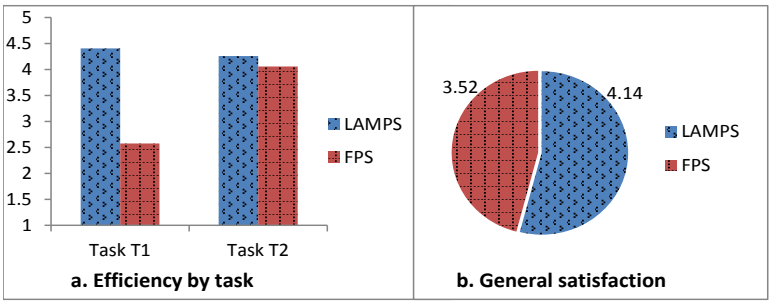


Figure 5.10 Participants feedback on the efficiency by task and the general satisfaction of FSP vs. LAMPS (values provided as total average in the range [1-5])

Figure 5.10 presents the comparison results on efficiency by each of the designed tasks T1 and T2 and on the general satisfaction with *FPS* and *LAMPS*. Regarding the efficiency, on task T2, we can read on the figure that the participants scored both *FPS* and *LAMPS* almost equal, with a slight advantage for *LAMPS*. However, for task T1, we can clearly see that participants perceived *LAMPS* as considerably more efficient than *FPS*. We think that this could be related to the big number of friends users have on their Facebook accounts that makes hard for them to organize them into groups and access lists. In fact, when we asked the participants if they think they have friends on their Facebook that they could have completely forgotten about, about 50% of them agreed with at least a yes. On the other hand, when asked if their Facebook friends are organized into groups with defined privacy settings on them, 80% answered with at least no and 50% reported a strong no. This shows a gap between what users think about their privacy needs and how they actually enforce them. For this, we believe that our method that enforces the expression of users privacy requirements by means of assigning labels would bridge over this gap. This task can be achieved by one single operation and can be set to take place right upon the establishment of a new friendship link; resulting in the enforcement of intended privacy settings without the burden of organizing the friends into groups or customizing the settings afterwards.

Finally, and regarding the general reported satisfaction, we can see on Figure 5.10.b that *LAMPS* performs better than *FPS*. However, we believe that more experiments on this are required before drawing generalized and statistically unbiased results. This is because the time that participants experimented with and used LAMPS's interface is very short compared to their acquaintance with Facebook.

Chapter 6

Conclusion

In this thesis, we have claimed that decentralization of online social networks may be a solution to privacy concerns on a macro-level but it is opening new challenges for meeting privacy at the micro-level. We have identified three of these main challenges that we specifically intended to address.

Firstly, the cyrptographic approach that is traditionally adopted to manage the confidentiality of data in decentralized systems in general, and in DOSNs in particular, is considerably limiting the *usability, and efficiency* of the decentralized social networks alternatives.

Secondly, the reliance on only a relationship based model for access control limits the expressiveness of possible privacy policies to links between users without consideration for other contextual and/or interactional aspects. This limits the ability of users to *express richer access policies* beyond personal relationships tracking.

Finally, the looseness of the identification process in social networking sites in general constitutes a major threat to users privacy and soundness of enforced access controls, as fake accounts may gain legitimate access to one's data simply by convincing her/him of befriending them. The identification, or identity validation problem, is expected to face more serious challenges under decentralized architectures for social networks, where no central control could be enforced (e.g., closing reported fake accounts, identifying accounts validity from usage patterns, etc.).

We recognize that spending efforts to achieve absolute security or privacy protection is merely a fool's errand. Not only is privacy a subjective concept that changes definition and scope based on whose privacy it is about, but it is also dependent on many factors, some of which may not be deterministic. Therefore, in addressing our three identified areas of action, we have taken the approach of exploring solutions that would offer flexibility, transparency, and that would actively engage the concerned actors (i.e., the users).

We first provide a general summary of the main contributions of this thesis, reflecting on the key principles and major learned lessons (Section 6.1). We conclude the thesis with insights on potential future directions drawn from this work, and with an open discussion

on the future of decentralized approaches to social networks from a privacy and security perspectives (Section 6.2).

6.1 Summary and Reflections

The core underlying principal in this thesis is to design privacy preserving services for decentralized social networks that are usable, offer increased efficiency, and that mostly are users engaging. There are many models and proposals offering different solutions for users to manage their online privacy in both centralized and decentralized social networks, but the challenge we take in this thesis lies in exploring alternative access control models and paradigms to improve the efficiency, usability, and richness of online privacy services, especially under decentralized social networks. Our work resulted in key contributions that could be summarized in three:

First, a substantial contribution to the problem of identity validation in social networks has been provided. A methodology for studying the validity of an online identity from profile content only and without the reliance on a central point of control has been suggested. The performed work under this line opens the potential for more research in the area of empowering honest social network users with systematic frameworks within which they could collaborate to maintain clean social networking environments. We believe that our work is the first to provide a formal study of identity validation in a fully decentralized users collaborative manner with reliance on minimum resources (i.e., profiles content only).

Second, a revolutionary paradigm to the access control problem in social networks have been explored. As far as we are informed, this is the first work that attempts to approach users online privacy in social networks with an aposteriori paradigm. We believe that transparency and accountability are required to offer a complete privacy preserving service within social networks, especially under the decentralized model. Relying on secure locking mechanisms, such as cyrptography, cannot ensure confinement of the data. That is, once a peer gets hold of an object, the system cannot provide any guarantees that this peer may not re share it under different access requirements, making it available to unintended audiences by its original owner. We claim that transparent controls, within environments of trust that are controlled by responsibility and accountability, may be the best solution for this problem. We hope that our suggested model opens doors to further research efforts along this direction of audit based controls for the social web.

Third, we have explored the design of access controls in social networks under a model different from the traditionally explored relationship based one. We have exploited label based access control, that is a form of the mandatory access control model in a user-centric fashion, to capture interactional controls and the relationships between data elements as resulting from these interactions in the system. We have proved that the suggested model provides a much larger scope of fine grain access controls compared to what is currently

enabled in major social networking sites. Our preliminary experiments pinpoint the potential of our suggested approach, and calls for more work towards exploring such alternative models.

Finally, we have shed the light on a special case concerning users online privacy in social networks once they are no more available to manage their data. We have performed a surveying of existing practices and research results, which we claim are not enough compared to the importance of the topic, and we have laid the foundation for an integrated model to address this problem.

Reflections

Throughout the elaboration of our performed studies, some key lessons have been learned. We summarize them here to provide more insight on our contributions.

First, starting from our first area of action regarding identities validation, we have learned that any trial to address this issue should take into account both topological (i.e., the underlying graph in terms of communities and users connections) and semantic characteristics (i.e., profile values). More specifically, we have found from our studies that there exists an existential leveling between users connections and the content of their profiles. What could be considered from a first view as stereotypes, such as musicians usually do not do bodybuilding sports, could be mapping to real existing patterns in given communities. Such patterns should not be over-sought as they could be a key to differentiating real from fake profiles in the realms of open and uncontrolled decentralized social networks. We specify here that we do not consider profiles as static end points that only describe basic information about users, but as dynamic representations of which the content is affected by the updates and the behaviors of the users in the network.

Second, we have realized that the development of any online privacy service for social networks requires the understanding of users behavior and interactions, with the hope to correctly capture the scope of their needs. Indeed, as social networks functionality, services, and data sharing patterns keep to evolve, users privacy, thus the corresponding services, should keep aligned. This alignment could only be achieved by engaging the concerned users in the process.

Finally, there are number of design and implementation challenges in approaching the problem of online privacy services for the social web. First, any in-depth study requires having hands on representative datasets and users base. Without the existence of these two elements, the validation of the usability and the practicality of any suggested solution remains speculative at best. Second, there is the essential challenge of understanding what users need and how to best model an underlying framework that would offer a pleasant system-human interaction.

