# *Analysis, Modelling and Protection of Online Private Data*

A DISSERTATION PRESENTED
BY
SILVIA PUGLISI
TO
THE DEPARTMENT OF TELEMATICS ENGINEERING

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
IN THE SUBJECT OF
PRIVACY AND SECURITY


UNIVERSITAT POLITÈCNICA DE CATALUNYA (UPC)
BARCELONA, CATALUNYA
JUNE 2017

# *Analysis, Modelling and Protection of Online Private Data*


## Abstract

On-line communications generate a consistent amount of data flowing between users, services and applications. This information results from the interactions between different parties and once collected is used for a variety of purposes, from marketing profiling, to product recommendations, from news filtering to relationships suggestions.

Understanding how data is shared and used by services on behalf of users is the motivation behind this work. When a user creates a new account on a certain platform, this creates a logical container that will use to store the user's activity. The service aims at profiling the user, therefore every time some data is created, shared or accessed, information about the user behaviour and interests is collected and analysed. Users produce this data but are unaware of how this will be handled by the service, nor who it will be shared with. More importantly, once aggregated, these data could reveal more over time that the same users initially intended. Information revealed by one profile could be used either to obtain access to another account or during social engineering attacks.

The main focus of this dissertation is modelling and analysing how user data flows between different application and how this represent an important threat for privacy. A framework defining privacy violation is used to classify threats and identify issues where user data are effectively mishandled. Users data is modelled as categorised events, and aggregated as histograms of relative frequencies of on-line

activity along predefined categories of interests. Furthermore, a paradigm based on hypermedia to model on-line footprints is introduced. This emphasises the interactions between different user-generated events and their effects on the user's measured privacy risk. Finally, lesson learned from applying the paradigm to different scenarios are discussed.

# Contents

# Listing of figures

The only way to deal with an unfree world is to become so absolutely free that your very existence is an act of rebellion.

Albert Camus

# Acknowledgments

It has been a great pleasure , working these years with the faculty, staff, and students at the Universitat Politècnica de Catalunya · BarcelonaTech (UPC). This work would never have been possible if it were not for the freedom I was given to explore my own research interests.

This is thanks, in large part, to the kindness patience and mentoring provided by my advisor Prof. Jordi Forné and my co-advisor David Rebollo-Monedero.

A great deal of thanks is also reserved for Prof. Mónica Aguilar Igartua.

*It is poor civic hygiene to install technologies that could someday facilitate a police state.*

Bruce Schneier

# 1 Introduction

ONLINE COMMUNICATIONS are increasingly opening new possibilities for people to access and create content and interact with one another on the web. On the one hand web application facilitate access to information and foster relationships creation. On the other hand, as networking systems are constantly evolving, and online interactions are becoming more frequents and complex, it is becoming impossible to retain control over what is perceived as our online footprint. More specifically, user can share data with different services, that can subsequently share this information to third parties, sometimes without asking for permission to do so. Third parties are entitled to retain data over time, even if they have no direct connection with the user of the original service. Moreover, it has become a general practice to share content on different platforms and applications simultaneously. Such behaviour creates multiple possibilities for users to be potentially targets of

various attacks and different profiling activities.

Up to now, in an online context, the right to privacy has commonly been interpreted as a right to *information self-determination*. Acts typically claimed to breach online privacy concern the collection of personal information without consent, the selling of personal information and the further processing of that information. This definition of privacy breach can be considered valid until the user has direct control of the data they have created. This is not always the case. In 2011, the amount of digital information created and replicated globally exceeded 1.8 zettabytes (1.8 trillion gigabytes). 75% of this information is created by individuals through new media fora such as blogs and via social networks. By the end of 2011, Facebook had 845 million monthly active users, sharing over 30 billion pieces of content [**?** ]. Three quarters of the 1.8 trillion gigabytes of digital information online has been created by individual users. On top of that, an increasing amount of additional data about those users is collected by public and private companies, for the most disparate range of uses.

## 1.1 Motivation

This dissertation is motivated by understanding how data, created by users, flows between applications and services. A very powerful example in this field is the use of federated log in mechanisms. To register to a new social application, users grant them a certain level of access to their identity data, through, for example, their Facebook, Twitter or Google accounts. This data includes details about their identity, their whereabouts and in some situations even the company they work for. Third parties, like Facebook or Google, offer log in technologies, allowing the application to identify the user and receive precise information about them. Once the user grant access to their data, the application stores it and assumes control over how it is further shared. The user will never be notified again on who is accessing their data, nor if these are transferred to third parties.

## 1.2 Contribution

In summary, this dissertation makes the following contributions to research within the field of Information Privacy:

1. An analysis of how PETs affect recommendation systems for social tagging platforms.
2. An analysis of privacy risks for proximity based social applications.
3. An analysis of how users are tracked while surfing the web.
4. An information theoretic approach to measure the differential update of the anonymity risk for time variant users' profiles.
5. A hypermedia model of the user online footprint.

Furthermore Fig. 1.2.1 illustrates how the contributions listed are mapped to chapters of this thesis.

## 1.3 Related Publications

Most of the research results presented in this dissertation have been published in journals and conferences. In this section we provide a list of such publications, together with their complete bibliographic information. Further, we include other complementary articles that are not directly related with the research topic of this thesis, but which are especially significant from the state-of-the-art perspective.

### 1.3.1 Journal publications

1. Puglisi, S., Parra-Arnau, J., Forné, J. and Rebollo-Monedero, D., 2015. *On content-based recommendation and user privacy in social-tagging systems.* Computer Standards & Interfaces, 41, pp.17-27.
   https://doi.org/10.1016/j.csi.2015.01.004

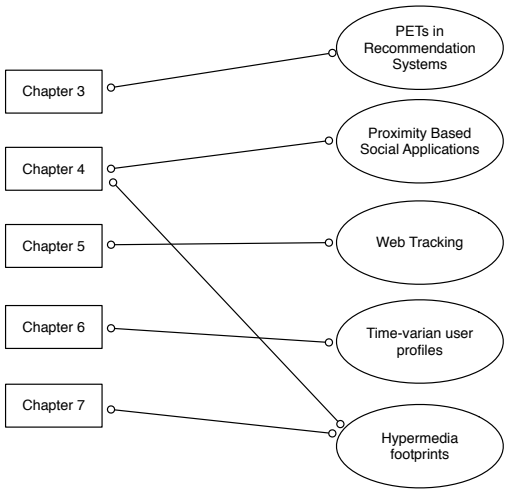



**Figure 1.2.1:** The following image illustrates how contributions are mapped to chapters.

2. Puglisi, S., Rebollo-Monedero, D. and Forné, J., 2017. *On web user tracking of browsing patterns for personalised advertising.* International Journal of Parallel, Emergent and Distributed Systems, pp.1-20. https://doi.org/10.1109/MedHocNet.2016.7528432

3. Puglisi, S., Rebollo-Monedero, D. and Forné, J., 2017. *On the Anonymity Risk of Time-Varying User Profiles.* Entropy, preprints https://doi.org/10.20944/preprints201704.0069.v1

### 1.3.2 Conference publications

1. Puglisi, S., Rebollo-Monedero, D. and Forné, J., 2015, August. *Potential mass surveillance and privacy violations in proximity-based social applications.* In Trustcom/BigDataSE/ISPA, 2015 IEEE (Vol. 1, pp. 1045-1052). IEEE. https://doi.org/10.1109/Trustcom-BigDataSe-ISPA.2015.481

2. Puglisi, S., Rebollo-Monedero, D. and Forné, J., 2015, September. *You Never Surf Alone. Ubiquitous Tracking of Users' Browsing Habits.* In International Workshop on Data Privacy Management (pp. 273-280). Springer International Publishing.
https://doi.org/10.1007/978-3-319-29883-2_20

3. Puglisi, S., Rebollo-Monedero, D. and Forné, J., 2016, June. *On Web user tracking: How third-party http requests track users' browsing patterns for personalised advertising.* In Ad Hoc Networking Workshop (Med-Hoc-Net), 2016 Mediterranean (pp. 1-6). IEEE.
https://doi.org/10.1109/MedHocNet.2016.7528432

Finally, we list the complementary publications.

1. Puglisi, S., 2015. *RESTful Rails Development: Building Open Applications and Services.* "O'Reilly Media, Inc.".

2. Puglisi, S., Moreira, Á.T., Torregrosa, G.M., Igartua, M.A. and Forné, J., 2016. *MobilitApp: Analysing mobility data of citizens in the metropolitan area of Barcelona.* In Internet of Things. IoT Infrastructures: Second International Summit, IoT 360° 2015, Rome, Italy, October 27-29, 2015. Revised Selected Papers, Part I (pp. 245-250). Springer International Publishing.
https://doi.org/10.1007/978-3-319-47063-4_23

## 1.4 Outline

The focus of this work is exploring the intersection between accurately modelling users' interactions and expressing private information in a way that it is possible to compute a numerical estimation of its impact for the user privacy. This first introduce this thesis and the outline of this work.

The second chapter presents a literature review of the problems considered throughout this work.

The third chapter introduces users' profile modelling and Privacy Enhancing Techniques in the field of social tagging systems. This chapter is particularly concerned with understanding how recommendations algorithms react to profile perturbation and how the utility of the algorithm is affected.

The fourth chapter is centred on how proximity-based social applications and the idea of serendipitous discovery of interests, places and social connections can be exploited by potential attackers. It is analysed how this services allow users to interact with people that are currently close to them, by revealing some information about their preferences and whereabouts. This information is acquired through passive geo-localisation and used to build a sense of serendipity. Unfortunately, while this class of applications opens different interactions possibilities for people in urban settings, obtaining access to certain identity information could lead a possible privacy attacker to identify and follow a user in their movements in a specific period of time. The same information shared through the platform could also help an attacker to link the victim's online profiles to physical identities. This chapter is also concerned with the possibilities presented by mobile devices to act as listening sensors and how these could eventually lead to newer privacy attacks.

The fifth chapter is focused on web tracking and how advertising networks are able *to follow* users while they surf the web. In this chapter it is highlighted the shift in the evolution of the Internet, from a stage when web sites were just hypertext documents, with no personalisation of the user experience offered, to the web of today, a world-wide distributed system following specific architectural paradigms. Nowadays, an enormous quantity of user-generated data is shared and consumed by a network of applications and services, reasoning upon users expressed preferences and their social and physical connections. Advertising networks follow users' browsing habits while they surf the web, continuously collecting their traces and surfing patterns. We analyse how users tracking happens on the web by measuring their online footprint and estimating how quickly advertising networks are able to profile users by their browsing habits.

In the sixth chapter it is explored how the user's profile change every time a user publishes a new post or creates a link with another entity, either another user, or

some online resource. When new information is added to the user profile, new private data is exposed. This doesn't only reveal information about single users' preferences, increasing their privacy risk, but can expose more about their network that single actors intended. This mechanism is self-evident on *social networks* where users receive suggestions based on their friends' activity. An information theoretic approach to measure the differential update of the anonymity risk for time variant users' profiles is proposed. This expresses how privacy is affected when new content is posted and how much third party services *get to know* about the users when a new activity is shared. We use real Facebook data to show how our model can be applied on a real world scenario.

In the seventh chapter it is presented a hypermedia model of the user online footprint. This model considers the architectural paradigms of the web and applies them to modelling of private information and especially on how this can be exchanged with a certain level of user control. We analyse the current models to grant access to private data and how this could be modified in order to achieve a better user supervision over their footprints. Furthermore, we analyse how this data could be applied with user willing to grant access to third-party apps to their Facebook profile in exchange for some service.

*Experience should teach us to be most on our guard to protect liberty when the government's purposes are beneficent. Men born to freedom are naturally alert to repel invasion of their liberty by evil-minded rulers. The greatest dangers to liberty lurk in insidious encroachment by men of zeal, well-meaning but without understanding.*

Louis D. Brandeis

# 2

# Background and Related Work

Privacy issues involve a plurality of complexities. The right to *privacy* is a concept that has evolved over human history and has enclosed different other rights. A few centuries ago having a right to privacy meant protecting property rights, along with life and cattle. This rights protected individuals from physical interference. At the end of the 19th century it was assumed that the common law needed to guarantee the right of deciding to what extent the thoughts, sentiments and emotions of an individual could be communicated to third parties [**?** ].

Nowadays privacy has acquired a completely different meaning because people conduct part of their existence through and on communication platforms. Privacy rights need to consider the implication of *information privacy*, given that a person shares parts of their activities, interests and even thoughts with online service providers. As a consequence, the philosophical definition of privacy has evolved,

while laws protecting individual privacy rights have tried to follow.

## 2.1 Privacy

The literature on privacy violation has struggled to agree on a definition of *Privacy* considered its elusive nature. Yet, the right to privacy is considered one of the most fundamental rights for modern democratic societies, which also includes freedom of thoughts, control over a person body, protection of reputation and from indiscriminate search, interrogation and surveillance, control over personal information and right to solitude. Article 12 of the Universal Declaration of Human Rights [**?** ], states that *"No one shall be subjected to arbitrary interference with his privacy"*. Article 18.4 of the Spanish constitution protects privacy and limits the use of information technology to safeguard personal intimacy of the citizens. In addition, the United States and a vast majority of of nations also protect privacy in their constitutions and in laws.

### 2.1.1 A taxonomy of privacy

Privacy violations involve a multitude of activities, some of these harmful others problematic. In fact, personal computers and more generally communication devices that are carried around by people are capable of being located, identified and tracked across different locations, networks and services [**?** ]. All these devices can therefore be used for a variety of surveillance activities, which are in itself detrimental to the user's interests. Until recently in fact, the cost of surveillance and tracking of people and activities was proportional to the cost of directly reaching, asking or following a single person or a group of people. Technology therefore enhances the surveillance capabilities by introducing tools that allow the collection of information arising from a person's activities. This information can furthermore be combined and inferred, therefore offering a more complete picture of that person. Daniel J. Solove in [**?** ] defines a taxonomy of privacy to classify violation and understand privacy issues in a comprehensive and concrete manner. Following this approach privacy violations are classified in four main categories (Table

2.1.1). These are:

1. Information collection
2. Information processing
3. Information dissemination
4. Invasion

Information collection:

Information collection results from activities such as surveillance, interrogation or information probing. It refers to actions aimed at watching, reading, listening, recording of individual activities or data about activities. It also refers to direct questioning of individuals or inference of information from data about them.

Information processing:

Information processing concerns the aggregation and identification of data. Failure to provide data security and the possibility for users to know who has accessed their data. This also includes secondary use of data to which the user has not been informed.

Information dissemination:

Information dissemination includes activity such as breach of confidentiality, unwanted disclosure and exposure of information. This also includes increased accessibility to individuals' information, appropriation and distortion of data about people. Information dissemination defines the very action of breaking the promise of keeping information confidential. It therefore implies actions aimed at the revelation of information about an individual that can change the image of that person within a group, including appropriation of identity information and dissemination of false or misleading facts.

Invasion:

Invasion is the threat of intrusion of an entity into someone private life and it includes acts that are said to disturb one tranquillity or solitude.

### 2.1.2 Identifying privacy violation on social networks and applications

The classification of privacy violations introduced suggests that users should be particularly careful with the information they share on social networks and applications. It has been show how leaking bits of personal information on one platform can be used for concrete privacy attacks. For example, physical identification and password recovery attacks can be based on the knowledge of personal information or the use of a known secret [**?** ]. It has been shown how the attribute set birth-date, gender, zip code poses concrete risks of individual identification [**?** ], leading to details that can be used to identify physical persons or to infer answers to password recovery questions.

Another important aspect to consider is that the average online user joins different social networks with the objective to enjoy distinct services and features. On each service or application an identity gets created, containing personal details, preferences, generated content and a network of relationships. The set of attributes used to describe these identities is often unique to the user. In addition, application or services sometimes require the disclosure of different personal information, such as email or full name, to create a profile. Users possessing different identities on different services, often use those to verify another identity on a particular application, i.e. a user will use their Facebook and LinkedIn profile to verify their account on the third service [**?** ]. A set of information required by one service could, in fact, add credibility to the information the user has provided for a second application, by demonstrating that certain personal details overlap, and by adding other information, like, for example, a set of shared social relationships.

The analysis of publicly available attributes in public profiles, shows a correla-

tion between the amount of information revealed in social network profiles, specific occupations or job titles and use of pseudonyms. It is possible to identify certain patterns regarding how and when users reveal precise information [**?** ]. Finally, aggregating this information can lead an attacker to obtain direct contact information by cross-linking the obtained features with other publicly available sources, such, for example, online phone directories.

A famous method for information correlation was presented by Alessandro Acquisti and Ralph Gross [**?** ]. Leveraging on the correlation between individuals' Social Security numbers and their birth data, they were able to infer people Social Security numbers by using only publicly available information.

Privacy attackers can also exploit loose privacy settings of a user's online social connections, taking advantage of how humans interpret messages and interact with one another [**?** ], developing semantic attacks [**?** ]. Therefore, mechanism helping to promote coordinated privacy policies could be more efficient to count attacks [**?** ].

Accurate coordinated policy could also warn users of which third party application they authorise to access their data. Social networking platforms, in fact, expose users' privacy to possible attacks by allowing third party application that access their data to be able to replicate it. Sandboxing techniques could be implemented allowing users to share information among social relationships, while also helping third party application to securely aggregate data according to differential privacy properties [**?** ].

Users should be allowed to choose an appropriate level of privacy for their needs and should be made aware of unwanted access to their data. This would permit protection of personal information that is being collected by mobile devices, including the derived inferences that could be drawn from the data. Semantic Web technologies can be implemented to specify high-level, declarative policies describing user information sharing preferences [**?** ].

A study on how users perceive the value of online and offline Personal Information (PI), shows that users value their PI related to their offline identities more (3 times) than what they wiling share online [**?** ]. This includes also valuing more

information related to their financial transactions and social network interactions than other online activities like search and shopping. Studies of this kind show how users are probably unknowingly sharing online more than they intended and how tracking technologies implements methods that collects user data without informing the users. In fact studies that have considered the users' perception of online advertising and the extent of online tracking have shown how the users' attitude generally changed when they found out that most of online advertising and therefore tracking activities happens without their consent [**?** ]

Users, in fact, consider three main deciding factors when consulted about how and to what extent they are willing to disclose personal and sensitive information, especially information about their location, to social relations [**?** ]. These factors were: who was requesting a particular information, why that information was requested, and what level of detail would be most useful to the requester.

This aspect of users' perception of sensitive information disclosure is particularly relevant when it has been shown [**?** ] that knowing a user location is used as a grounding mechanism in applications that lets users interact with their nearby. Geo-tagged information set the basis for a platform for honest and truthful signals in the process of forming new social relations.

At the same time, geo-localised information attached to users' activities can be used, by an attacker, to derive models of user mobility and provide data for context-aware applications and recommendation systems [**?** ]. This information can also be used to cluster communities with different preferences and interests into different geographical communities [**?** ].

Also, while some social networking applications use some form of obfuscation of the users' actual positions, precise location information can be still be derived. An attacker could use the partial information to identify a user's real position even when their exact coordinates are hidden or obfuscated by various location hiding techniques [**?** ].

While malicious attackers can target users, online services and platforms can also track their behaviour for a variety of purposes. Therefore, although there are certainly innumerable advantages in creating services that enable people to com-

municate so easily, it is as well important for users to retain control over which data they have been shared online over time. In the private sphere it has been said that "literally, Google knows more about us than we can remember ourselves." This situation has led to growing concerns regarding online privacy. In China, for example, one estimate suggests there are over 30.000 [**?** ] government censors monitoring online information.

In addition to user generated content, *"metadata"* regarding this content, are collected and stored by public and private organisations. Metadata are description of actual documents that can be easily read by a machine for a variety of uses, from searching and sorting to pattern recognition. This has lead in the last few years to the development of a new term to describe hyperlinked data objects: hyperdata. Hyperdata indicates data objects linked to other data objects in other places as hypertext indicates text linked to another text in other documents. Hyperdata enables the formation of a web of data, evolving from the "data on the web" that is not interrelated (or at least, not linked). Tools and information technology architectures employing visualisation and privacy enhancing technologies, become, therefore, central to help users maintain a desired online footprint and retain a certain level of control over their data. At the same time, these tools can be useful to developers as well, to be aware of the possible privacy and security implication of their work.

### 2.1.3 User profiling

With user profile we mean a container of an individual tastes, preferences and behaviour that can be used to predict future activities. A user's profile gives away the answer to whether or not that person can be interested on a certain product or service.

In recommendation systems employing tags or in any system allowing resource annotation, users decide to disclose personal data in order to receive, in exchange, a certain benefit. This earned value can be quantified in terms of the customised experience of a certain product [**?** ]. For such a recommendation system to work,

and successfully propose items of interest, user preferences need to be revealed and made accessible partially or in full, and thus exposed to possible privacy attacks.

When a user expresses and shares their interests by annotating a set of items, these resources and their categorisation will be part of their activity. The recorded users' activities will allow the used platform to "know more" about each of them, and therefore suggesting over time useful resources. These could be items similar to others tagged in the past, or simply close to the set of preferences expressed in their profile. In order to protect their privacy, a user could refrain from expressing their preferences altogether. While in this case an attacker would not be able to build a profile of the user in question, it would also become impossible for the service provider to deliver a personalised experience: the user would then achieve the maximum level of privacy protection, but also the worst level of utility.

Various and numerous approaches have been proposed to protect user privacy by also preserving the recommendation utility in the context of social tagging platform. These approaches can be grouped around four main strategies [**?** ]: encryption-based methods, approaches based on trusted third parties (TTPs), collaborative mechanisms and data-perturbative techniques. In traditional approaches to privacy, users or application designers decide whether certain sensitive information is to be disclosed or not. While the unavailability of this data, traditionally attained by means of access control or encryption, produces the highest level of privacy, it would also limit access to particular content or functionalities. This would be the case of a user freely annotating items on a social tagging platform. By adopting traditional PETs, the profile of this user could be made available only to the service providers, but kept completely or partially hidden from their network of social connections on the platform. This approach would indeed limit the chances of an attacker profiling the user, but would, unfortunately, prevent them from receiving content suggested by their community.

A conceptually simple approach to protecting user privacy consists in a TTP acting as an intermediary or *anonymiser* between the user and an untrusted information system. In this scenario, the system cannot know the user ID, but merely the identity of the TTP involved in the communication. Alternatively, the TTP

may act as a *pseudonymiser* by supplying a pseudonym ID' to the service provider, but only the TTP knows the correspondence between the pseudonym ID' and the actual user ID. In online social networks, the use of either approach would not be entirely feasible as users of these networks are required to authenticate to login. Although the adoption of TTPs in the manner described must, therefore, be ruled out, the users could provide a pseudonym at the sign-up process. In this regard, some sites have started offering social-networking services where users are not required to reveal their real identifiers. Social Number [**?** ] is an example of such networks, where users must choose a unique number as their ID.

Unfortunately, none of these approaches effectively prevents an attacker from profiling a user based on the annotated items content, and ultimately inferring their real identity. This could be accomplished in the case of a user posting related content across different platforms, making them vulnerable to techniques based on the ideas of re-identification. As an example, suppose that an observer has access to certain behavioural patterns of online activity associated with a user, who occasionally discloses their ID, possibly during interactions not involving sensitive data. The same user could attempt to hide under a pseudonym ID' to exchange information of confidential nature. Nevertheless, if the user exhibited similar behavioural patterns, the unlinkability between ID and ID' could be compromised through the exploitable similarity between these patterns. In this case, any past profiling inferences carried out by the pseudonym ID' would be linked to the actual user ID.

A particularly rich group of PETs resort to users collaborating to protect their privacy. One of the most popular is *Crowds* [**?** ], which assumes that a set of users wanting to browse the Web may collaborate to submit their requests. Precisely, a user wishing to send a request to a Web server selects first a member of the group at random, and then forwards the request to them. When this member receives the request, it flips a biased coin to determine whether to forward this request to another member or to submit it directly to the Web server. This process is repeated until the request is finally relayed to the intended destination. As a result of this probabilistic protocol, the Web server and any of the members forwarding the re-

quest cannot ascertain the identity of the actual sender, that is, the member who initiated the request.

We consider collaborative protocols [**?** **?** **?** ] like Crowds, not suitable for the applications addressed in this work although they may be effective in applications such as information retrieval and Web search. The main reason is that users are required to be logged into online social tagging platforms. That is, users participating in a collaborative protocol would need the credentials of their peers to log in, and post on their behalf, which in practice would be unacceptable. Besides, even if users were willing to share their credentials, this would not entirely avoid profiling based on the observation of the resources annotated.

In the case of perturbative methods for recommendation systems, [**?** ] proposes that users add random values to their ratings and then submit these perturbed ratings to the recommender. When the system has received these ratings, it executes an algorithm and sends the users some information that allows them to compute the final prediction themselves. When the number of participating users is sufficiently large, the authors find that user privacy is protected to some degree, and the system reaches an acceptable level of accuracy. However, even though a user may disguise all their ratings, merely showing interest in an individual item may be just as revealing as the score assigned to that item. For instance, a user rating a book called "How to Overcome Depression" indicates a clear interest in depression, regardless of the score assigned to this book. Apart from this critique, other works [**?** **?** ] stress that the use of certain *randomised* data-distortion techniques might not be able to preserve privacy completely in the long run.

In line with these two latter works, [**?** ] applies the same perturbative technique to collaborative filtering algorithms based on singular-value decomposition, focusing on the impact that their technique has on privacy. For this purpose, they use the privacy metric proposed by Agrawal, and Aggarwal, [**?** ], effectively a normalized version of the mutual information between the original and the perturbed data, and conduct some experiments with data sets from Movielens [**?** ] and Jester [**?** ]. The results show the trade-off curve between accuracy in recommendations and privacy. In particular, they measure accuracy as the mean absolute

error between the predicted values from the original ratings and the predictions obtained from the perturbed ratings.

The approach considered in this study follows the idea of perturbing the information implicitly or explicitly disclosed by the user. It, therefore, represents a possible alternative to hinder an attacker in their efforts to profile their activity precisely, when using a personalised service. The submission of false user data, together with genuine data, is an illustrative example of data-perturbative mechanism. In the context of information retrieval, query forgery [**?** **?** **?** **?** ] prevents privacy attackers from profiling users accurately based on the *content* of queries, without having to trust the service provider or the network operator, but obviously at the cost of traffic overhead. In this kind of mechanisms, the perturbation itself typically takes place on the user side. This means that users do not need to trust any external entity such as the recommender, the ISP or their neighbouring peers. Naturally, this does not signify that data perturbation cannot be used in combination with other third-party based approaches or mechanisms relying on user collaboration.

Certainly, the distortion of user profiles for privacy protection may be done not only by means of the insertion of false activity, but also by suppression. An example of this latter kind of data perturbation is the elimination of tags as a privacy-enhancing strategy [**?** **?** **?** **?** ], applied in the context of the semantic Web. This strategy allows users to preserve their privacy to a certain degree, but it comes at the cost of a degradation in the semantic functionality of the Web. Precisely, the the privacy-utility trade-off posed by the suppression of tags was investigated mathematically [**?** **?** **?** ], measuring privacy as the Shannon entropy of the perturbed profile, and utility as the percentage of tags users are willing to eliminate. Closely related to this are also other studies regarding the impact of suppressive PETs [**?** **?** **?** ], where the impact of tag suppression is assessed experimentally in the context of various applications and real-world scenarios.

While PETs to protect user profiles have been introduced and implemented we also believe that the privacy and sensitiveness of the information becoming accessible to third parties can be easily overlooked. The problem of measuring user

privacy in systems that profile users on the basis of the items they rate or tag is approached adopting a quantifiable measure of user privacy. Jaynes' rationale on maximum entropy methods [**?** **?** ] was used to measure the privacy of confidential data modeled by a probability distribution by means of its Shannon entropy and Kullbach-Lieber divergence [**?** **?** ]. This is particularly relevant when online services provide the users with the perception that sharing less data impact their optimal services experience.