6.2 Future Directions and Insights

Although the access control problem could be viewed as massively studied in the literature, and probably considered at its maturity level already, we believe that products, such as social networks, have and continue to challenge this probably well established concern. Social networks have been developing in ways that could be challenging their own scope of definition. People are encouraged to share more and more of their data within scopes that stretch to merge wider spans of personal aspects with public information. Status updates, twitting, expressing likes and reactions to published content that could be either personal or public news, joining pages and interest groups, etc., are examples of functions that are encouraged in today's major social networking sites. They result in the generation of highly connected and semantically rich content, making computer programs learn trends and patterns that are, probably, beyond the understanding of most people. Therefore, privacy concerns in the social web, on all levels, will continue to challenge the research body for more usable, more efficient, and more transparent solutions.

The suggestion of decentralized architectures for social networks might bring technically strong potential to give back power to the users w.r.t the management of their data. However, it is challenged not only from a technical perspective, but also from a business model one. The industry of social networks, with all the businesses that have been blossoming on top of its infrastructure, would collapse under the decentralized model, unless alternative business models, with accompanying proper mechanisms for collaboration, are conceptualized. Our focus in this thesis was on the technical challenges to the conception of decentralized social networks from a data online privacy focal point. In light of the corresponding investigations and the studies, as reported in this thesis, we highlight the key directions that we found may be of more pertinent interest for future work.

First, it is planned to evaluate the potential of automating the suggested identity validation scheme in a decentralized architecture. Thus far, we have achieved unsupervised and fully decentralized learning of the identity trends within communities; however, the validation is still dependent on users judgment. Automating the evaluation task could be explored as a prediction problem that would take into account both topological and semantic characteristics. The challenge in achieving this under a decentralized architecture is in ensuring efficiency, reliability, and security of the model.

Second, it is projected to expand experiments on the aposteriori paradigm to access control within real environment. The idea is to put the model under deployment testing to study users interactions and potential to collaborate. The achievement of this requires the engagement of a representative users base for a time that should be long enough to significantly prove the results. We believe that this could be done in an in-lab long term experiment where participants should have appropriate incentives to use the alternative solution as their main media for socializing among the group throughout all the period of

the experiment. We believe that the major weakness of most studies that could approach comparing new paradigms to what is currently available in major social networks is related to usage significance. That is, the level of engagement of participants in the experiment in terms of usage time, frequency, and intensity.

Finally, it is considered to run a statistically representative study to poll users needs w.r.t the management of their online privacy in the social web. As we have discussed under Section 6.1, users needs should be understood and formally modeled towards any practical trial to solve the online privacy dilemma. This dilemma is reflected on statistics that keep revealing that more and more users are concerned about their privacy, and that state, at the same time, huge numbers of users who report having never investigated the privacy setting tools available for them. The intriguing question herein is related to understanding the real causes of this gap. We acknowledge the value of designing fancy techniques, such as exploring different models for performing access control or taking the challenge of finding solutions in varied fields such as game theory. However, we strongly believe that, if practicability is sought, it is first required to scientifically understand what users really need, and to formally model their expected human system interaction from a privacy point of view. That is, we consider working on the gray area between privacy preservation and human-computer interaction (HCI). We believe that this might drive the construction of systems that could achieve acceptable or optimal trade offs between security and privacy concerns, and usability and quality of service without or with minimal effort from users side.

Bibliography

[1] Adu-Oppong, F., Gardiner, C. K., Kapadia, A., and Tsang, P. P. (2008). Social circles: Tackling privacy in social networks. In *Symposium on Usable Privacy and Security (SOUPS)*.

[2] Agrawal, R., Imieliński, T., and Swami, A. (1993). Mining association rules between sets of items in large databases. In *ACM SIGMOD Record*. ACM.

[3] Agrawal, R., Srikant, R., et al. (1994). Fast algorithms for mining association rules. In *Proc. 20th int. conf. very large data bases, VLDB*, volume 1215, pages 487–499.

[4] Akcora, C. G., Carminati, B., and Ferrari, E. (2011). Network and profile based measures for user similarities on social networks. In *Information Reuse and Integration (IRI), 2011 IEEE International Conference on*, pages 292–298. IEEE.

[5] Akcora, C. G., Carminati, B., and Ferrari, E. (2012). Privacy in social networks: How risky is your social graph? In *ICDE'12*, pages 9–19. IEEE.

[6] Ala-Kleemola, T. and Tolvanen, S. (2012). Reputation management system. US Patent 8,112,515.

[7] Allen, A. L. (1988). *Uneasy access: Privacy for women in a free society*. Rowman & Littlefield.

[9] Baden, R., Bender, A., Spring, N., Bhattacharjee, B., and Starin, D. (2009). Persona: an online social network with user-defined privacy. In *ACM SIGCOMM Computer Communication Review*, volume 39, pages 135–146. ACM.

[10] Baez, J. C., Fritz, T., and Leinster, T. (2011). A characterization of entropy in terms of information loss. *Entropy*, 13(11):1945–1957.

[Bahri et al.] Bahri, L., Carminati, B., and Ferrari, E. Coip - continuous, operable, impartial, and privacy-aware identity validity estimation for osn profiles. *ACM TWEB*.

[12] Bahri, L., Carminati, B., and Ferrari, E. (2014). Community-based identity validation on online social networks. In *Proceedings of the 2014 IEEE 34th International Conference on Distributed Computing Systems (ICDCS)*, pages 21–30. IEEE.

[13] Bahri, L., Carminati, B., and Ferrari, E. (2015a). Cards - collaborative audit and report data sharing in decentralized social networks. In *Proceedings of the 2015 IEEE International Conference on Collaboration and Internet Computing (CIC)*. IEEE.

[14] Bahri, L., Carminati, B., and Ferrari, E. (2015b). What happens to my online social estate when i am gone? an integrated approach to posthumous online data management. In *Proceedings of the 2015 IEEE International Conference on Information Reuse and Integration (IRI)*, pages 31–38. IEEE.

[15] Bahri, L., Carminati, B., Ferrari, E., and Lucia, W. (2016a). Lamps - label-based access control for more privacy settings in osns. In *Submission process*. Under revision.

[16] Bahri, L., Soliman, A., Squillaci, J., Carminati, B., Ferrari, E., and Girdzijauskas, S. (2016b). *BeatTheDIVa* - decentralized identity validation for online social networks. In *Proceedings of the 2016 IEEE 32nd International Conference on Data Engineering (ICDE)[Demo Track]*. IEEE.

[17] Bambauer, D. E. (2013). Privacy versus security. *Journal Of Criminal Law and Criminology 103.3: 667-683*.

[18] Banks, L. and Wu, S. F. (2009). All friends are not created equal: An interaction intensity based approach to privacy in online social networks. In *Computational Science and Engineering, 2009. CSE'09. International Conference on*, volume 4, pages 970–974. IEEE.

[19] Bernoff, J. and Anderson, J. (2010). Social technographics defined. *Forrester. Retrieved December*, 8:2010.

[20] Berry, M. J. (1997). Gordon. s. linoff. *Data Mining Technique: For Marketing, Sales, and Customer Relationship Management*.

[22] Bielenberg, A., Helm, L., Gentilucci, A., Stefanescu, D., and Zhang, H. (2012). The growth of diaspora-a decentralized online social network in the wild. In *Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on*, pages 13–18. IEEE.

[23] Boshmaf, Y., Beznosov, K., and Ripeanu, M. (2013). Graph-based sybil detection in social and information systems. In *Advances in Social Networks Analysis and Mining (ASONAM), 2013 IEEE/ACM International Conference on*. IEEE.

[24] Brickell, J. and Shmatikov, V. (2008). The cost of privacy: destruction of data-mining utility in anonymized data publishing. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 70–78. ACM.

[26] Buchegger, S., Schiöberg, D., Vu, L.-H., and Datta, A. (2009). Peerson: P2p social networking: early experiences and insights. In *Proceedings of the Second ACM EuroSys Workshop on Social Network Systems*, pages 46–52. ACM.

[27] Cai, X., Bain, M., Krzywicki, A., Wobcke, W., Kim, Y. S., Compton, P., and Mahidadia, A. (2011). Collaborative filtering for people to people recommendation in social networks. In *AI 2010: Advances in Artificial Intelligence*, pages 476–485. Springer.

[28] Cao, J., Carminati, B., Ferrari, E., and Tan, K. L. (2008). Castle: A delay-constrained scheme for k s-anonymizing data streams. In *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, pages 1376–1378. IEEE.

[29] Carminati, B., Ferrari, E., and Perego, A. (2009). Enforcing access control in web-based social networks. *ACM Transactions on Information and System Security (TISSEC)*, 13(1):6.

[30] Carter, T. and Fe, S. (2007). An introduction to information theory and entropy. *Complex Systems Summer School, Santa Fe*.

[32] Chairunnanda, P., Pham, N., and Hengartner, U. (2011). Privacy: Gone with the typing! identifying web users by their typing patterns. In *Privacy, security, risk and trust (passat), 2011 ieee third international conference on and 2011 ieee third international conference on social computing (socialcom)*, pages 974–980. IEEE.

[33] Cheng, Y., Park, J., and Sandhu, R. (2012). Relationship-based access control for online social networks: Beyond user-to-user relationships. In *Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Confernece on Social Computing (SocialCom)*, pages 646–655. IEEE.

[34] Ciriani, V., di Vimercati, S. D. C., Foresti, S., and Samarati, P. (2007). κ-anonymity. In *Secure data management in decentralized systems*, pages 323–353. Springer.

[35] Clifton, C. and Tassa, T. (2013). On syntactic anonymity and differential privacy. In *ICDE Workshops*, pages 88–93.

[36] Conrad, E., Misenar, S., and Feldman, J. (2012). *CISSP study guide*. Newnes.

[37] Cover, T. M. and Thomas, J. A. (2012). *Elements of information theory*. John Wiley & Sons.

[38] Cutillo, L. A., Molva, R., and Strufe, T. (2009a). Privacy preserving social networking through decentralization. In *Wireless On-Demand Network Systems and Services, 2009. WONS 2009. Sixth International Conference on*, pages 145–152. IEEE.

[39] Cutillo, L. A., Molva, R., and Strufe, T. (2009b). Safebook: Feasibility of transitive cooperation for privacy on a decentralized social network. In *World of Wireless, Mobile and Multimedia Networks & Workshops, 2009. WoWMoM 2009. IEEE International Symposium on a*, pages 1–6. IEEE.

[40] Daraghmi, E. Y. and Yuan, S.-M. (2014). We are so close, less than 4 degrees separating you and me! *Computers in Human Behavior*, 30:273–285.

[41] DeCew, J. (2015). Privacy. In Zalta, E. N., editor, *The Stanford Encyclopedia of Philosophy*. Spring 2015 edition.

[42] Dewan, P. and Dasgupta, P. (2010). P2p reputation management using distributed identities and decentralized recommendation chains. *Knowledge and Data Engineering, IEEE Transactions on*.

[43] Dijkstra, E. W. (1982). On the role of scientific thought. In *Selected writings on computing: a personal perspective*, pages 60–66. Springer.

[44] Dumas, J. S. (2002). User-based evaluations. In *The human-computer interaction handbook*, pages 1093–1117. L. Erlbaum Associates Inc.

[45] Dwork, C. (2008). Differential privacy: A survey of results. In *Theory and applications of models of computation*, pages 1–19. Springer.

[46] Dwork, C., Naor, M., Pitassi, T., and Rothblum, G. N. (2010). Differential privacy under continual observation. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 715–724. ACM.

[47] Ellison, N. B. and Boyd, D. (2013). Sociality through social network sites. *The Oxford handbook of internet studies*, pages 151–172.

[48] Famulari, A. and Hecker, A. (2013). Mantle: a novel dosn leveraging free storage and local software. In *Advanced Infocomm Technology*, pages 213–224. Springer.

[49] Ferrara, E. (2012). Community structure discovery in facebook. *International Journal of Social Network Mining*.

[50] Ferrari, E. (2010). *Access Control in Data Management Systems*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers.

[51] Flaxman, A. D. (2007). Expansion and lack thereof in randomly perturbed graphs. *Internet Mathematics*, 4(2-3):131–147.

[52] Foster, B. (2014). How many users on facebook? number of facebook users, january 2014.

[54] Fung, B. C., Wang, K., and Yu, P. S. (2005). Top-down specialization for information and privacy preservation. In *Data Engineering, 2005. ICDE 2005. Proceedings. 21st International Conference on*, pages 205–216. IEEE.

[55] Gates, C. (2007). Access control requirements for web 2.0 security and privacy. *IEEE Web*, 2(0).

[56] Gavison, R. (1980). Privacy and the limits of law. *The Yale Law Journal*, 89(3):421–471.

[57] Goga, O., Lei, H., Parthasarathi, S. H. K., Friedland, G., Sommer, R., and Teixeira, R. (2013). Exploiting innocuous activity for correlating users across sites. In *Proceedings of the 22nd international conference on World Wide Web*, pages 447–458. International World Wide Web Conferences Steering Committee.

[58] Gong, N. Z., Talwalkar, A., Mackey, L., Huang, L., Shin, E. C. R., Stefanov, E., Song, D., et al. (2011). Jointly predicting links and inferring attributes using a social-attribute network (san). *arXiv preprint arXiv:1112.3265*.

[59] Grabowicz, P. A., Ramasco, J. J., and Eguíluz, V. M. (2013). Dynamics in online social networks. In *Dynamics On and Of Complex Networks, Volume 2*, pages 3–17. Springer.

[60] Graffi, K., Podrajanski, S., Mukherjee, P., Kovacevic, A., and Steinmetz, R. (2008). A distributed platform for multimedia communities. In *Multimedia, 2008. ISM 2008. Tenth IEEE International Symposium on*, pages 208–213. IEEE.

[62] Gurses, S. and Diaz, C. (2013). Two tales of privacy in online social networks. *Security & Privacy, IEEE*, 11(3):29–37.

[63] He, B.-Z., Chen, C.-M., Su, Y.-P., and Sun, H.-M. (2014). A defence scheme against identity theft attack based on multiple social networks. *Expert Systems with Applications*, 41(5):2345–2352.

[64] Herbener, J. M. (1997). The pareto rule and welfare economics. *The Review of Austrian Economics*, 10(1):79–106.

[67] Hu, H., Ahn, G.-J., and Jorgensen, J. (2013). Multiparty access control for online social networks: model and mechanisms. *Knowledge and Data Engineering, IEEE Transactions on*, 25(7):1614–1627.

[68] Hu, H., Ahn, G.-J., Zhao, Z., and Yang, D. (2014). Game theoretic analysis of multiparty access control in online social networks. In *Proceedings of the 19th ACM symposium on Access control models and technologies*, pages 93–102. ACM.

[69] Ioinson, A. N. and Paine, C. B. (2007). Self-disclosure, privacy and the internet. *The Oxford handbook of Internet psychology*, page 2374252.

[70] Jahid, S., Mittal, P., and Borisov, N. (2011). Easier: Encryption-based access control in social networks with efficient revocation. In *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*, pages 411–415. ACM.

[71] Jahid, S., Nilizadeh, S., Mittal, P., Borisov, N., and Kapadia, A. (2012). Decent: A decentralized architecture for enforcing privacy in online social networks. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2012 IEEE International Conference on*, pages 326–332. IEEE.

[72] Jin, L., Takabi, H., and Joshi, J. B. (2011). Towards active detection of identity clone attacks on online social networks. In *Proceedings of the first ACM conference on Data and application security and privacy*, pages 27–38. ACM.

[73] Jøsang, A., Ismail, R., and Boyd, C. (2007). A survey of trust and reputation systems for online service provision. *Decision support systems*, 43(2):618–644.

[75] Kayem, A. V., Akl, S. G., and Martin, P. (2010). *Adaptive cryptographic access control*, volume 48. Springer Science & Business Media.

[77] Kleineberg, K.-K. and Boguna, M. (2014). Trade-off between virality and mass media influence in the topological evolution of online social networks. *arXiv preprint arXiv:1403.1437*.

[78] Krivitsky, P. N., Handcock, M. S., Raftery, A. E., and Hoff, P. D. (2009). Representing degree distributions, clustering, and homophily in social networks with latent cluster random effects models. *Social networks*.

[79] Kruk, S. R., Grzonkowski, S., Gzella, A., Woroniecki, T., and Choi, H.-C. (2006). D-foaf: Distributed identity management with access rights delegation. In *The Semantic Web–ASWC 2006*, pages 140–154. Springer.