## 2.2 Web tracking

Information regarding locations, browsing habits, communication records, health information, financial information, and general preferences regarding user online and offline activities are shared by different parties. This level of access is often directly granted from the user of such services. In a wide number of occasion though, private information is captured by online services without the direct user consent or even knowledge. We believe that the privacy and sensitiveness of the information becoming accessible to third parties can be easily overlooked.

To personalise their services or offer tailored advertising, web applications use tracking services that identify a user through different networks [**?** **?** ]. These tracking services usually combine information from different profiles that users create, for example their Gmail address or their Facebook or LinkedIn accounts. In addition, specific characteristics of the user's devices can be used to identify them through different sessions and websites, as described by the Panopticlick project [**?** ].

Browser fingerprinting is a technique implemented by analytics services and tracking technologies to identify uniquely a user while they browser different websites. Different features of a specific browser setup can be used to identify uniquely a user. Supported languages, browser extensions or installed fonts [**?** ] can be used to identify a browser setup among others. More advanced techniques distinguish between browsers' JavaScript execution characteristics [**?** ]. These features are particularly interesting since they are more difficult to simulate or mitigate in prac-

tice. Targeting JavaScript execution characteristics actually means looking at the innate performance signature of each browser's JavaScript engine, allowing the detection of browser version, operating system and micro-architecture. These attacks can also work in situations where traditional forms of system identification (such as the user-agent header) are modified or hidden. Other techniques exploit the whitelist mechanism of the popular NoScript Firefox extension.This mechanism allow the user to selectively enabling web pages' scripting privileges to increase privacy by allowing a site to determine if particular domains exist in a user's NoScript whitelist.

It is important to note that while tracking creates serious privacy concerns for Internet users, the customisation of results is also beneficial to the end user [**?** ]. In fact, while tailored services offer to the user only information relevant to their interests, it also allows some companies and institutions to concentrate an enormous amount of information about Internet users in general. [**?** ] investigate user profiling and access mechanisms offered by online data aggregator to users' collected data. Both the collected data and its accuracy was analysed together with the user's concerns. In their findings about 70% of the participants to the study expressed some concerns about the collection of sensitive data, its level of detail and how it might be used by third parties, especially for credit and health information.

Generally speaking, the activity of tracking a user across different websites, visits and devices, involves three main actors: the user, the tracking network, the list of websites visited. Every time a user visits a website a piece of code on the page is called asynchronously from the user's browser. When the call to the tracing network is performed a number of user data is transferred and used to profile the user at a later time and/or on a different website or device. By modelling the user behaviour as a directed graph, it is possible to uncover the underling network structure of the user footprint and the tracking networks tracing the user across the web [**?** ] [**?** ].

It has been shown how most successful tracking networks exhibit a consistent structure across markets, with a dominant connected component that, on average, includes 92.8% of network vertexes and 99.8% of the connecting edges [**?** ]. [**?** ]

have measured the chance that a user will become tracked by all top 10 trackers in approximately 30 clicks on search results to be of 99.5%. More interesting, [**?** ] have shown how tracking networks present properties of the small world networks. Therefore, implying a high-level global and local efficiency in spreading the user information and delivering targeted ads.

It is interesting to note that the behaviour of tracking networks follows that of telemarketing operations of the 80s and 90s. In [**?** ] the authors present an analysis of the history of telemarketing from cold calling potential customers on the phone, to the modern web tactics of tracking them across their browsing activities. It is particularly relevant how they point out that although users can try to avoid some modern communication tracking techniques, it is not guaranteed to assume that advertisers will respect individuals' choices and will not try to find alternative methods. In the past, technologies adopted to avoid sales calls were circumvented through clever new approaches by telemarketers. In 2010 in fact, the Wall Street Journal presented a series of articles on monitoring [**?** ], stating how the "nation's 50 top websites on average installed 64 pieces of tracking technology onto the computers of visitors, usually with no warning."

An interesting property of networks to understand their architecture is the behaviour of the average degree of nearest neighbours [**?** ] [**?** ]. The average degree of the nearest neighbours of a node $k_{nn}(k)$ is a quantity related to the correlations between the degree of connected vertices [**?** ], since it can be expressed as the conditional probability that a given vertex with degree $k$ is connected to a vertex of degree $k'$. This property defines if the network in consideration is assortative, if $k_{nn}$ is an increasing function of k or dissortative [**?** ] if it is not. The property of assortativity has been used in the field of epidemiology, to help understand how a disease or cure spreads across a network. It is particularly interesting to note that assortativity can give a measurement if the removal of a set of network's vertices may correspond in curing, vaccinating or quarantining individual cells in the network.

Another interesting aspect of networks is the presence of *communities*. A common activity when analysing large network is to start finding communities by di-

viding the nodes into *modules*. A common approach applies *generative models* able to infer the model parameters directly from the data. A simple generative process is the Stochastic Block Model (SBM) [**?** ]. A stochastic block model is able to explicitly describe the global structure of a network, providing a model of how the network can be partitioned into subgroups (blocks) and how the probability distribution of the connections between the nodes (i.e. probability that a node is connected to another) depends on the blocks to which the nodes belong [**?** ].

The microcanonical formulation [**?** ] of blockmodels takes as parameters the partition of the nodes into groups $b$ and a $B \times B$ matrix of edge counts $e$, where $e_{rs}$ is the number of edges between groups $r$ and $s$. Since edges are then placed randomly, nodes belonging to the same group possess the same probability of being connected with other nodes of the network. Furthermore, to be able to find small groups in large network nested SBM are used. With nested SBM groups are clustered into groups, and the matrix $e$ of edge counts are generated recursively from another SBM [**?** ]. Agglomerative multilevel Markov chain Monte Carlo (MCMC) algorithm as described in [**?** ], can be implied to compute a partition of the resulting graph.

Protection techniques against tracking networks are implemented through software agents able to identify if third-party requests are accessing private data. These agents include Privacy Badger [**?** ], Mozilla Lightbeam [**?** ], Ghostery [**?** ], AdBlock [**?** ], and so on. Some of these agents block certain JavaScript functions, or attempts to access determined browser functionality that can be used to uniquely identify the user. Some others implement a Tracking Protection Lists (TPL). A TLP can be seen as a blacklist of identified tracking domains that user might want to block.

Another interesting aspect of advertising services is how they are designed to work on feedback loops [**?** ]. An advertising service can in fact be seen as a black-box providing the tracker trying to identify or profile the user, and the returned advertising content. The tracker is used to send information back to the advertising service, which in response will return a certain content tailored to the user preferences. Within this feedback loop different aspect of the user behaviour are taken

in consideration. These include certainly the users browsing history and their click through rate, i.e. a measurement of the amount of time users in a population are more likely to interact with an ad. In more sophisticated advertising solution also user social connections are taken in consideration.

Advertising therefore services raise the problem of confidentiality of the user reading activity [**?** ]. Up to know an eloquent example of this problem was provided by the way public library in the US operates. Reading activities were considered historically private and were protected through a set of rules that restricted libraries ability to exploit reading records. This regime is clearly bypassed when libraries decide to provide digital services to their users. Digital services providers and third parties can in fact access users reading activities without agreeing to the library confidentiality regime.

## 2.3 ONLINE FOOTPRINTS

As users spend time online they produce private information across a multitude of services. These are web and mobile apps, websites, different platforms, social media, mobile and Internet of Things (IoTs) devices. Furthermore, data shared with one platform can be then shared with third-parties without the user having to consent again. The notion of secondary privacy diffusion was introduced to describe when user data are either deliberately transmitted or inadvertently leaked to a third-party [**?** ]. Example of secondary privacy diffusion in today's web are numerous. Imagine a scenario where a user is setting up their mobile phone for the first time. When they configure the device, all their data is transferred to various service providers. Among this data are also contact details of other people. Some of these people might have gone a long way trying to protect their details from disclosure, nor have they consented to their communications and information to be sent and stored by a third-party.

Different projects have tried to capture how services track users across websites, applications and devices, some of these are: Mozilla Lightbeam [**?** ], which allow users to visualise how web trackers are connected to the websites they visit,

Facebook-Tracking-Exposed [**?** ], a project aiming at increase transparency behind personalization algorithms and expose how Facebook filtering works, Data Selfie, a browser extension that tracks users on Facebook to show their data traces and reveal how machine learning algorithms the very same data to gain insights about their profile [**?** ].

Hyperdata represents the evolution of the web as we know now. When Tim Berners-Lee envisioned the semantic web in 2001 [**?** ], the web of data was described as a framework where autonomous agents could access structured information and conduct automated reasoning. These agents can be imagined as interconnected services accessing streams of data through a set of protocols or interfaces. APIs can provide such interfaces by specifying how software components can interact between each others through one or more protocols. When a request is sent to an individual service through an API, a stream of data is obtained as a reply. This reply is expressed in a format that can be parsed and interpreted. A hypermedia API would additionally specify links between the data object returned; therefore, a hypermedia browser would be able to explore such flow of information as web browser can navigate through the hyperlink in a web page.

Secondary data leakages are in reality a by-product of the way the web works. Data on the web is consumed in the form of objects, like documents, or simple snippets of data, linked to other objects. This objects are often referred to as hyperdata. Hyperadata can be easily explained by considering it as an evolution of the hypertext.Within a hypertext document in fact, paragraphs composing the document could be linked to some other text in the same or a different location. Hyperdata objects instead are either consumed through an Application Programming Interface (API), specifying how the different software components should interact with each other's or also embedded into existing document.

An example of hyperdata are markup standards like Microformats, Microdata and RDFa used by websites to embed structured data to describe product, services, events, and make user information available already into their HTML pages [**?** ].

A microformat (sometimes abbreviated $\mu F$) is an approach to describe data in a way that can be understandable both to machines and to humans. It builds on top of existing standards, and it is used to include metadata or other attributes into existing web pages or RSS feeds. This way software agents can process information that would otherwise be readable only for humans, such as contact information, geographical coordinates, or calendar events.

When hyperdata objects are explored through an API, this would probably implement different communication protocols to allow several technologies to access independently to hyperdata objects. To enable this exchange of information among heterogeneous systems, the API can implement a language-neutral message format to communicate. This could be the case of XML or JSON languages, used as containers for the exchanged messages. In this extent, an 'Hypermedia API' is one that is designed to be accessed and explored by any device or application. Its architecture is hence similar to the structure of the web and the same reasoning when serving and consuming the API it is applied.

The response data for any API call can be returned in a desired format. Most RESTful services return either XML or JSON, while some give the options to choose a preferred format. The format is defined either in the request header or the URI called. It is also possible to set the default format that is returned unless another format is specified.

JSON stands for JavaScript Object Notation, and it is defined as a lightweight data-interchange format. It has been based on a subset of the JavaScript Programming Language, Standard ECMA262 3rd Edition December 1999. JSON is a language to exchange data, so it is defined as language independent format and easy to be read by humans as well as being parsed by programs. A JSON object is a collection of name/value pairs, like a dictionary data structure in python or a hash in ruby. An object begins with { (left brace) and ends with } (right brace). Each name is followed by : (colon) and the name/value pairs are separated by , (comma). JSON also supports ordered lists. These can be seen as a list of values, as in an array. An array begins with [ (left bracket) and ends with ] (right bracket). Values

are separated by , (comma). A value can be a string in double quotes, or a number, or true or false or null, or an object or an array. These structures can be nested (Table: 2.3.1).

XML stands for Extensible Markup Language, and it is designed as a language to define a set of rule to encode documents in a format that is both human-readable and machine-readable. It is defined in the XML 1.0 Specification produced by the W3C. XML was created to structure, store, and transport information, so this is why it is so handy and straightforward to use for application to communicate between each other's. With XML, it is possible to define the tags, attributes and nesting rules that make a document valid according to a particular document type definition (DTD) or XML schema (XSD, XML Schema Definition), according to the application-specific choices. A DTD is a set of markup declarations that define a document type, while an XML schema express a set of rules to which an XML document must conform in order to be considered 'valid' according to that schema (Table: 2.3.2).

To protect data collected by third-parties and preserve the confidentiality of users' footprints a privacy framework around the concept of "virtual walls" was proposed in [**?** ]. A virtual wall extends the notion of real world privacy provided by a closed room, sheltering a person from the outside world. A virtual wall would be a set of user specified policies controlling access to all their personal data in a way that is as intuitive and consistent with their notion of physical privacy.

A common problem for user footprints protection tools has been identified in the user attitude towards disclosing new information and their awareness, or lack-of-there-of, regarding possible data leakage. These aspects are amplified by the economics of web services based on advertising. It has been shown though, that an efficient client-side tool that maximizes users' awareness over their online footprint can help users making informed decisions over how they disclose new data [**?** ].

Different approaches for data management have also been proposed using cryptographic techniques. *Anonrep* [**?** ] is an anonymous reputation system where users anonymously post messages and tag them with their reputation score, without revealing other sensitive information. AnonRep reliably tallies other users' feedback (e.g., likes or votes) without leaking the user identity or the exact reputation score, and also maintaining a level of security against duplicate feedback and score tampering. Smart contracts based on the concept of decentralised cryptocurrencies can facilitate data transactions and service management between individuals, applications and devices. In the field of smart contracts, Hashcash [**? ?** ] was probably the first of such systems. Hashcash propose a CPU cost-function to compute a token that can be used as a proof-of-work. This concept introduced by Hashcash, together with previous ideas from other systems as e-cash and b-money, create the basis for a cryptocurrency. Bitcoin [**?** ] uses and expands these ideas to define a cryptographically secure mechanism to reach consensus over a series of cryptographically signed financial transactions. Bitcoin can be considered the first decentralised transaction ledger. Bitcoin itself has been forked several times and different version of the crypto-coin have been created introducing a number of variations over the protocols used [**?** ] [**?** ]. Other projects instead re-purpose core paradigms of Bitcoin to different applications and domains.