[81] Lam, X. N., Vu, T., Le, T. D., and Duong, A. D. (2008). Addressing cold-start problem in recommendation systems. In *Proceedings of the 2nd international conference on Ubiquitous information management and communication*, pages 208–211. ACM.

[82] Lamprecht, C. J. (2012). Adaptive security.

[83] Levin, D., Wundsam, A., Heller, B., Handigol, N., and Feldmann, A. (2012). Logically centralized?: state distribution trade-offs in software defined networks. In *Proceedings of the first workshop on Hot topics in software defined networks*, pages 1–6. ACM.

[84] Li, H., Zhao, B., and Fuxman, A. (2014). The wisdom of minority: discovering and targeting the right group of workers for crowdsourcing. In *Proceedings of the 23rd international conference on World wide web*, pages 165–176. International World Wide Web Conferences Steering Committee.

[85] Li, N., Qardaji, W. H., and Su, D. (2011). Provably private data anonymization: Or, k-anonymity meets differential privacy. *CoRR, abs/1101.2604*, 49:55.

[86] Lika, B., Kolomvatsos, K., and Hadjiefthymiades, S. (2014). Facing the cold start problem in recommender systems. *Expert Systems with Applications*, 41(4):2065–2073.

[87] Liu, G., Wang, Y., and Orgun, M. A. (2011). Trust transitivity in complex social networks. In *AAAI*, volume 11, pages 1222–1229.

[88] Liu, X., Lu, M., Ooi, B. C., Shen, Y., Wu, S., and Zhang, M. (2012). Cdas: a crowdsourcing data analytics system. *Proceedings of the VLDB Endowment*, 5(10):1040–1051.

[89] Low, Y., Bickson, D., Gonzalez, J., Guestrin, C., Kyrola, A., and Hellerstein, J. M. (2012). Distributed graphlab: a framework for machine learning and data mining in the cloud. *Proceedings of the VLDB Endowment*.

[90] Machanavajjhala, A., Kifer, D., Gehrke, J., and Venkitasubramaniam, M. (2007). l-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):3.

[92] Manolescu, D. (2011). Privacy: online vs. offline. In *E-Crime Expert: Data protection and privacy awareness*.

[93] Masoumzadeh, A. and Joshi, J. (2010). Osnac: An ontology-based access control model for social networking systems. In *Social Computing (SocialCom), 2010 IEEE Second International Conference on*, pages 751–759. IEEE.

[95] Mehregan, P. and Fong, P. W. (2014). Design patterns for multiple stakeholders in social computing. In *Data and Applications Security and Privacy XXVIII*, pages 163–178. Springer.

[96] Meyerhenke, H., Monien, B., and Sauerwald, T. (2008). A new diffusion-based multilevel algorithm for computing graph partitions of very high quality. In *Parallel and Distributed Processing, 2008. IPDPS 2008. IEEE International Symposium on*. IEEE.

[99] Moore, A. D. (2003). Privacy: its meaning and value. *American Philosophical Quarterly*, 40(3):215–227.

[101] Narayanan, A. and Shmatikov, V. (2010). Myths and fallacies of personally identifiable information. *Communications of the ACM*, 53(6):24–26.

[102] Narayanan, A., Toubiana, V., Barocas, S., Nissenbaum, H., and Boneh, D. (2012). A critical look at decentralized personal data architectures. *arXiv preprint arXiv:1202.4503*.

[103] Narendula, R., Papaioannou, T. G., and Aberer, K. (2012). A decentralized online social network with efficient user-driven replication. In *Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Confernece on Social Computing (SocialCom)*, pages 166–175. IEEE.

[104] Netter, M., Riesner, M., Weber, M., and Pernul, G. (2013). Privacy settings in online social networks–preferences, perception, and reality. In *System Sciences (HICSS), 2013 46th Hawaii International Conference on*, pages 3219–3228. IEEE.

[105] Newman, M. E. (2001). Clustering and preferential attachment in growing networks. *Physical Review E*, 64(2):025102.

[106] Nilizadeh, S., Jahid, S., Mittal, P., Borisov, N., and Kapadia, A. (2012). Cachet: a decentralized architecture for privacy preserving social networking with caching. In *Proceedings of the 8th international conference on Emerging networking experiments and technologies*, pages 337–348. ACM.

[107] Nissenbaum, H. (1998). Protecting privacy in an information age: The problem of privacy in public. *Law and philosophy*, 17(5):559–596.

[108] of Science, E. C. (2015). Probability theory and mathematical statistics.

[109] Pang, J. and Zhang, Y. (2015). A new access control scheme for facebook-style social networks. *Computers & Security*, 54:44–59.

[110] Paul, T., Famulari, A., and Strufe, T. (2014). A survey on decentralized online social networks. *Computer Networks*, 75:437–452.

[111] Rahimian, F., Girdzijauskas, S., and Haridi, S. (2014). Parallel community detection for cross-document coreference. In *Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2014 IEEE/WIC/ACM International Joint Conferences on*. IEEE.

[112] Roffo, G., Segalin, C., Vinciarelli, A., Murino, V., and Cristani, M. (2013). Reading between the turns: Statistical modeling for identity recognition and verification in chats. In *Advanced Video and Signal Based Surveillance (AVSS), 2013 10th IEEE International Conference on*, pages 99–104. IEEE.

[113] Sabelfeld, A. and Myers, A. C. (2003). Language-based information-flow security. *Selected Areas in Communications, IEEE Journal on*, 21(1):5–19.

[114] Saran, C. (2014). Tim Berners-Lee: Data sharing needs accountability.

[115] Schachter, E. P. (2004). Identity configurations: A new perspective on identity formation in contemporary society. *Journal of personality*, 72(1):167–200.

[118] Schoeman, F. D. (1992). *Privacy and social freedom*. Cambridge university press.

[119] Schwartz, S. J., Luyckx, K., and Vignoles, V. L. (2011). *Handbook of identity theory and research*. Springer.

[120] Schwittmann, L., Boelmann, C., Wander, M., and Weis, T. (2013). Sonet–privacy and replication in federated online social networks. In *Distributed Computing Systems Workshops (ICDCSW), 2013 IEEE 33rd International Conference on*, pages 51–57. IEEE.

[121] Seong, S.-W., Seo, J., Nasielski, M., Sengupta, D., Hangal, S., Teh, S. K., Chu, R., Dodson, B., and Lam, M. S. (2010). Prpl: a decentralized social networking infrastructure. In *Proceedings of the 1st ACM Workshop on Mobile Cloud Computing & Services: Social Networks and Beyond*, page 8. ACM.

[122] Shakimov, A., Lim, H., Ćaceres, R., Cox, L. P., Li, K., Liu, D., and Varshavsky, A. (2011). Vis-a-vis: Privacy-preserving online social networking via virtual individual servers. In *Communication Systems and Networks (COMSNETS), 2011 Third International Conference on*, pages 1–10. IEEE.

[123] Shakimov, A., Varshavsky, A., Cox, L. P., and Cáceres, R. (2009). Privacy, cost, and availability tradeoffs in decentralized osns. In *Proceedings of the 2nd ACM workshop on Online social networks*, pages 13–18. ACM.

[124] Shokri, R., Pedarsani, P., Theodorakopoulos, G., and Hubaux, J.-P. (2009). Preserving privacy in collaborative filtering through distributed aggregation of offline profiles. In *Proceedings of the third ACM conference on Recommender systems*, pages 157–164. ACM.

[125] Sicari, S., Rizzardi, A., Grieco, L. A., and Coen-Porisini, A. (2015). Security, privacy and trust in internet of things: The road ahead. *Computer Networks*, 76:146–164.

[126] Sirivianos, M., Kim, K., Gan, J. W., and Yang, X. (2014). Leveraging social feedback to verify online identity claims. *ACM Transactions on the Web (TWEB)*, 8(2):9.

[127] Snow, R., O'Connor, B., Jurafsky, D., and Ng, A. Y. (2008). Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *Proceedings of the conference on empirical methods in natural language processing*, pages 254–263. Association for Computational Linguistics.

[129] Soliman, A., Bahri, L., Carminati, B., Ferrari, E., and Girdzijauskas, S. (2015). Diva: Decentralized identity validation for social networks. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*, pages 383–391. ACM.

[Soliman et al.] Soliman, A., Bahri, L., Carminati, B., Ferrari, E., and Girdzijauskas, S. (2016). Cadiva - cooperative and adaptive decentralized identity validation model for social networks. *Social Network Analysis and Mining*, To appear.

[130] Squicciarini, A., Paci, F., and Sundareswaran, S. (2010a). Prima: an effective privacy protection mechanism for social networks. In *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*, pages 320–323. ACM.

[131] Squicciarini, A. C., Shehab, M., and Wede, J. (2010b). Privacy policies for shared content in social network sites. *The VLDB Journal—The International Journal on Very Large Data Bases*, 19(6):777–796.

[133] Such, J. M. and Rovatsos, M. (2014). Privacy policy negotiation in social media. *arXiv preprint arXiv:1412.5278*.

[135] Tipton, H. F. and Krause, M. (2003). *Information security management handbook*. CRC Press.

[136] Tootoonchian, A., Saroiu, S., Ganjali, Y., and Wolman, A. (2009). Lockr: better privacy for social networks. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies*, pages 169–180. ACM.

[139] Vijayarani, S., Tamilarasi, A., and Sampoorna, M. (2010). Analysis of privacy preserving k-anonymity methods and techniques. In *Communication and Computational Intelligence (INCOCCI), 2010 International Conference on*, pages 540–545. IEEE.

[140] Wang, J. and Ipeirotis, P. (2013). Quality-based pricing for crowdsourced workers.

[141] Warren, S. D. and Brandeis, L. D. (1890). The right to privacy. *Harvard law review*, pages 193–220.

[142] Weitzner, D. J., Abelson, H., Berners-Lee, T., Feigenbaum, J., Hendler, J., and Sussman, G. J. (2008). Information accountability. *Communications of the ACM*, 51(6):82–87.

[144] Yang, J. and Leskovec, J. (2015). Defining and evaluating network communities based on ground-truth. *Knowledge and Information Systems*, 42(1):181–213.

[145] Yeung, C.-m. A., Liccardi, I., Lu, K., Seneviratne, O., and Berners-Lee, T. (2009). Decentralization: The future of online social networking. In *W3C Workshop on the Future of Social Networking Position Papers*, volume 2, pages 2–7.

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

[147] Yu, H., Kaminsky, M., Gibbons, P. B., and Flaxman, A. (2006). Sybilguard: defending against sybil attacks via social networks. In *ACM SIGCOMM Computer Communication Review*. ACM.

[149] Zhang, D., Zou, Q., and Xiong, H. (2013). Cruc: Cold-start recommendations using collaborative filtering in internet of things. *arXiv preprint arXiv:1306.0165*.

Appendix A

BeatTheDiva: Game App

beatTheDiva is a game application that challenges a player to sophisticate her profile against the strategies suggested by DIVa for the validation of OSN identities in a community. Players of the game need to create a profile and try to convince other nodes in the system to befriend them. The nodes in the system deploy DIVa strategies to evaluate the validity of the player profile requesting their friendship.

beatTheDiva emulates real case scenarios where new profiles in an OSN attempt to establish new friendships and to build their trust in the network. In the scenario of a fake profile, there are two possible situations. First, the fake profile is targeting specific individuals by impersonating one of their friends (e.g., a fake account impersonates Alice who is a known friend to a target group of friends). In this case, the fake profile targets specific profiles and hopes to deceive them into accepting her friendship requests. In the second situation, the fake profile is trying to infiltrate within some community without targeting specific individuals. *beatTheDiva* can be applicable to impersonation attacks; however, the focus is more on the second type that deploy tailoring strategies to become members of some target community, hence gaining trust in the network. As such *beatTheDiva* allows players to create a profile and to gradually tailor it, against spending some score points, to infiltrate within some community. Players gain points for every node they succeed at befriending and are able to view her profile values. The game is won when the player establishes enough links to become part of a community. Community membership is determined based on the community detection algorithm as used in the DIVa model (crf. Chapter 4, Section 4.3). Before describing the game scenarios (see Section A.2), we first present its architecture and deployed strategies to challenge the players.

A.1 Game Engine Design

Figure A.1 shows the general architecture of *beatTheDiva*. The view controller manages the graphical changes on the interface in accordance with the screen manager that prepares the

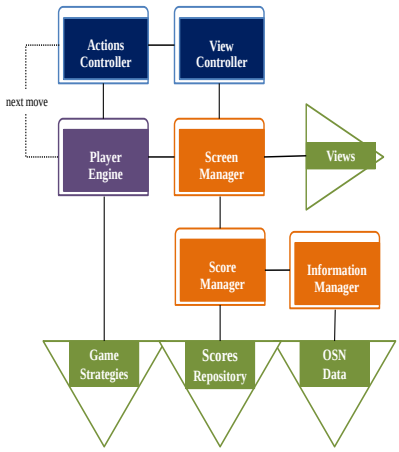


Figure A.1 *beatTheDiva* game engine architecture.

snapshot screens to be displayed in real time. The actions controller listens to the player's actions and communicates them to the player engine that returns the resulting game strategy to the screen manager. This latter collaborates with the score manager to update and display the score of the player. The information manager takes care of revealing more information about the OSN to the player depending on the earned score. In what follows, we detail the strategies deployed by the game engine.

Players create a profile, p_p, according to the schema S adopted in the system, and send friendship requests to chosen nodes in the network. We use the terms *player* and *player's profile* interchangeably. These requests are evaluated based on a simple strategy that accounts for three main factors: 1) *the CAS factor* that is the evaluation of the player against the DIVa CASes in the community of the target node; 2) *the player infiltration factor* that reflects the positioning of the player in the target community; 3) *the behavioral prior factor* that models the behavioral uncertainty that a user in real life would accept or deny a friendship request regardless of its rationality. We define these factors as follows:

Definition A.1.1 ***The CAS factor***. *Let $S = \{A_1, A_2, \cdots, A_k, \cdots, A_m\}$ be the profile schema in the system, where A_k is an attribute name. Let p_p be the profile of the player and p_t, member of community C_t, be the target node by an issued friendship request from p_p. Let F be the set of profiles of the friends of p_t. Let CAS_t be the collection of CASes defined by DIVa in community C_t. Let M_i be the set of profiles from F having the same profile values for $cas_i \in CAS_t$ as p_p. The CAS factor assigned to p_p w.r.t p_t is defined as: $CASF_p(t) = \frac{\sum_{\forall cas_i \in CAS_t} M_i}{|CAS_t|}$.*

To define the *infiltration factor*, we assume that every node v_i in the OSN has a known *influence factor* NI_i that is defined as its clustering coefficient. This latter is a graph based

measure that gives an indication on how the region surrounding a node is densely or sparsely connected.

Definition A.1.2 ***The infiltration factor.*** *Let p_p be the profile of the player and p_t, member of community C_t, be the target node by an issued friendship request from p_p. Let $LF \subset C_t$ be the set of nodes belonging to C_t that the player has successfully befriended. The infiltration factor of the player in community C_t is defined as: $I_p(C_t) = \sum_{\forall v_i \in LF} NI_i$.*

The behavioral prior factor reflects the probabilistic prevision that a node will accept a friendship request. We consider that the more friends a node has, the more prone to accept new friendships it is, especially if its friends are diverse. We base this assumption on a model in graph theory known as preferential attachment [105]. This one suggests that an old node creates an edge with a new one with a probability proportional to its degree. We assume that the average node degree in the input graph and the average community size are known, and we define the following:

Definition A.1.3 ***The behavioral prior factor.*** *Let ad be the average node degree and acs be the average community size in the network $G = (V, E)$. The behavioral prior factor of node $v_i \in V$ with degree d_i is defined as $bpf_i = \frac{d_i}{ad * acs}$.*

The game engine evaluates a friendship request based on Procedure EvaluateFriendRequest. A *strategy* is computed as the value of the CAS factor if the infiltration factor is zero (lines 1-2), or as the fraction of the CAS to the infiltration factors of the player w.r.t the target node (lines 3-4). The fraction is considered because the higher the infiltration factor, the more information the player is supposed to have on the community structure; thus, the higher the CAS factor should be. A CAS factor that is lower than the infiltration factor reflects bad choices of the player. The ratio of the CAS factor to the infiltration factor should outweigh the behavioral prior factor of the target node for the request to be accepted (line 5). If it is not, the request is denied (line 6)

A.2 Playing *beatTheDiva*

The goal of a player in *beatTheDiva* is to befriend enough friends from a community to become member of it. To achieve this, a player creates a profile, sends friendship requests, scores points for each accepted friendship request (evaluated based on Algorithm EvaluateFriendRequest), accesses profile information of befriended nodes, and uses score points to improve her profile presentation based on the network information she gains throughout the game, and/or to send more friendship requests.