The Ethereum project builds upon previous work on the usage of a cryptographic proof of computational expenditure as a means of transmitting a value signal over the Internet [**?** ]. in Ethereum the Bitcoin ledger is considered as a state transition system. The current state in Bitcon is the collection of all unspent transaction outputs (UTXO) with each UTXO having a denomination and an owner (defined by an address of a given length which can be considered as a cryptographic public key). A state transition function takes the current state and a transaction as inputs, and the new resulting state as output. This is similar to the standard banking system where the state is the balance sheet, a transaction is a request to move a sum of money $X$ from A to B, and the state transition function is the mechanism reducing the vale in A's account by $X$ and incrementing the value in B's account by $X$. Moreover, UTXO in Bitcoin can be owned not just by a public key but also by

a more complicated script. Scripts in Bitcoin are expressed through a stack-based programming language allowing simple operations. With this paradigm a transaction spending in UTXO must provide data satisfying the script. Likewise, the basic public key ownership mechanism of Bitcoin is implemented via scripts. In this case the script take an elliptic curve signature as input, verifies it against the transactions and address owning the UTXO and return 1 for success and 0 otherwise. More complicated scripts can be created for different purposes, allowing a decentralised cross-cryptocurrency exchange. Bitcoin scripting capabilities are however quite limited. The lack of Turing completeness and different states are a drawback to build more complex applications on top of the Bitcoin paradigm. Ethereum provide a blockchain with a turing-complete programming language. A computer program that runs on the blockchain is a contract. It consists of program code, storage file and account balance. A contract is created by posting a transaction to the blockchain. Once created the program code of a contract is fixed, and its code executed whenever it receives a message, either from a user or from another contract. This concept has been used to define decentralised autonomous organisation and trust [**?** ] [**?** ].

In the field of the Internet of Things (IoTs) a number of techniques have been proposed. An interesting research effort in anonymous authentication systems is EPID [**?** ]. EPID is technology for active anonymity aiming at solving the problems of authentication, anonymity and revocation with finite field arithmetic and elliptic curve cryptography (ECC). In the EPID ecosystem three entities are defined: the authority responsible for generating, signing and revoking keys, the platform device receiving a service, the verifier that provides the service to the device. EPID provides a solution for a device to authenticate itself anonymously to a service provider. The defined protocol is one-way because the service provider is not authenticating back to the platform.

An extension of EPID, ChainAnchor [**?** ], uses the blockchain as a mechanism to anonymously register device commissioning and decommissioning.ChainAnchor provides a privacy-preserving technique for device commissioning and assurance to service providers that the device is a genuine product issued by the manufac-

turer. Another blockchain-based approach proposes a combination of blockchain and off-blockchain storage instead. This combination is used to construct a privacy-focused personal data management platform [**?** ]. With a decentralised approach, users are not required to blindly trust any third-party and are always aware of how their data is being managed and used. In addition, the blockchain recognises data ownership to the user, and not to the company providing the service.

The blockchain has also be used to extend the GPG approach to the *web of trust* [**?** **?** ], providing an alternative certificate format based on Bitcoin which allows a user to verify a PGP certificate using Bitcoin identity-verification transactions. The user will be able to form first degree trust relationships that are tied to actual values. Furthermore, the blockchain approach can also be used to design a novel distributed PGP key server and store and retrieve, to and from the ledger, Bitcoin-Based PGP certificates.

Certcoin is a Public key infrastructures (PKIs) with no central authority [**?** ] leveraging the consistency guarantees provided by cryptocurrencies such as Bitcoin and Namecoin to build a PKI that ensures identity retention, effectively preventing one user from registering a public key under another's already-registered identity.

Other digital identities management techniques have been built on top of common cross-site authentication schemes such as OAuth and OpenID. An example of such approach is Crypto-Book [**?** ] an approach which extends existing digital identities through public-key cryptography and ring signatures. A similar technique is proposed by UnlimitID [**?** ] a method for enhancing the privacy of common mechanisms for authorisation and authentication, such as OAuth.

**Table 2.1.1:** Classification of privacy violations

| **Violation** | **Activities** | **Actions** |
|---|---|---|
| Collection | - Surveillance;<br>- Information probing;<br>- Interrogation. | - Watching, listening, recording of individuals' activities.<br>- Questioning individuals directly.<br>- Inferring information from data. |
| Processing | - Aggregation;<br>- Identification;<br>- Insecurity;<br>- Secondary use;<br>- Exclusion. | - Gathering of data about individuals.<br>- Identification of physical identities from online data.<br>- Carelessness in protecting data.<br>- Failure in allowing users to know who has accessed to their data. |
| Dissemination | - Breach of confidentiality;<br>- Disclosure;<br>- Exposure;<br>- Increased accessibility;<br>- Data appropriation;<br>- Distortion. | - Breaking the promise of keeping the information confidential.<br>- Revelation of information about an individual that impacts the way other see them.<br>- Appropriation of identity information.<br>- Dissemination of false or misleading information.<br>- Transfer of personal data to third party or threat to do so. |
| Invasion | - Intrusion of someone private life. | - Acts that can disturb one tranquillity or solitude. |

The table summarises the classification used to categorise privacy violation in proximity-based social application.

```
{
”products”: [
{ ”type”:”Sneakers” , ”brand”:”Adidas” }, { ”type”:”Runners” , ”brand”:”Nike” },
{ ”type”:”Accessories” , ”brand”:”Puma” }
]
}
```

**Table 2.3.1:** A JSON example

```
< product >
< type > Sneakers < /type >
< brand > Adidas < /brand >
< /product >
```

**Table 2.3.2:** An XML example

*I live on Earth at present, and I don't know what I am. I know that I am not a category. I am not a thing — a noun. I seem to be a verb, an evolutionary process – an integral function of the universe.*

R. Buckminster Fuller

# 3

# Users profiling in social tagging systems

Recommendation systems and content filtering approaches based on annotations and ratings, essentially rely on users expressing their preferences and interests through their actions, in order to provide personalised content. This activity, in which users engage collectively has been named social tagging, and it is one of the most popular in which users engage online, and although it has opened new possibilities for application interoperability on the semantic web, it is also posing new privacy threats. It, in fact, consists of describing online or offline resources by using free-text labels (i.e. tags), therefore exposing the user profile and activity to privacy attacks. Users, as a result, may wish to adopt a privacy-enhancing strategy in order not to reveal their interests completely.

In this chapter we investigate the impact of PETs on comment recommendation systems extending results from [**?** ]. Tag forgery is a privacy enhancing technology

consisting of generating tags for categories or resources that do not reflect the user's actual preferences. By modifying their profile, tag forgery may have a negative impact on the quality of the recommendation system, thus protecting user privacy to a certain extent but at the expenses of utility loss. The impact of tag forgery on content-based recommendation is, therefore, investigated in a real-world application scenario where different forgery strategies are evaluated, and the consequent loss in utility is measured and compared.

## 3.1 Background

Recommendation and information filtering systems have been developed to predict users' preferences, and eventually use the resulting predictions for a variety of services, from search engines to resources suggestions and advertisement. The system functionality relies on users implicitly or explicitly revealing their activity and personal preferences, which are ultimately used to generate personalised recommendations.

Such annotation activity has been called *social tagging* and it consists of users collectively assigning keywords (i.e. *tags*) to real life objects and web-based resources that they find interesting. Social tagging is currently one of the most popular online activities. Therefore, different functionalities have been implemented in various online services, such as Twitter [**?** ], Facebook [**?** ], YouTube [**?** ], and Instagram [**?** ], to encourage their users to tag resources collectively.

Tagging involves classifying resources according to one own experience. Unlike traditional methods where classification happens by choosing labels from a controlled vocabulary, in social tagging systems users freely choose and combine terms. This is usually referred to as free-form tag annotation, and the resulting emergent information organisation has been called *folksonomy*.

This scenario has opened new possibilities for semantic interoperability in web applications. Tags, in fact, allow autonomous agents to categorise web resources easily, obtaining some form of semantic representation of their content. However, annotating online resources poses potential privacy risks, since users reveal

their preferences, interests and activities. They may then wish to adopt privacy-enhancing strategies, masquerading their real interests to a certain extent, by applying tags to categories or resources that do not reflect their actual preferences. Specifically, *Tag forgery* is a privacy enhancing technology (PET) designed to protect user privacy, by creating bogus tags in order to disguise real user's interests. As a perturbation-based mechanism, tag forgery poses an inherent trade-off between privacy and usability. Users are able to obtain a high level of protection by increasing their forgery activity, but this can substantially affect the quality of the recommendation.

The primary goal of this work is to investigate the effects of tag forgery to content-based recommendation in a real-world application scenario, studying the interplay between the degree of privacy and the potential degradation of the quality of the recommendation. An experimental evaluation is performed on a dataset extracted from Delicious [**?** ], a social bookmarking platform for web resources. In particular, three different tag forgery strategies have been evaluated, namely: *optimised tag forgery* [**?** ], *uniform tag forgery* and *TrackMeNot* (TMN) [**?** ], the last consists of simulating a possible TMN like agent, periodically issuing randomised tags according to popular categories.

Using the dataset and a measure of utility for the recommendation system, a threefold experiment is conducted to evaluate how the application of tag forgery may affect the quality of the recommender. Hence, we simulate a scenario in which users only apply one of the different tag forgery strategies considered. Measures of the recommender performances are computed before and after the application of each PET, obtaining an experimental study of the compromise between privacy and utility.

To the best of our knowledge, this is the first systematic evaluation of the impact of applying perturbation-based privacy technologies on the usability of content-based recommendation systems. For this evaluation, both suitable privacy and usability metrics are required. In particular, as suggested by Parra et al. [**?** ], the KL divergence is used as privacy metric of the user profile; while the quality of the recommendation is computed following the methodology proposed by Cantador

el al. [**?** ].

In this chapter we first describe the adversary model considered §3.1. Following, we explain a possible practical application for the proposed PET through the implementation of a communication module §3.2. Therefore, we discuss the evaluation methodology and obtained results §3.3.

sectionAdversary Model Users tagging online and offline resources generate what is has been called a folksonomy, that is, a set composed by all the users that have expressed at least a tag, the tags that have been used and the items that have been described through them. Formally, a folksonomy $\mathcal{F}$ can be defined as a tuple $\mathcal{F} = \{\mathcal{T}, \mathcal{U}, \mathcal{I}, \mathcal{A}\}$, where $\mathcal{T} = \{t_1, \ldots, t_L\}$ is the set of tags, or more generally tag categories, which comprise the vocabulary expressed by the folksonomy; $\mathcal{U} = \{u_1, \ldots, u_M\}$ is the set of users that have expressed at least a tag; $\mathcal{I} = \{i_1, \ldots, i_N\}$ is the set of items that have been tagged; and $\mathcal{A} = \{(u_m, t_l, i_n) \in \mathcal{U} \times \mathcal{T} \times \mathcal{I}\}$ is the set of annotations of each tag category $t_l$ to an item $i_n$ by a user $u_m$ [**?** ].

As we shall see in §3.1.1, our user-profile model will rely on categorising tags into categories of interest. This will provide a certain mathematical tractability of the user profile while at the same time allowing for a classification of the user interests into macro semantic topics.

In our scenario, users assign tags to online resources, according to their preferences, taste or needs. It follows that while the user is contributing to categorise a specific content through their tags, hence adding semantic information to the whole folksonomy, their activity is revealing something regarding their interests, reducing their privacy overall.

We assume that the set of potential privacy attackers includes any entity capable of capturing the information users convey to a social tagging platform. Accordingly, both service providers and network operators are deemed potential attackers. However, since tags are often publicly available to other users of the tagging platform, any entity able to collect this information is also taken into consideration in our adversary model.

In our model, we suppose that the privacy attacker aims at profiling users through their expressed preferences, specifically on the basis of the tags posted. Through-

out this work, we shall consider that the objective of this profiling activity is to *individuate* users, meaning that the attacker wishes to find users whose preferences significantly diverge from the interests of the whole population of users. This assumption is in line with other works in the literature [**? ? ?** ].

### 3.1.1 Modelling the User/Item Profiles

A tractable model of the user profile as a probability mass function (PMF) is proposed in [**? ? ? ?** ] to express how each tag contributes to expose how many times the user has expressed a preference toward a specific category of interest. This model follows the intuitive assumption that a particular category is weighted according to the number of times this has been used in the user or item profile.

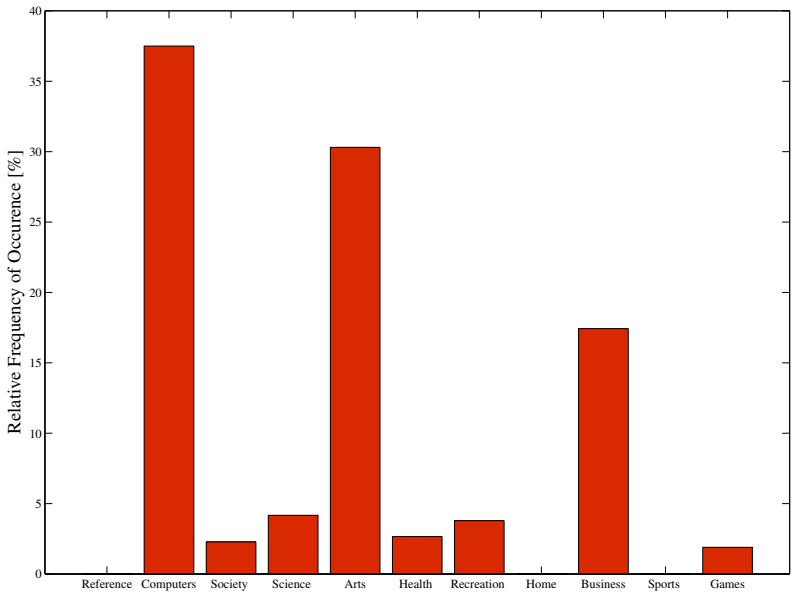



**Figure 3.1.1:** Example of a user's profile expressed as a PMF across a set of tag categories.

Exactly as in those works, we define the profile of a user $u_m$ as the PMF $p_m$ =

$(p_{m,1}, \ldots, p_{m,L})$, conceptually a histogram of relative frequencies of tags across the set of tag categories $\mathcal{T}$. More formally, in terms of the notation introduced at the beginning of Section 3.1, the $l$-th component of such profile is defined as

$$p_{m,l} = \frac{|\{(u_m, t_l, i) \in \mathcal{A} | i \in \mathcal{I}\}|}{|\{(u_m, t, i) \in \mathcal{A} | t \in \mathcal{T}, i \in \mathcal{I}\}|}.$$

Similarly, we define the profile of an item $i_n$ as the PMF $q_n = (q_{n,1}, \ldots, q_{n,L})$, where $q_{n,l}$ is the percentage of tags belonging to the category $l$ which have been assigned to this item. Both user and item profiles can then be seen as normalised histograms of tags across categories of interest. Our profile model is in this extent equivalent to the tag clouds that numerous collaborative tagging services use to visualise which tags are being posted, collaboratively or individually by each user. A tag cloud, similarly to a histogram, is a visual representation in which tags are weighted according to their relevance. Fig. 3.1.1 shows an example of a user's profile.