Figure A.2 represents the general process of *beatTheDiva*. When starting the game, players can view nodes in the adopted OSN graph and some general profile information from

Procedure EvaluateFriendRequest($CASF_pt, I_pC_t, bpf_t$)

```
1  if I_p(C_t) = 0 then
2  |   strategy  = CASF_p(t) ;
3  else
4  |   strategy  = CASF_p(t) / I_p(C_t);
5  end
6  if strategy ≥ 1/bpf_t then
7  |   Send(accepted);
8  else
9  |   Send(accepted);
10 end
```

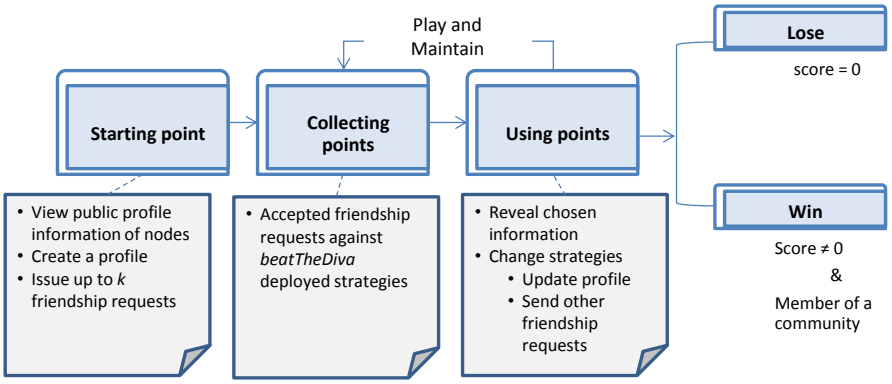
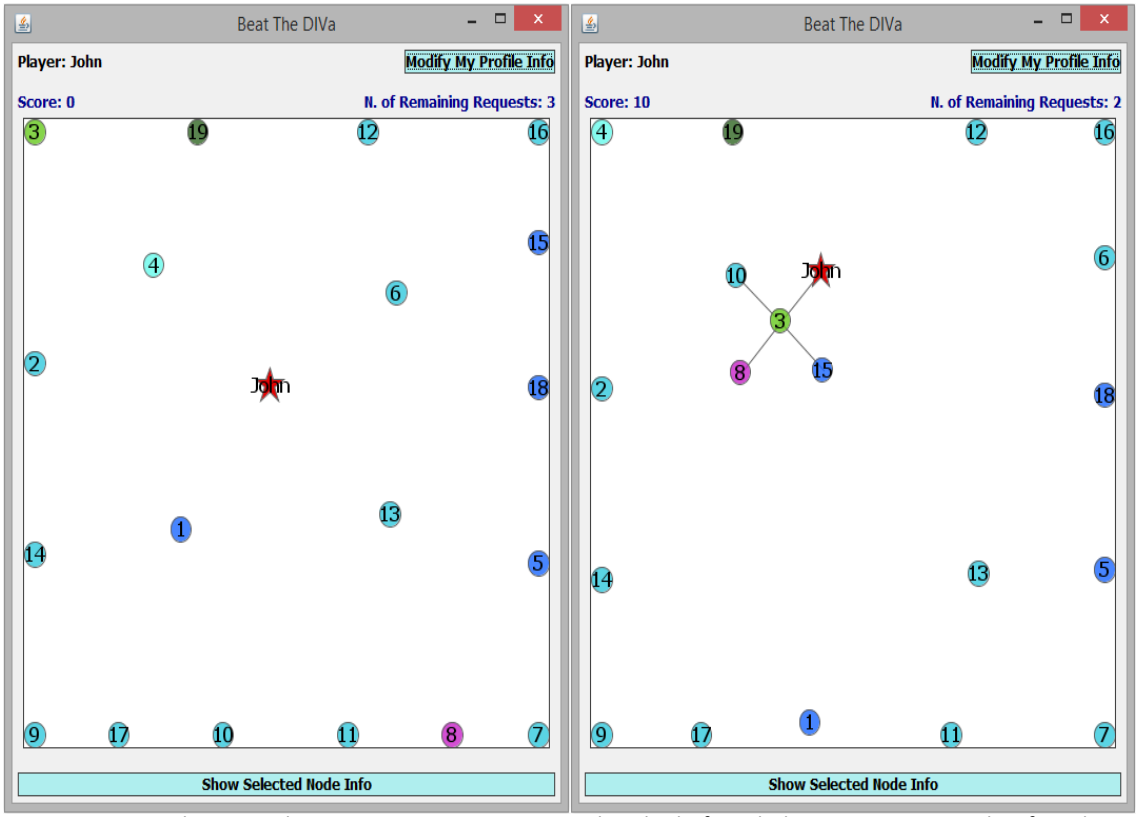


Figure A.2 *beatTheDiva* main playing process.

some nodes. This corresponds to the profile information that the nodes set as public in their privacy settings. Players create a profile in accordance with the profile schema adopted by the adopted OSN graph and can send up to k friendship requests, where k is a variable set within the application to limit the initial moves a player can perform. With every friendship request sent by the player and accepted by the game engine, the player earns score points that vary depending on the influence of the newly befriended node in the community and on the infiltration factor of the player's profile in that same community. The player can use these points to make updates to her profile, or to send new friendship requests. The player has access to all the profile information of the node(s) that have accepted its friendship requests. The player loses if the score falls down to a value zero and wins if successfully getting enough friends from a community to become member of it.

A summary of moves that a player can perform in the game is given in what follows:

- **Create a profile:** the player starts the game by creating a profile then views the game's main monitor as on Figure A.3.a. The nodes in the graph are displayed without links.
- **Send friendship requests:** the player can view public information of nodes and send up to k friendship requests (k=3 in the example in Figure A.3).
- **View response notifications:** the game notifies the player about the refused requests and draws edges between the player's node and the ones that accepted the friendship (see Figure A.3.b). The player's score is updated accordingly.
- **View befriended nodes information:** the player can access complete information about the befriended nodes.
- **Spend earned points:** the player can spend score points to send more friendship requests, to update her profile, or to view the influence factor of some chosen nodes.
- **View achieved scores and factors:** the game score is permanently displayed on the monitor. The player can click on their node to view information on achieved factors such as their infiltration factor within communities.



a. John starts the game

b. John befriended 3. He can now see her friends

Figure A.3 Game monitors of a starting round and after one node is befriended.

Appendix B

Detailed Theorems Proofs

B.1 COIP Model properties proofs

The COIP model specified four properties that it has to meet, as specified under Section 4.2.2 under Chapter 4. We theoretically prove that COIP meets all its four properties.

Assuring anonymity guarantee

We prove here that the k-anonymization of evaluation streams by CASTLE combined with privacy preserving raters selection ensures property P1, where A_g is a function of k. More precisely, we prove that $A_g = (1 - \frac{1}{k})$.

Proof. Let $u \in SN$ be a target user and let $Q^S(u)$ be her profile on the OSN with schema S. When $Q^S(u)$ goes through the system's evaluation process, it is first tupled. Every tuple is streamized and every stream is k-anonymized by CASTLE. Let $tuple^{\overline{CG}}(u)$ be a tuple of u for the correlated group $\overline{CG}$, and let t be its corresponding anonymization as output by CASTLE. From [28], it is proved that the anonymized streams generated by CASTLE are k-anonymized. We know that by k-anonymity, the anonymized tuple t is not differentiated from at least k other tuples. Hence, the probability for an attacker to link information in t to u is at most $\frac{1}{k}$. If we assume that a rater $r \in$ SN is selected to rate t and $\mathcal{B}_r(\overline{CG})$ is the probability that r identifies u from t , then $\mathrm{P}(\mathcal{B}_r(\overline{CG})) \leq \frac{1}{k}$. Therefore, the probability for r not to identify u from $tuple^{\overline{CG}}(u)$ is: $\mathrm{p}(\neg\mathcal{B}_r(\overline{CG})) \geq 1 - \frac{1}{k}$.

Because the system performs privacy preserving raters selection by which r is never selected to rate any two overlapping CGs for a user u, say $\overline{CG}$ and $\hat{\mathrm{C}}\mathrm{G}$, the events $\mathcal{B}_r(\overline{CG})$ and $\mathcal{B}_r(\hat{\mathrm{C}}\mathrm{G})$ are independent. Thus, the probability for r to not identify u, represented as $\mathrm{p}(\neg\mathcal{B}_r)$, is equal to the probability for r to not identify u on $\overline{CG}$. ■

Our anonymity guarantee relies on the assumptions made for the anonymization of evaluation streams and on the criteria put on the raters selection. Providing absolute guarantees on privacy might not be realistic given the inability to model all extrinsic factors that might

contribute to enabling de-anonymization, such as raters background knowledge. However, we believe that the privacy guarantee offered by the model provides practical assurance above the acceptable risks threshold.

Assuring system utility

Property P2 requires ensuring a minimum threshold, *utility*, of the system for all target users. The information loss threshold, τ, considered during the anonymization process and the rate precision factor applied to collected rates on anonymized tuples, ensure this property. We demonstrate how the system maintains a *utility* which is lower bounded by the worst rate precision factor.

Proof. Let $u \in SN$ be a target user and $Q^{\overline{CG}}(u)$ be her profile values for $\overline{CG} \in CG^*$. Let t be a tuple of u from $Q^{\overline{CG}}(u)$ and let us assume t went through generalization into its anonymized evaluation stream. Consider the worst case by which $\overline{CG}$ is the correlated group with the highest threshold for allowed information loss (i.e., $\tau_{\overline{CG}} \geq \tau_{cg}\ \forall cg \in CG^*$). Let us refer to this highest threshold by τ_{max}. Let us consider again the worst case by which tuple t was generalized up to τ_{max}. Let f_{max} be the average of all rates provided to t. Since t lost information in anonymization, f_{max} will be subject to a rate precision factor. By Definition 4.2.5, the precision factor is 1, if the information loss (i.e., τ_{max}) is less than $\frac{1}{e}$, or it is given by -ln(τ_{max}), otherwise. Let us go with worst case with τ_{max} greater than $\frac{1}{e}$. Hence, f_{max} is multiplied by $PrecFactor(\overline{CG}_u)$ = -ln(τ_{max}), to reflect the real value of the rate (the rate considering generalization of t). Let f_{ano} reflect the rate after its multiplied by the precision factor: $f_{ano} = f_{max}$ * $PrecFactor(\overline{CG}_u)$. Therefore, f_{max} - $f_{ano} = f_{max}$ * (1- $PrecFactor(\overline{CG}_u)$). We know that both f_{max} and $PrecFactor(\overline{CG}_u)$ are in [0,1], then, f_{max} * (1- $PrecFactor(\overline{CG}_u)$) $\leq$ (1- $PrecFactor(\overline{CG}_u)$). Therefore, f_{max} - $f_{ano} \leq$ (1- $PrecFactor(\overline{CG}_u)$)[**I**].

We know that the ITL is an aggregated average of all the average feedback across all correlated groups. Hence, $\frac{1}{|CG^*|} * \sum_{\forall cg \in CG^*} f_{max}(cg) = ITL_{max}$, where $f_{max}(cg)$ is the average of all feedback that u received on $Q^{cg}(u)$ without considering the precision factors. Assuming the worst case where all CGs allow the worst τ_{max} and where all the tuples of user u for all the CGs are generalized up to τ_{max}, the worst precision factor for all CG feedback is: $PrecFactor_{worse} = PrecFactor(\overline{CG}_u)$. Therefore, with reference to inequality [**I**], ITL_{max} - $ITL_{ano} \leq$ (1- $PrecFactor_{worse}$). Consequently, the system ensures a *utility* threshold of $PrecFactor_{worse}$ for u.

Since the precision factor is a function of information loss, the *utility* of the system can be tuned to the desired value by fixing a corresponding value of τ_{max}. ■

Guarantying operability

By property P3, operability is defined as ensuring timeliness in evaluating a user profile. This property is subject to two factors: (1) the delay needed for the tuples to get anonymized, and (2) the delay needed for the community to respond and rate the tuples. Factor (2) is bound to the basis of our system and is in fact its underlying assumption: the community is granted to be involved in the process. However, we believe that the way to ensure both participation of the community and timeliness in reacting is to provide incentives to raters for responding to evaluation requests. We assume that such a mechanism is in place and that the delay caused by factor (2) is meaningless. We then demonstrate that the delay caused by factor (1) is deterministic and is operated by the δ constraint on the evaluation stream corresponding to the CG with the highest support.

Proof. By Definition 4.2.3, the δ time constraint on an evaluation stream is directly proportional to the support of its CG. Let us assume the worst case with the CG with the highest support. Let us refer to the time constraint on its evaluation stream by δmax. By CASTLE, we know that a tuple t of user u with position $t.tp$ in this stream will be output at worst at time $t_{max} = t.tp + \delta_{max}$. Considering that $t.tp$ represents the logical time of the arrival of the profile of u to the system ($t_{start}(u)$), the maximum time that the tuple t can be delayed is δ_{max}. Therefore, $t_{end}(u)$ - $t_{start}(u) = \delta_{max}$, with $t_{end}(u)$ being the time at which the anonymized tuple t is available for rating. Since tuple t belongs to the stream with the worst δ, the system ensures that all tuples of u's profile are available for the ITL computation at most within a time interval of δ_{max}. ■

Guarantying impartiality

Property P4 requires ensuring impartiality in evaluations between similar tuples. This is assured by the similarity rating technique in the scenarios where it is applicable. That is, similar tuples are anonymized to the same generalization. We prove property P4 under the assumption of the similarity rating technique, and we discuss the implications when this technique is not applicable.

Proof. Let u and v be two users in the SN such that $Q^{\overline{CG}}(u) =_s Q^{\overline{CG}}(v)$ for a correlated group $\overline{CG} \in CG^*$, where $=_s$ is a similarity function.[1] `Insert` ý [1]We consider profiles similarity to be evaluated based on semantic similarity computation techniques, such as suggested in [4].

Assume that the time constraint on $\overline{CG}$'s evaluation stream is given by δ_x. Let us consider that u's and v's profiles are going through evaluation. The evaluation of their correlated tuples for $\overline{CG}$, $l = tuple^{\overline{CG}}(u)$ and $h = tuple^{\overline{CG}}(v)$, can happen under one of these two scenarios:

1) the positions of the two tuples in the stream are at a distance less than δ_x (i.e., $|l.tp$ - $h.tp| \leq \delta_x$). Given that $Q^{\overline{CG}}(u) =_s Q^{\overline{CG}}(v)$, the two tuples will in this case be output by CASTLE within the same anonymized output stream and with the same generalization. By the similarity rating technique, the two tuples will undergo the propagation of the same

averaged feedback, and hence: $|f(\overline{CG}_v) - f(\overline{CG}_u)| = 0$, where $f(\overline{CG}_v)$ and $f(\overline{CG}_u)$ are the average feedback associated with v and u on $\overline{CG}$ respectively. Therefore, property P4 is satisfied with a tolerance bias $\nabla_{max} = 0$.

2) the positions of the two tuples are more than δ_x distant. In this case, the tuple with the smallest position will be output first. Let us assume this is tuple l. In this case, l will know a generalization X to get anonymized and be associated with a corresponding average feedback $f(\overline{CG}_u)$ that, by similarity rating, will be saved for future tuples generalized with X. When tuple h is output, it is either anonymized with X or with some other generalization Y depending on the tuples it was clustered with. If tuple h is generalized with X, then, by similarity rating, it will get the same feedback as assigned to tuple l, and hence property P4 is satisfied with a tolerance bias $\nabla_{max} = 0$.