In view of the assumptions described in the previous section, our privacy attacker boils down to an entity that aims to profile users by representing their interests in the form of normalised histograms, on the basis of a given categorisation. To achieve this objective, the attacker exploits the tags that users communicate to social tagging systems. This work assumes that users are willing to submit false tags, to mitigate the risk of profiling. In doing so, users gain some privacy, although at the cost of certain loss in usability. As a result of this, the attacker observes a perturbed version of the genuine user profile, also in the form of a relative histogram, which does not reflect the actual interests of the user. In short, the attacker believes that the observed behaviour characterises the actual user's profile.

Thereafter, we shall refer to these two profiles as the *actual* user profile $p$ and the *apparent* user profile $t$.

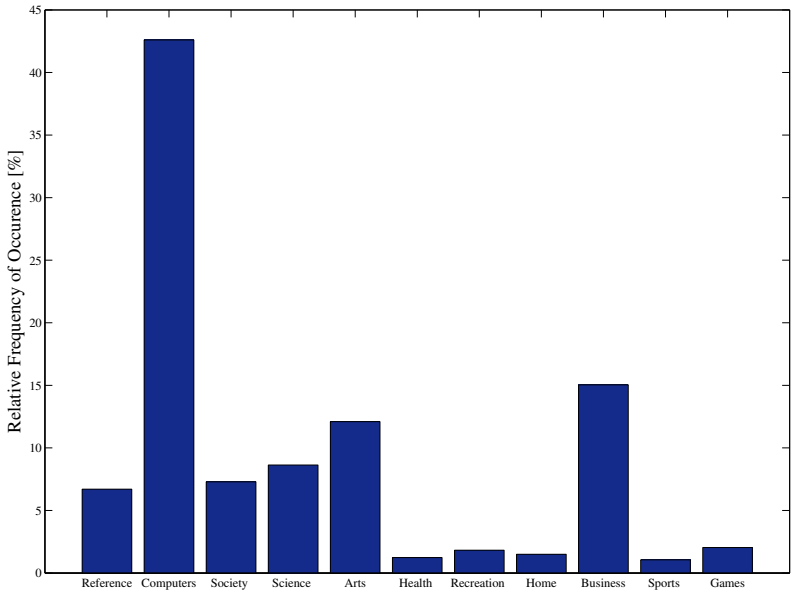



**Figure 3.1.2:** Profile of the whole population of users in our dataset.

### 3.1.2 Privacy Metric

In this section, we propose and justify an information-theoretic quantity as a measure of user privacy in social tagging systems. For the readers not familiar with information theory, next we briefly review two key concepts.

Recall [**?** ] that Shannon's entropy $\mathrm{H}(p)$ of a discrete random variable (r.v.) with PMF $p = (p_i)_{i=1}^{L}$ on the alphabet $\{1, \ldots, L\}$ is a measure of the uncertainty of the outcome of this r.v., defined as

$$\mathrm{H}(p) = -\sum p_i \log p_i.$$

Given two probability distributions $p$ and $q$ over the same alphabet, the Kullback-Leibler (KL) divergence is defined as

$$\mathrm{D}(p \,\|\, q) = \sum p_i \log \frac{p_i}{q_i}.$$

The KL divergence is often referred to as *relative entropy*, as it may be regarded as a generalisation of the Shannon entropy of a distribution, relative to another.

Having reviewed the concepts of entropy and relative entropy, we define the *initial privacy risk* as the KL divergence between the user's genuine profile $p$ and the population's tag distribution $\bar{p}$, that is,

$$\mathcal{R}_\circ = \mathrm{D}(p \,\|\, \bar{p}).$$

Similarly, we define the *(final) privacy risk* $\mathcal{R}$ as the KL divergence between the user's apparent profile $t$ and the population's distribution,

$$\mathcal{R} = \mathrm{D}(t \,\|\, \bar{p}).$$

Next, we justify the Shannon entropy and the KL divergence as measures of privacy when an attacker aims to individuate users based on their tag profiles. The rationale behind the use of these two information-theoretic quantities as privacy metrics is documented in greater detail in [**?** ].

Leveraging on a celebrated information-theoretic rationale by Jaynes [**?** ], the Shannon entropy of an apparent user profile may be regarded as a measure of privacy, or more accurately, anonymity. The leading idea is that the method of types from information theory establishes an approximate monotonic relationship between the likelihood of a PMF in a stochastic system and its entropy. Loosely speaking and in our context, the higher the entropy of a profile, the more likely it is, and the more users behave according to it. Under this interpretation, entropy is a measure of anonymity, although not in the sense that the user's identity remains unknown. Entropy has, therefore, the meaning that the higher likelihood of an apparent profile can help the user go unnoticed. In fact, the apparent user profile makes the user more typical to an external observer, and hopefully, less attractive to an attacker whose objective is to target peculiar users.

If an aggregated histogram of the population is available as a reference profile, as we assume in this work, the extension of Jaynes' argument to relative entropy

also gives an acceptable measure of anonymity. The KL divergence is a measure of discrepancy between probability distributions, which includes Shannon's entropy as the particular case when the reference distribution is uniform. Conceptually, a lower KL divergence hides discrepancies with respect to a reference profile, say the population's profile. Also, it exists a monotonic relationship between the likelihood of a distribution and its divergence with respect to the reference distribution of choice. This aspect enables us to deem KL divergence as a measure of anonymity in a sense entirely analogous to the above mentioned.

Under this interpretation, the KL divergence is, therefore, interpreted as an (inverse) indicator of the commonness of similar profiles in said population. As such, we should hasten to stress that the KL divergence is a measure of anonymity rather than privacy. The obfuscated information is the uniqueness of the profile behind the online activity, rather than the actual profile. Indeed, a profile of interests already matching the population's would not require perturbation.

### 3.1.3 Privacy-Enhancing Techniques

Among a variety of PETs, this work focuses on those technologies that rely on the principle of *tag forgery*. The key strengths of such tag-perturbation technique are its simplicity in terms of infrastructure requirements and its strong privacy guarantees, as users need not trust the social tagging platform, nor the network operator nor other peers.

In conceptual terms, tag forgery is a PET that may help users tagging online resources to protect their privacy. It consists of the simple idea that users may be willing to tag items that are unknown to them and that do not reflect their actual preferences, in order to appear as similar as possible to the average population profile. A simple example of such technique can be illustrated by thinking to a specific thematic community, such that of a group of individuals interested in jazz music. In this scenario if a user is particularly interested in rock music, their profile could be easily spotted and identified, as they would probably express interest towards artists and tracks that could be categorised outside of the jazz category.

When a user wishes to apply tag forgery, first they must specify a *tag-forgery rate* $\rho \in [0, 1]$. This rate represents the ratio of forged tags to total tags the user is disposed to submit. Based on this parameter and exactly as in [**?** ], we define the user's apparent tag profile as the convex combination $t = (1 - \rho)\, p + \rho\, r$. Here $r$ is some *forgery strategy* modeling the percentage of tags that the user should forge in each tag category. Clearly, any forgery strategy must satisfy that $r_i \geqslant 0$ for all $i$ and that $\sum r_i = \rho$.

In this work, we consider three different forgery strategies, which result in three implementations of tag forgery, namely, optimised tag forgery [**?** ], the popular TMN mechanism [**?** ] and a uniform tag forgery. The optimised tag forgery corresponds to choosing the strategy $r^*$ that minimises privacy risk for a given $\rho$, that is,

$$r^* = \arg\min_r \mathrm{D}((1 - \rho)\, p + \rho\, r \,\|\, \bar{p}).$$

Please note that this formulation of optimised tag forgery relies on the appropriateness of the criteria optimised, which in turn depends on a number of factors. These are: the specific application scenario and the tag statistics of the users; the actual network and processing over-head incurred by introducing forged tags; the assumption that the tag-forgery rate $\rho$ is a faithful representation of the degradation in recommendation quality; the adversarial model and the mechanisms against privacy contemplated.

The TMN mechanism is described next. Said mechanism is a software implementation of query forgery developed as a Web browser add-on. It exploits the idea of generating false queries to a search engine in order to avoid user profiling from the latter. TMN is designed as a client-side software, specifically a browser add-on, independent from centralised infrastructure or third-party services for its operation. In the client software, a mechanism defined dynamic query lists has been implemented. Each instance of TMN is programmed to create an initial seed list of query terms that will be used to compute the first flow of decoys searches. The initial list of keywords is built from a set of RSS feeds from popular websites, mainly news sites, and it is combined with a second list of popular query words

gathered from recently searched terms. When TMN is first enabled, and the user sends an actual search query, TMN intercepts the HTTP response returned from the search engine, and extracts suitable query-like terms that will be used to create the forged searches. Furthermore, the provided list of RSS feeds is queried randomly to substitute keywords in the list of seeds [**?** ].

Because TMN sends arbitrary keywords as search queries, the user profile resulting from this forgery strategy is completely random [**?** ]. Although the user possess the ability to add or remove RSS feeds that the extension will use to construct their bogus queries, there is no possible way to control which actual keywords are chosen. Moreover, the user has no control on the random keywords that are included in the bursts of bogus queries, since these are extracted from the HTTP response received from the actual searches that the user has performed. While TMN is a technique designed to forge *search queries*, we have implemented a TMN-like agent generating bogus *tags*. To initialise our TMN-like agent we have considered an initial list of seed using RSS feeds from popular news sites, the sites included were the same ones that TMN uses in its built-in list of feeds. By querying the RSS feeds, a list of keywords was extracted. Hence, using the extracted keywords a distribution of tags into eleven categories was constructed, these eleven categories corresponds to the first taxonomy levels of the Open Directory Project (ODP) classification scheme [**?** ]. The profile obtained with this technique has then been assumed as a reference to implement a TMN agent and is denoted by the distribution $w$.

Last but not least, the proposed uniform tag forgery strategy is constructed similarly to TMN. We have in fact supposed a TMN agent that would send disguise tags created according to a uniform distribution across all categories. More specifically, in the uniform forgery strategy we have that $r = u$. Table 3.1.1 summarises the tag-forgery strategies considered here.

**Table 3.1.1:** Summary of the tag-forgery strategies under study. In this work, we investigate three variations of a data-perturbative mechanism that consists of annotating false tags. The optimised tag forgery implementation corresponds to the strategy that minimises the privacy risk for a given forgery rate. The TMN-like approach generates false tags according to the popular privacy-preserving mechanism TrackMeNot [**?** ]. The uniform approach considers the uniform distribution as forgery strategy.

| Tag-forgery implementation | Forgery strategy $r$ |
|---|---|
| Optimised [**?** ] | $\arg\min_r \mathrm{D}((1-\rho)\,p + \rho\, r \,\|\, \bar{p})$ |
| TMN [**?** ] | $w$ (TMN distribution) |
| Uniform | $u$ (uniform distribution) |

### 3.1.4 Similarity Metric

A recommender, or a recommendation system, can be described as an information filtering system that seeks to predict the rating or preference that a user would give to an item. For the purpose of our study, the idea of rating a resource or expressing a preference has been considered as the action of tagging an item. This assumption follows the idea that a user will most likely tag a resource if they happen to be interested in this resource.

In the field of recommendations systems, we may distinguish three main approaches to item recommendation: content-based, user-based and collaborative filtering [**?** ]. In content-based filtering items are compared based on a measure of *similarity*. The assumption behind this strategy is that items similar to those a user has already tagged in the past would be considered more relevant by the individual in question. If in fact a user has been tagging resources in certain categories with more frequency, it is more probable that they would also annotate items belonging to the same categories.

In user-based filtering, users are compared with other users based again on a defined measure of similarity. It is supposed, in this case, that if two or more users have similar interests, i.e. they have been expressing preference in resources in sim-

ilar categories, items that are useful for one of them can also be significant for the others.

Collaborative filtering employs both a combination of the techniques described before as well as the collective actions of a group or network of users and their social relationships [**?** ]. In collaborative filtering then, not only the tags and categories that have been attached to certain items are considered, but also what are called item-specific metadata are taken into account, these could be the item title or summary, or other content-related information [**?** ].

In the coming sections, we shall use a generic content-based filtering algorithm [**?** ] to evaluate the three variations of tag forgery described in §3.1.3.

We have chosen a content-based recommender because this class of algorithms models users and items as histograms of tags, which is essentially the model assumed for our adversary (§3.1.1). Loosely speaking a content-based recommendation system is composed of: a proper technique for representing the items and users' profiles, a strategy to compare items and users and produce a recommendation. The field of content recommendation is particularly vast and developed in the literature and its applications are numerous. Recommendation systems in fact span different topics in computer science, information retrieval and artificial intelligence.

For the scope of this job we are only concentrating on applying a suitable measure of similarity within items and users' profiles. The recommendation algorithm we have implemented therefore aims to find items that are closer to a particular user profile (i.e. more similar). Three commons measurement of similarity between objects are usually considered in the literature. These are namely: Euclidean distance, Pearson correlation and Cosine similarity [**?** ].

The Euclidean distance is the simplest and most common example of a distance measure. The Pearson correlation is instead a measurement of the linear relationship between objects. While there are certainly different correlation coefficients that have been considered and applied, the Pearson correlation is among the most commonly used.

Cosine similarity is another very common approach. It considers items as docu-

ment vectors of an n-dimensional space and compute their similarity as the cosine of the angle that they form. We have applied this approach in our study.

More specifically, we have considered a cosine-based similarity [**?** ] as a measure of distance between a user profile and an item profile. The cosine metric is a simple and robust measure of similarity between vectors which is widely used in content-based recommenders. Hence if $p_m = (p_{m,1}, \ldots, p_{m,L})$ is the profile of user $u_m$ and $q_n = (q_{n,1}, \ldots, q_{n,L})$ is the profile of item $i_n$, the cosine similarity between these two profiles is defined as

$$s(p_m, q_n) = \frac{\sum_l p_{m,l}\, q_{n,l}}{\sqrt{\sum_l p_{m,l}^2}\sqrt{\sum_l q_{n,l}^2}}.$$

### 3.1.5 Utility Metric

A utility metric is being introduced in order to evaluate the performances of the recommender and understand how these degrade with the application of a specific PET. Prediction accuracy is among the most debated property in the literature regarding recommendation systems. For the purpose of this work it is assumed that a system providing on average more accurate recommendation of items would be preferred by the user. Furthermore the system is evaluated considering a content retrieval scenario where a user is provided with a ranked list of N recommended items, hence performances are evaluated in terms of ranking based metrics used in the Information Retrieval field of study [**?** ] . The performance metric adopted is therefore among the most commonly used for ranked list prediction, i.e. precision at top V results. In the field of information retrieval, precision can be defined as the fraction of recommended items that are relevant for a target user [**?** ] . If the recommendation system evaluated retrieves V items, the previously defined ratio is precision at top V or P@V. Precision at top V is then a metric that measures how many relevant documents the user will find in the ranked list of results. The overall performance value is then calculated by averaging the results over the set of all available users. Considering a likely scenario, for which a user would be presented with a list of top-$V$ results that the system has considered most similar to their pro-

file, we have evaluated precision of the recommender in two possible situations: with $V = 30$ in one case and $V = 50$ in the other.

## 3.2 Architecture

In this section, we present an architecture of a communication module for the protection of user profiles in social tagging systems (Fig. 3.2.1). We consider the case in which a user would retrieve items from a social tagging platform, and would occasionally submit annotations in the form of ratings or tags to the resource they would find interesting. This would be the case of a user browsing resources on StumbleUpon, tagging bookmarks on Delicious or exploring photos on Flickr. The social tagging platform would suggest web resources through its recommendation system that would gradually learn about the user interest, hence trying to suggest items more related to the user expressed preferences.

While the user would normally read the suggested documents, these would also be intercepted by the communication module, running as a software on the user space. This can be imagined as a browser extension analysing the communication between the user and the social tagging platform under consideration.

More generally, the communication module can be envisioned as a proxy or a firewall, i.e. a component between the user and the outside Internet, responsible for filtering and managing the communication flows that the user generates. While the user would browse the Internet the communication module would be in sleeping mode, and it could be turned on at the user's discretion only when visiting certain social tagging platforms. It is assumed that while the user would surf a certain platform, eventually annotating resources that they find relevant, they would receive and generate a stream of data, or more specifically a data flow. This is composed of the resources that the platform is sending to the user in the form of recommendation and of those that the user is sending back to the platform in the form of tagged items.

These data flows are analysed in the communication module by a component, the population profile constructor, and used to build a population profile of ref-

erence. We have supposed that these data streams would probably contain annotations that would help the module profiling the average population of users, together with other information regarding trends and current news. It is also possible that the module would contain specific, pre-compiled profiles, corresponding to particular population that the user would consider either safe or generic.

The user generated stream of data instead, composed by each annotated item, would be feeding the user profile constructor. This component would keep track of the actual expressed user preferences and feed this data into the forgery controller.

At this point the forgery controller would calculate a forgery strategy, that at the user discretion is either applied or not to the stream of tagged resources, and that would be sent to the social tagging platform, as the flow of data comprising the user activity. If the user kept the communication module on its off state, no forgery would modify the documents sent to the social tagging service, otherwise a certain stream of annotations would be computed and applied to certain resources.

This means that according to the strategy and a forgery rate that the user has chosen, the forgery controller would produce a number of bogus tags to certain items. These would be sent to the social tagging platform together with the actual user annotations. The user would hence present to the platform not their real profile, but an apparent profile $t$ resulting from both their real activity and the forged categorisation stream.

### 3.2.1 Further considerations

We would like to stress the fact that at the centre of our approach is the user. The communication module can in fact be used either to calculate a forgery strategy, or to simply warn the user when their privacy risk reaches a certain threshold. At this point the user would be presented with a possible forgery strategy and eventually are set of keywords and resources that could be used to produce bogus tags. We are aware that a mechanism generating tags could eventually produce a strategy introducing sensible topics in the user profile. We have, therefore, addressed this situation by using exclusively a curated list of websites and news portals whose

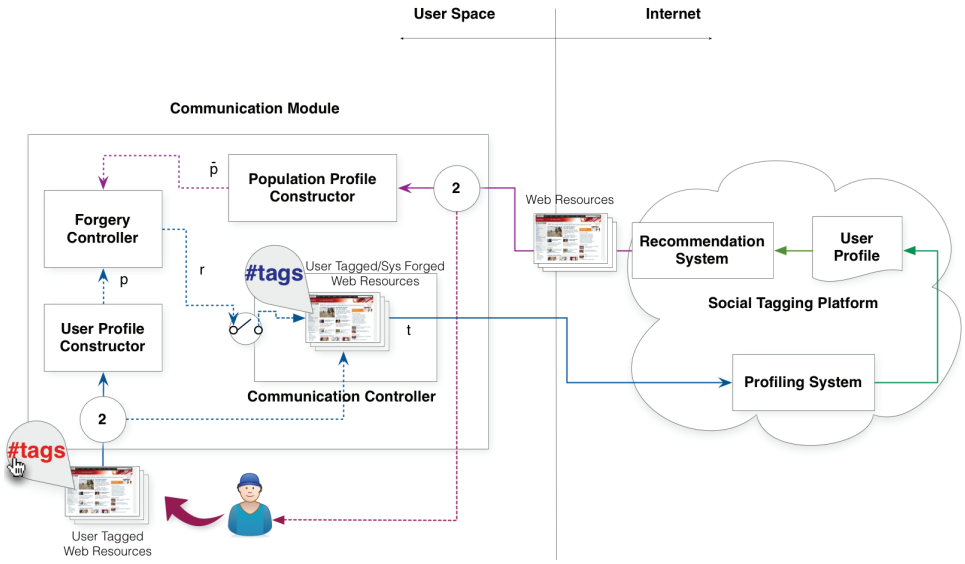



**Figure 3.2.1:** The proposed architecture of a communication module managing the user data flows with a social tagging platform and implementing different possible forgery algorithms.

content can be considered safe. In addition keywords in categories considered sensible could be excluded, either automatically or by the users. In our architecture is the user who ultimately decides whether to follow the recommendations proposed by our communication module or not.

Additionally, it is worth mentioning that, if the user decided to reduce excessively the number of categories used to produce a possible forgery strategy, their user profile would inevitably exhibits a spike in activity in the chosen categories. As a consequence, the apparent user profile would probably become more identifiable to an external attacker. We therefore believe that although the user should be allowed to tweak their forgery strategy, they should also be informed of the consequences of applying some settings instead of others to the communication module.

We have also considered the possibility to implement our proposed architecture as a mobile application. We are aware this might add a computational, and networking overhead on the platform where the module will be installed, yet we

also believe that in modern mobile platforms and personal computers this shall not be an issue. More importantly we believe that the benefit of controlling the user perceived profile shall overcome the cost of implementing the proposed architecture.

Profile data are in fact collected not only by social tagging platforms but also by websites, web applications and third parties even when the user is not connected to a personal account. Through tracking technologies and a networks of affiliated web sites users can be *followed* online and their footprint collected for a variety of uses. If aggregated, these data could reveal more over time that the same users initially intended. The data then turn from merely figures to piece of information able to describe users' identity and behaviours. Social engineering attacks could exploit users' profiles on different social networks to gather certain sensitive information. Similarly users' profiles crawling across different services and applications can disclose relevant facts about the users. It is, therefore, important for users to maintain a desired online privacy strategy. At the same time, this approach could also be implemented by developers and systems architects who need to be aware of the possible privacy and security implication of their work.

## 3.3 Evaluation

Evaluating how a recommender system would be affected when tag forgery is applied in a real world scenario is interesting for a different range of applications. We have particularly considered both the point of view of the privacy researcher interested in understanding how user privacy can be preserved, and also the perspective of an application developer willing to provide users with accurate recommendation regarding content and resources available on their platform.

Every PET must in fact ensure whether the semantic loss incurred in order to protect private data can be acceptable for practical use.

**Table 3.3.1:** Statistics regarding Delicious dataset

| Statistics about the built dataset | | | |
|---|---|---|---|
| Categories | 11 | Users | 1867 |
| Item-Category Tuples | 98998 | Avg. Tags per User | 477.75 |
| Items | 69226 | Avg. Items per Category | 81044 |
| Avg. Categories per Item | 1.4 | Tags per item | 13.06 |

Thus, different tag forgery strategies were considered in a scenario where all the users were willing to apply the techniques. It was also considered that a user would try to apply a certain technique at different forgery rates, in order to evaluate how utility would be affected on average at each rate. When forgery rate is equal to zero it means the technique is not applied.

Hence, the overall utility for the recommender system, based on the applied forgery rate was evaluated against the privacy risk reduction calculated after each step.

In our simulated scenario, a user would ideally implement a possible PET at a time. We have therefore considered what percentage of utility the hypothetical user would lose when incrementing the ratio of forged tags with each strategy, consequently underlining what percentage of privacy risk reduction has gained in front of a certain loss in utility.

The user in this setup is presented over time with a list of top results, they would then decide to click or not on a number of these resources. This number divided by the total number of results gives us the percentage of items that the user has actually

found interesting. Our utility metric is then evaluated considering the cases for which the user has been presented with the top 30 results, and the top 50 results.

Note that since in our experimental setting, we have split the data into a testing and a training set [**?** **?** ], considering relevant only the items in the user's profile, it is not possible to evaluate items that are as yet unknown to the user but that could also be considered relevant (Fig. 3.3.1). In a real world application in fact, a user could be presented with results that are unknown to them, but that do reflect their expressed interests. Therefore our estimation of precision is in fact an underestimation [**?** ].

In order to evaluate the impact of a determined PET on the quality of the recommendation, and elaborate a study of the relationship between privacy and utility, a dataset rich in collaborative tagging information was needed. Considering different social bookmarking platforms, Delicious was identified as a representative system. Delicious is a social bookmarking platform for web resources [**?** ]. The dataset containing Delicious data was obtained from the ones publicly available at the 2nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems [**?** ], accessible on *http://ir.ii.uam.es/hetrec2011/datasets.html*, and kindly hosted by GroupLens research group at University of Minnesota. Furthermore, the dataset also contained category information about their items, this corresponds to the first and second taxonomy levels of the ODP classification scheme (Table 3.3.1) [**?** ]. The ODP project, now DMOZ, is the largest, most comprehensive human-edited directory of the Web, constructed and maintained by a passionate, global community of volunteers editors.

The chosen dataset specifically contains activity on the most popular tags in Delicious, the bookmarks tagged with those tags, and the users that tagged each bookmark. Starting from this specific set of users, the dataset also exhibits their contacts and contacts' contacts activity. Therefore it both covers a broad range of document's topics while also presenting a dense social network [**?** ].

The experimental methodology is described also by Fig. 3.3.1. The dataset is

randomly divided between two subsets, namely a testing and a training set. The training set contains 80% of the items for each user, and was used to build the users' profiles. The testing set contained the remaining 20% of the items tagged by each user, and was considered to evaluate (test) the recommender itself.

The first step of the experiment involved obtaining a metric of the recommender performance without applying any PET. The recommender would then produce estimation of how relevant an item potentially is for a user, by comparing the calculated user profile with each profile of the items in the testing set. This step would return a list of top items for each user. At this point our precision metric is calculated by verifying which of the top $V$ items have actually being tagged by each user. This process is repeated at each value of $\rho$ to understand how applying a different PET affects the prediction performances of a simple recommendation system. Please note that the three different PET have been considered independently for one another, i.e. the users would apply one of the techniques at a time and not a strategy involving a combination of the three.

### 3.3.1 Experimental results

In our experimental setup, we have firstly evaluated what level of privacy users will reach implementing each of the strategies considered. Fig.3.3.3 shows how the application of the different PETs at different values of $\rho$ affect the privacy risk $\mathcal{R}$.

The first interesting result can be observed by considering how the privacy risk $\mathcal{R}$ is affected by the application of a certain PET. For values of $\rho \in [0, 0.25]$ (Fig. 3.3.6), $\mathcal{R}$ is decreasing for all three strategies, although with optimised forgery this seem to be happening faster.

When larger values of $\rho$ are considered, the apparent user profile will most likely mimic the profile of either the population distribution, in the case of optimised forgery, the TMN distribution in the case of TMN and the uniform distribution in the case of uniform forgery. If we consider this apparent effect, we understand why, while the privacy risk approaches 0 in the case of optimised forgery, it actu-

ally increases both for TMN and uniform forgery (Fig. 3.3.3). Recalling that our privacy metric, and adversary model, consider the case for which a possible attacker would try to isolate a certain user from the rest of the population, applying a forgery strategy that would generate an apparent profile $t$ that would increase the divergence from an average profile, would actually result in making the user more easily identified from a possible observer.

This undesirable consequence is also more eloquently present when applying the uniform strategy, in fact as the user apparent profile approached the uniform distribution for higher values of $\rho$, it would become evident to an external observer which users are forging their tags according to this strategy.

In the case of optimised forgery instead, privacy risk decreases with $\rho$. Naturally for $\rho = 0$ the privacy risk for all the users applying a technique is actually maximum, while it will approach 0% when $\rho = 1$. It is particularly interesting to see how our optimised tag forgery strategy allows users to reduce their privacy risk more rapidly even for small values or $\rho$.

We have therefore measured the total number of users that would actually increase their privacy risk as a consequence of having applied a certain PET (Figs. 3.3.8). It is surprisingly striking to observe how almost 90% of the total number of users, when applying TMN or uniform forgery, would make their apparent profile more recognisable than without implementing any PET. This reflects the intuitive assumption that in order to conceal the actual user's profile, with the privacy metric considered throughout this work, it would be advisable to make it as close as possible to an average profile of reference, so that it is not possible to individuate it, or in other words to distinguish it from the average population profile.