In contrast, if tuple h is generalized with Y and l with X, then property P4's tolerance bias cannot be deterministically quantified w.r.t tuples l and h as the difference between the rates provided to them is subjective to the available and selected raters for each one of them. Yet, property P4 can still be defended in this case from two complementary qualified arguments. First, assuming tuple h is generalized with Y and l with X and recalling that the initial assumption is that h and l are similar (i.e. they share a common parent on their generalization hierarchies), makes generalizations X and Y be from the same taxonomy tree. As such, the pools of available raters for selection by the values of X and Y would be consistently intersecting if not almost overlapping. This suggests that the rates provided to l and to h in that case would not be largely distant. The second argument comes from the underlying concept in crowd-sourced learning systems of the wisdom of the crowd by which averaged ratings from multiple random peers would converge to the same value regardless of the peers selected [88]. To sum up, property P4 is precisely proved with a tolerance bias, ∇_{max}, brought to 0 when two similar tuples l and h are anonymized to the same generalization. When these tuples are anonymized to different generalizations, property P4 can only be qualified to be holding to an acceptable tolerance bias that cannot be precisely quantified.■

B.2 DIVa model theorems proofs

DIVa model has two security properties formulated as Theorems 4.3.1 and 4.3.2. We provide their proofs as follows:

Proof to Theorem 4.3.1

Proof. Consider C_f of size z ($|C_f| = z$) is carrying a correlation $CAS_f = \{A, B\}$, that is unknown to the nodes in $\overline{C}$. Assume all C_f successfully joins $\overline{C}$. Therefore $\overline{C}.V = \overline{C}.V \cup C_f$ and $|\overline{C}.V| = n + z$. That is, the aggregate support of CAS_f in $\overline{C}$ would be: $support(CAS_f) =$

$\frac{values-co-occurrence(A,B)}{n+z}$. Since all nodes in C_f carry the correlation in CAS_f that is unknown to $\overline{C}$ initial n nodes, the support for CAS_f will be: $support(CAS_f) = \frac{z}{n+z}$. According to the proposed method, for CAS_f to be recognized as a CAS in $\overline{C}$ ($support(CAS_f) \geq sup_{lowest}$), this inequality shall hold: $z \geq \frac{sup_{lowest}}{(1-sup_{lowest})} * n$. ■

Proof to Theorem 4.3.2

Proof. Let $CAS_v = \{A, B\}$ be a valid CAS in $\overline{C}$ with aggregate support S_v: $S_v = \frac{m}{n} \geq sup_{lowest}$, where $m = values - co - occurrence(A, B)$. Let C_f of size z ($|C_f| = z$) be not carrying the correlation between attributes A and B. Assume all the nodes in C_f successfully join $\overline{C}$. Therefore $\overline{C}.V = \overline{C}.V \cup C_f$ and $|\overline{C}.V| = n + z$. That is, the aggregate support of CAS_v in $\overline{C}$ becomes: $S_{v1} = \frac{values-co-occurrence(A,B)}{n+z}$. Since all nodes in C_f do not carry the correlation in CAS_v, $values - co - occurrence(A, B)$ is still equal to m; therefore $S_{v1} = \frac{m}{n+z}$. For CAS_v to no more be a valid CAS, its new support shall be: $S_{v1} < sup_{lowest}$. That is, $\frac{m}{n+z} < sup_{lowest}$. From where $m < sup_{lowest} * (n + z)$. Dividing the inequality by n ($n \in \mathbb{N}^+ and n > 0$), we get: $\frac{m}{n} < \frac{n*sup_{lowest}+z*sup_{lowest}}{n}$. Therefore, dividing the inequality by the positive number sup_{lowest} gives, $\frac{S_v}{sup_{lowest}} < \frac{n+z}{n}$. From that, $z > \frac{S_v*n}{sup_{lowest}} - n$ ■

B.3 LAMPS model theorems proofs

The LAMPS model has a correctness property formulated by Theorem 5.2.1. We prove the Theorem by contradiction as follows:

Proof ($\Rightarrow$)**. (Algorithm 5 issues a granted message for $ir_{new} \Rightarrow SIR \cup \{ir_{new}\}$ is a correct state)**: By Theorem 5.2.1, given a current correct state, SIR, of the system, if ir_{new} is an IR for which Algorithm 5 issues a granted message, then $SIR_{new} = SIR \cup \{ir_{new}\}$ is a correct state. We prove this by contradiction. Given SIR the current state of the system, suppose that ir_{new} is a new IR that satisfies all security axioms. This means that one the following cases holds: (1) ir_{new} is a *read* IR that satisfies the FSP, and so the if statement of Algorithm 5 at line 2 is satisfied and Algorithm 5 returns a granted message (lines 2-6), (2) ir_{new} is a *write* IR that satisfies both the FSP and the WHP, and so the if statement of Algorithm 5 at line 9 is satisfied and Algorithm 5 returns a granted message (lines 9-13), (3) ir_{new} is a *share* IR that satisfies FSP and SHP, and so the if statement of Algorithm 5 at line 16 is satisfied and Algorithm 5 returns a granted message (lines 16-18), (4) ir_{new} is a *add-tag* IR that satisfies the FSP and the WHP, and so the if statement of Algorithm 5 at line 21 is satisfied and Algorithm 5 returns a granted message (lines 21-25), or (5) ir_{new} is a *add-comment* or *add-like* IR that satisfies FSP, and so the if statement of Algorithm 5 at line 28 is satisfied and Algorithm 5 returns a granted message (lines 28-31). Therefore ir_{new} satisfies all the method's axioms. Suppose now that $SIR_{new} = SIR \cup \{ir_{new}\}$ is not a correct state. This implies that $\exists ir_d \in SIR_{new}$ s.t., ir_d does not satisfy all the method's axioms. SIR

is a correct state; thus, $\forall ir \in SIR$, ir satisfies all the method's security axioms. Therefore, $ir_{new} \in SIR_{new}$ is the one that does not satisfy the method's axioms, then a contradiction arises. ■

Proof ($\Leftarrow$)**.** (**$SIR \cup \{ir_{new}\}$ is a correct state $\Rightarrow$ Algorithm 5 issues a granted message for** ir_{new}**)**: By Theorem 5.2.1, given a current correct state, SIR, of the system, if $SIR_{new} = SIR \cup \{ir_{new}\}$ is a correct state then ir_{new} is an IR for which Algorithm 5 issues a granted message. We prove this by contradiction. Assume $SIR_{new} = SIR \cup \{ir_{new}\}$ is a correct state. This implies that $\forall ir_i \in SIR_{new}$, ir_i satisfies all the method's axioms. Therefore, ir_i results in Algorithm 5 issuing a granted message. Now, assume that Algorithm 5 does not issue a granted message for ir_{new} and issues a denied message instead. That is, (1) ir_{new} is a *read* IR and the if statement of Algorithm 5 at line 2 is not satisfied, and hence ir_{new} does not satisfy the FSP (lines 7-8), (2) ir_{new} is a *write* IR and the if statement of Algorithm 5 at line 9 is not satisfied, and hence ir_{new} that does not satisfy both the FSP and the WHP or does not satisfy one of them (lines 14-15), (3) ir_{new} is a *share* IR and the if statement of Algorithm 5 at line 16 is not satisfied, and hence ir_{new} does not satisfy FSP and SHP or does not satisfy one of them (lines 19-20), (4) ir_{new} is a *add-tag* IR and the if statement of Algorithm 5 at line 21 is not satisfied, and hence ir_{new} does not satisfy the FSP and the WHP or does not satisfy one of them (lines 26-27), or (5) ir_{new} is a *add-comment* or *add-like* and the if statement of Algorithm 5 at line 28 is not satisfied, and hence ir_{new} does not satisfy FSP (lines 32-33). Therefore, ir_{new} does not satisfy the method's axioms. However, $ir_{new} \in SIR_{new}$, then a contradiction arises.■