We then have evaluated how our utility metric was affected by the application of the tag forgery strategies, for different values of $\rho$. We have considered two situations to evaluate our utility metric. In the first case the user would be presented with the top 30 results, and in the second with the top 50. This allowed us, not only to evaluate the impact of noise on the metric itself, but also to consider the impact of a certain strategy over longer series of results.

Fig. 3.3.5 and Fig. 3.3.4, show the obtained utility versus the rate of tag forgery

applied, this has been evaluated again for optimised forgery, uniform forgery, and TMN strategy, in order to understand how these PETs perform in the described scenario.

In this case we noticed how a uniform forgery strategy, which generates bogus tags according to a uniform distribution across all categories, is able to better preserve utility than either optimised tag forgery or TMN, especially for bigger forgery ratios.

What we found particularly relevant in our study is that for smaller values of $\rho$, hence for a forgery rate up to 0.1, corresponding to a user forging 10% of their tags, our optimised forgery strategy shows a privacy risk reduction $\mathcal{R}$ of almost 34% opposed to a degradation in utility of 8%. This result is particularly representative of the intuition that it is possible to obtain a considerable increase in privacy, with a modest degradation of performance of the recommender system, or in other words a limited utility loss (Fig. 3.3.7).

The results obtained therefore present a scenario where applying a tag forgery technique perturbs the profile observed from the outside, thus enabling users to protect their privacy, in exchange of a small semantic loss if compared to the privacy risk reduction. The performance degradation measured for the recommendation systems, is small if compared to the privacy risk reduction obtained by the user when applying the forgery strategy considered.

## 3.4 Discussion

Information filtering systems that have been developed to predict users' preferences, and eventually use the resulting predictions for different services, depend on users revealing their personal preferences by annotating items that are relevant to them. At the same time, by revealing their preferences online users are exposed to possible privacy attacks and all sorts of profiling activities by legitimate and less legitimate entities.

Query forgery arises, among different possible PETs, as a simple strategy in terms of infrastructure requirements, as no third parties or external entities need

to be trusted by the user in order to be implemented.

However, query forgery poses a trade-off between privacy and utility. Measuring utility by computing the list of useful results that a user would receive from a recommendation system, we have evaluated how three possible tag forgery techniques would perform in a social tag application. With this in mind a dataset for a real world application, rich in collaborative tagging information has been considered.

Delicious provided a playground to calculate how the performance of a recommendation system would be affected if all the users implemented a tag forgery strategy. We have hence considered an adversary model where a passive privacy attacker is trying to profile a certain user. The user in response, adopts a privacy strategy aiming at concealing their actual preferences, minimising the divergence with the average population profile. The results presented show a compelling outcome regarding how implementing different PETs can affect both user privacy risk, as well as the overall recommendation utility.

We have firstly observed how while the privacy risk $\mathcal{R}$ decreases initially, for smaller values of $\rho$ (for both TMN and uniform forgery), it increases as bigger forgery ratios are considered. This is because the implied techniques actually modify the apparent user profile to increase its divergence from the average population profile. This actually makes the user activity more easily recognised from a possible passive observer. On the other hand, optimised forgery has been designed to minimise the divergence between the user and the population profile, therefore the effect described is not observed in this case.

Considering this unfavourable effect, we have computed the number of users that would actually increase their privacy risk. This particular result showed how applying a certain PET could actually be detrimental to the user's privacy: if the user implemented a strategy that is not accurately chosen, they would be exposed to a higher privacy risk than the one measured before applying the PET. Observing how the application of a PET affects utility, we have found out that especially for a small forgery rate (up to 20%) it is possible to obtain a consistent increase in privacy, or privacy risk reduction, against a small degradation of utility. This re-

flects the intuition that users would be able to receive personalised services while also being able to reasonably protect their privacy and their profiles from possible attackers.

This study furthermore shows in a simple experimental evaluation, of a real world application scenario, how the performances degradation of a recommendation system, is small if compared to the privacy risk reduction offered by the application of these techniques. This opens many possibilities and paths that need to be explored to better understand the relationship between privacy and utility in recommendation systems. In particular it would be interesting to explore other definitions of the metrics proposed and apply these on different class of recommendation systems.

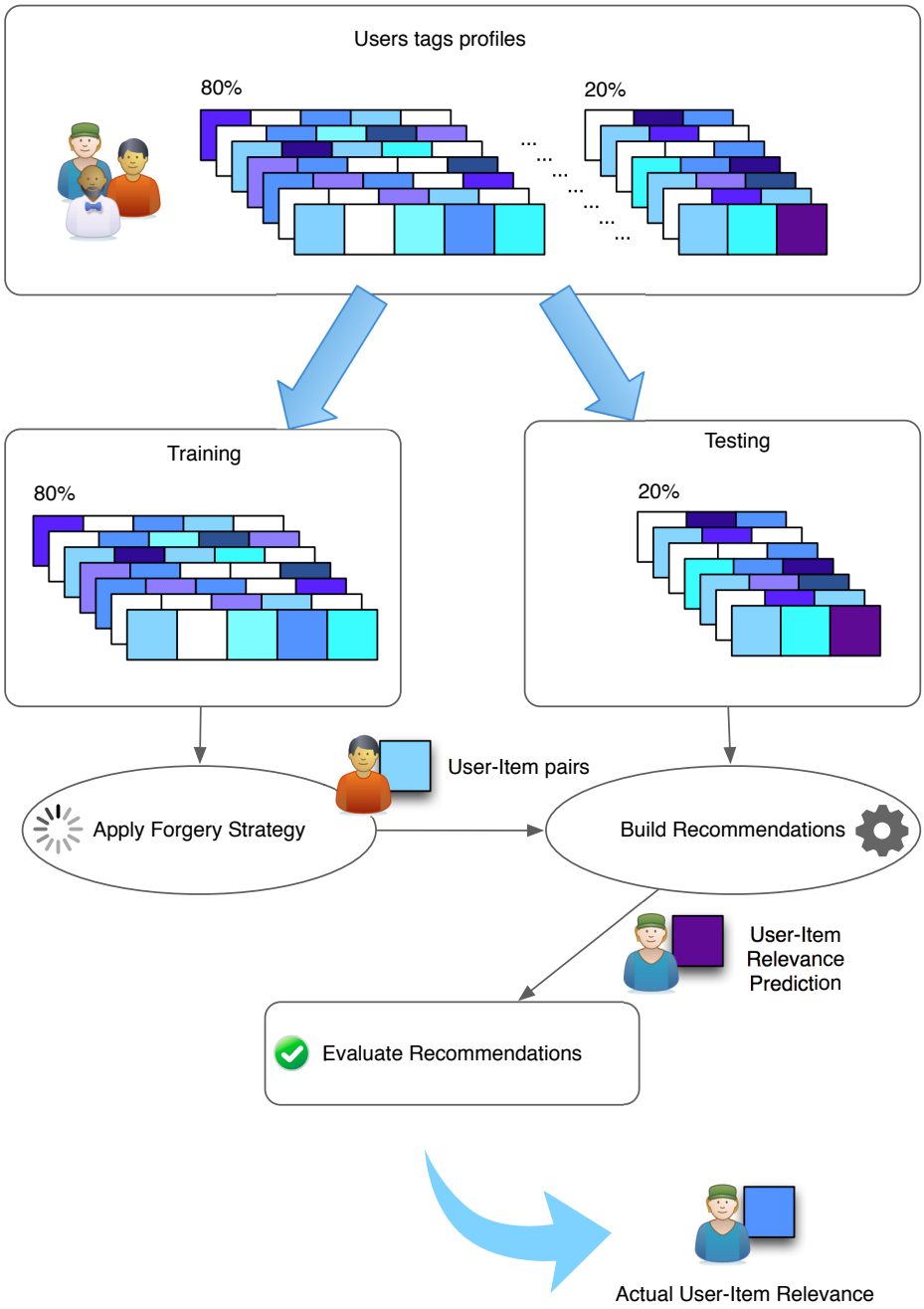



**Figure 3.3.1:** Experimental methodology.

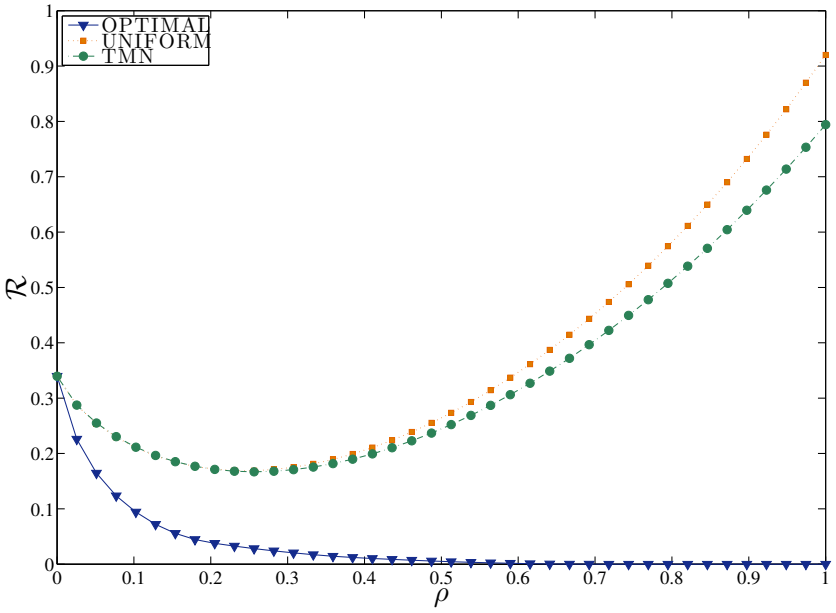



**Figure 3.3.2:** Privacy risk $\mathcal{R}$ against forgery rate $\rho$ for a single user.

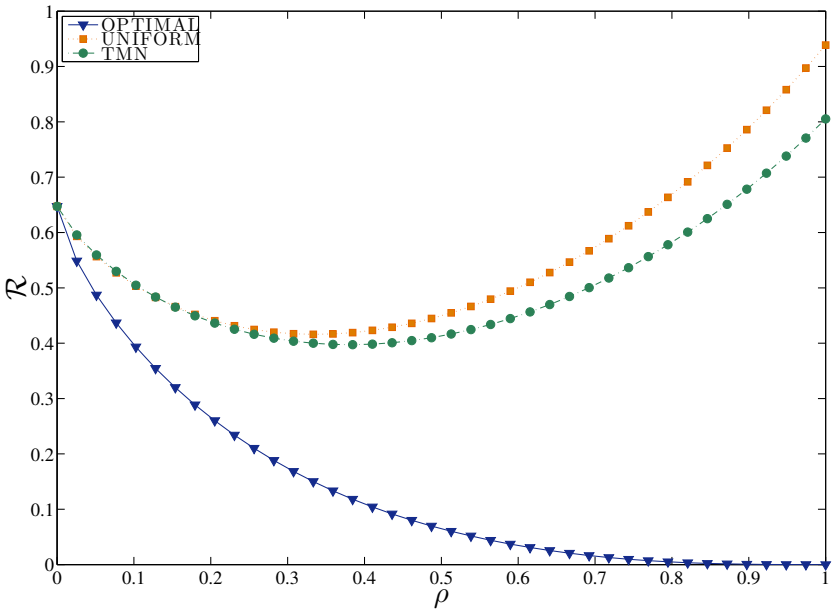



**Figure 3.3.3:** Privacy risk $\mathcal{R}$ against forgery rate $\rho$ for all users. For the optimised forgery strategy the privacy risk $\mathcal{R}$ decreases with $\rho$. Naturally for $\rho = 0$ the privacy risk for all the users applying a technique is actually maximum, while it will approach 0% when $\rho = 1$. The graph shows how the optimised tag forgery strategy allows users to reduce more rapidly their privacy risk even for small values or $\rho$. This confirms the intuitive assumption that applying a forgery strategy that actually modifies the user's apparent profile to increase its divergence from the average population profile, would produce the unfavourable result to make the user activity more easily recognised from a possible passive observer.

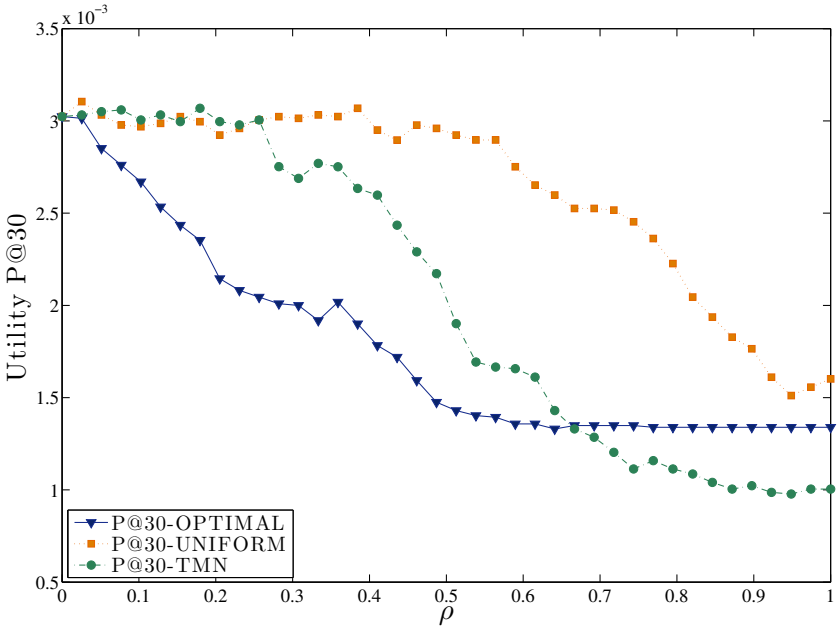



**Figure 3.3.4:** Average value of utility P@30 calculated for different values of $\rho$.

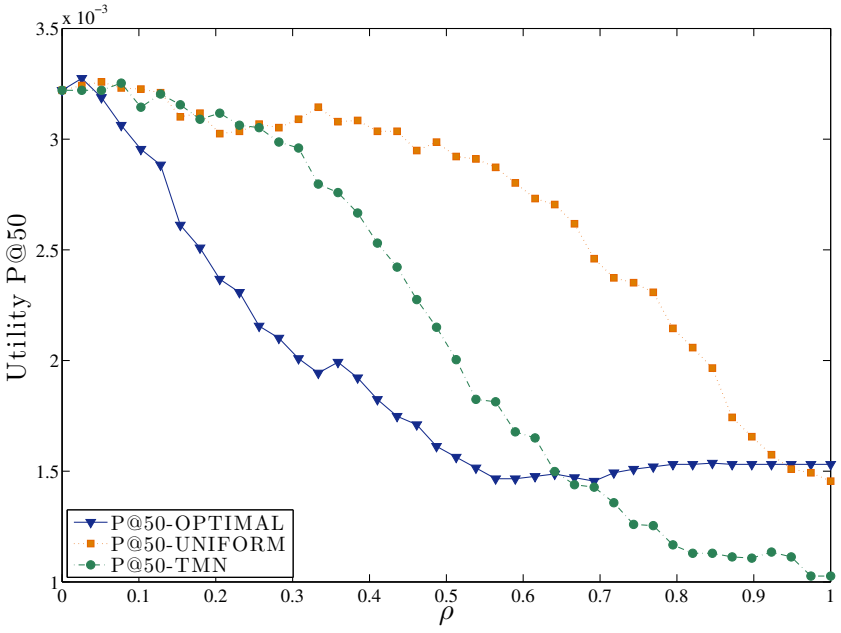



**Figure 3.3.5:** Average value of utility P@50 calculated for different values of $\rho$. It is important to note that the measure of utility averaged across the user population is affected by statistical noise creating some glitches in the function that we can see attenuated if presenting each user with a larger list of results to choose from.

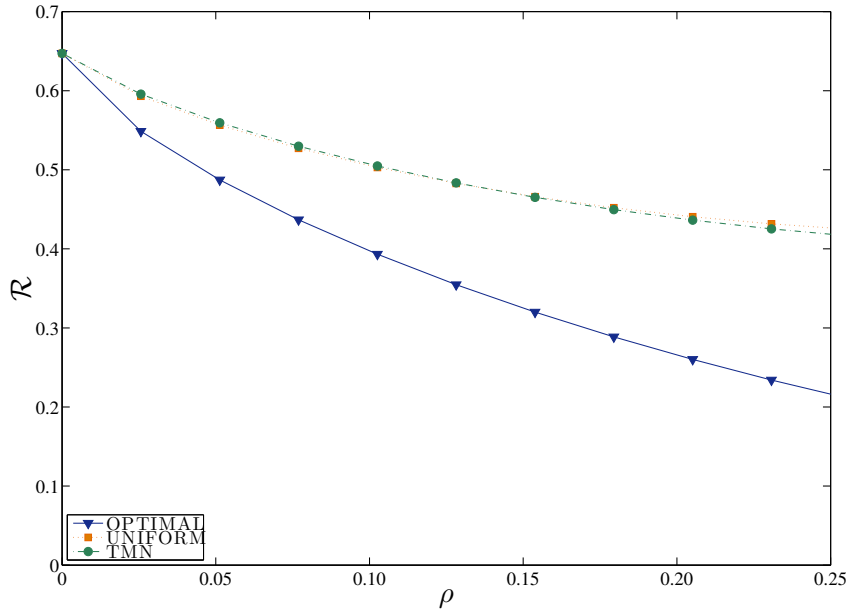



**Figure 3.3.6:** Privacy risk $\mathcal{R}$ against forgery rate $\rho$ for all users applying a PET considering only values of $\rho \leqslant 0.25$.

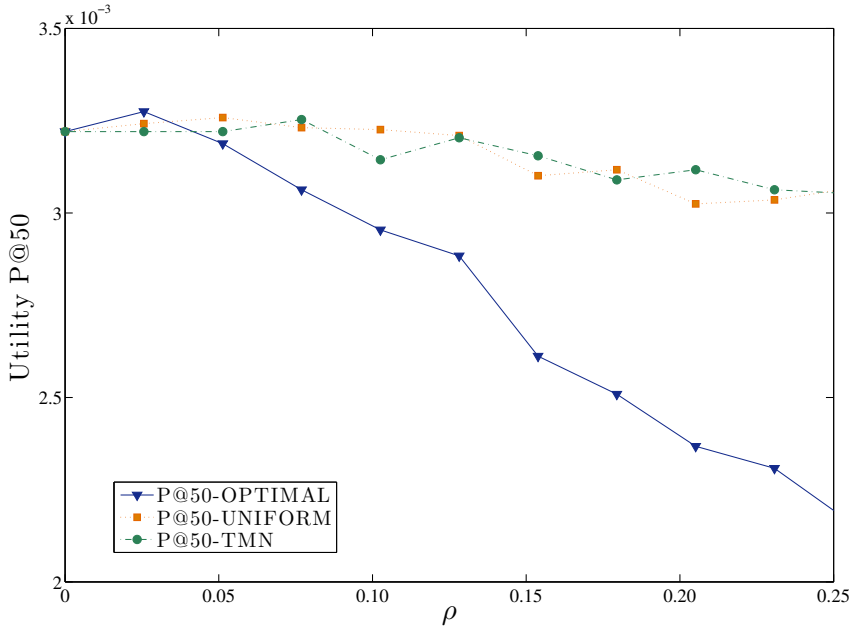



**Figure 3.3.7:** Privacy risk $\mathcal{R}$ against forgery rate $\rho$, compared with the average value of utility P@50, for small values of $\rho$, for all users applying a PET. It is interesting to note the ratio between the privacy risk $\mathcal{R}$ and the utility loss only for small values of $\rho$.

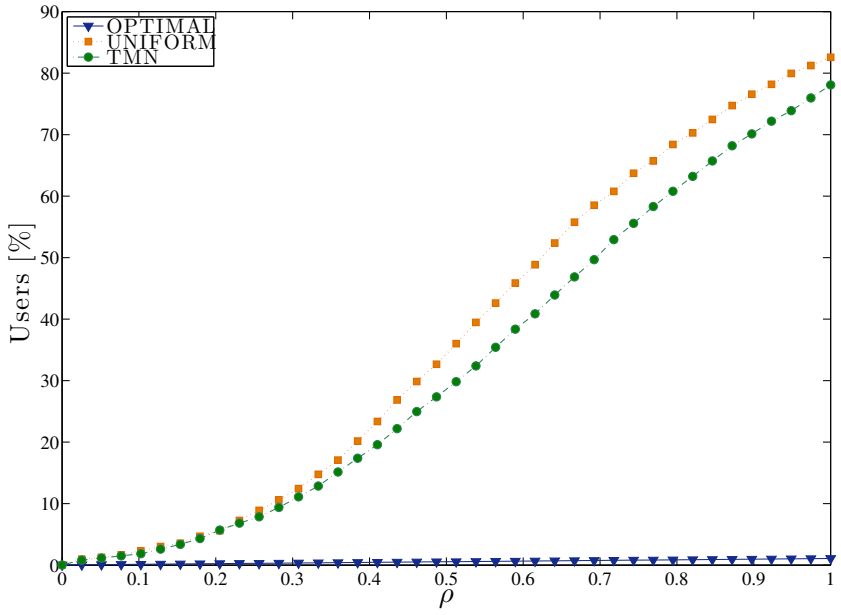



**Figure 3.3.8:** Actual number of users increasing their privacy risk as a side effect of applying a certain strategy for a given value of $\rho$.

*If you want to keep a secret, you must also hide it from yourself.*

George Orwell, 1984

# 4

# Privacy in proximity based apps: the nightmare of serendipitous discovery

The communication possibilities opened by online services are almost endless. Social media allow people every day to know more about themselves, their friends and their surroundings. To use such services, users grant them a certain level of access to their private data. This data includes details about their identity, their whereabouts and in some situations even the company they work for. This level of access is obtained leveraging on third parties, like Facebook or Google, which offer login technologies, allowing the application to identify the user and receive precise information about them.

In this chapter we focus on the privacy issues posed by mobile social applications, continuing work presented in [**? ?** ].

We start by analysing how, mobile apps request access permission to user's information by using a federated login mechanism. Once the user has granted access to their data, the application stores it and assumes control over how it is further shared. The user will never be notified again on who is accessing their data, nor if these are transferred to third parties. Furthermore mobile applications can access data generated by sensors on the device, disclosing even more information about the user and exposing them to privacy attacks, while in addition preventing users to retain direct control over their data and who has access to it over time.

This aspect of privacy protection is particularly relevant since usually the right to privacy is interpreted as the user's right to prevent information disclosure. online services use this interpretation to ask the user to access certain information, yet no concrete information is passed on how the data will be used or stored. Furthermore, these services are often designed as mobile applications where all the devices installing the app communicate with a centralised server and constantly exchange users' information, eventually allowing for unknown third parties, or potential attackers, to fetch and store this data. In addition, this information is often shared with insecure communication through the HTTP protocol, making it possible for a malicious entity to intercept these communication and steal user data.

We have observed how proximity-based social applications have access to certain identity information that could lead a possible privacy attacker to easily identify users and link their online profiles to physical identities. In our study we analyse a set of popular dating application, which are built on the assumption that users can preserve a certain level of privacy by only sharing their relative distance with other users on the platform. Furthermore, the user also shares Facebook likes or common categories of interests.

These application are built on the notion of serendipitous discovery of people, places and interests around the user's surrounding. We consider these applications an example of how many privacy violation users are subjected to without being aware of it. Furthermore, this scenario offers a playground to prove how little details about the user's whereabouts and personal sensitive information are needed to track the user and discover their real identities. For example, we prove how the

user's relative distance or their first name and what common interest their share on Facebook, can allow an attacker to follow them along the day and across their movements, or even profile their full interests and discover personal details about them.

## 4.1 Background

Online communications in general and social media in particular, are increasingly opening up new possibilities for users to share and interact with people and content online. At the same time, social networking services collect and share valuable information regarding locations, browsing habits, communication records, health information, financial information, and general preferences regarding user online and offline activities. This level of access is often directly granted from the user of such services, although the privacy and sensitiveness of the information becoming accessible to third parties can be easily overlooked.

Furthermore, social networks are no longer a novelty and user have become used to share their information with both social relationships as well as third party applications. Leveraging on this perception of social media by Internet users, another class of applications is being developed based on the concept of *serendipitous* discoveries. The idea of *serendipity* in mobile applications wants the user to accidentally discover people, places and/or interests around them, by using passive geo-localisation and recommendation systems. Passive geo-localisation is a mechanism using the ability of mobile devices to know the user's position without having to constantly asking for it. Technologies that provide this capability are GPS, wireless and mobile networks, iBeacon and so on.

To present the user with a tailored and seamless experience, serendipity applications need to learn the user's preferences and interests. This is usually accomplished by connecting several of the user's identities on other social networks. A typical example is asking the user to register onto an application through their Facebook, Twitter, or Google+ accounts. This technique usually consists in a variant of the OAuth2.0 protocol used to confirm a person's identity and to control

what data they will share with the application requesting login.

We have specifically analysed Facebook login since it was the common login mechanism offered in all applications examined, although the same functionality apply for other third party login mechanisms. Facebook login provides both authentication and authorisation. The mechanism is used on the web as well as on iOS and Android, although on those platforms the primary mechanism uses the native Facebook application instead of the web API.

When an application is connected to the user's Facebook profile using Facebook login, it can always access their *public profile* information. Facebook consider this information public and will not apply any restriction on it. Information that is shared with the public profile vary from user to user and depends on their privacy settings. By default, the Facebook public profile includes some basic attributes about the person such as the user's age range, language and country, but also the name, gender, username and user ID (account number), profile picture, cover photo and networks.

An application may also ask for more information about the user. These can include the list of friends using the app, their email, the events that they are attending, their hometown or the things they have liked. This information can be obtained by requesting for optional permissions, which are asked for during login process. Apps can also ask for additional permissions later, after a person has logged in.

The information obtained from Facebook is often displayed on the application platform or used to match people with similar interests, thus giving away more hints about an individual real identity. For example, a user *swiping* through other people on *Tinder* [**?** ] will know if they have liked similar pages on Facebook. These hints or traces can be used to further identify that individual on other platforms. In fact, this information crossed with the city the user lives in, the user's photo, and their first name could already be enough to identify their Facebook profile.

The attacker could hence use what they know about the user to identify a number of profiles of people living in a certain city. A query of the form *people named John who live in Barcelona and like surfing and volleyball* could be used to restrict the attacker's search space to a smaller number of profiles. Finally, since these ap-

plications tend to fetch the profile photo directly from Facebook, the actual user's profile can be identified by matching the two profile pictures.

Notice that while some queries might seem very generic, some others might already restrict significantly the set of targeted profiles. It is particularly concerning in fact that these applications might be used to target specific individuals with the objective to reach confidential information about their actual job or company they work for, as reported recently by IBM in a report about security of dating apps [**?** ].

The ubiquitous streams of data that users create while they use different application can be seen as a network of interconnected data snippets. Information shared on the web can be linked together so that it is possible to construct semantic connections between user's activity data. A possible attacker could therefore try to link data between different source of information to identify and target users both online and offline. Users become more frequently exposed to social engineering attacks that can now leverage on facts gathered online about their personal offline lives.

In this chapter we formalise an attack showing how proximity based social application are inherently insecure. Our attack retrieves information about nearby users, stores certain information about them, and subsequently uses these to retrieve their updated profiles at regular intervals. Our attacker agent is also able to change their relative position at will and therefore can easily perform a multilateration attack and identify the victim position with a arbitrary precision. Furthermore, the attacker can keep following the user, eventually categorising their interests, movements and even identifying their Points of Interest (POI) around the city.

Therefore, we build a Social Graph attack using Facebook likes to know the victim interests. The applications examined, in fact, allow the attacker to know *what they have in common* with the victim and use the known expressed interests to identify the user's Facebook profile through their Graph Search while also profiling individuals nearby.

## 4.2 Modelling the location probe method

Proximity based social application collect users' positions and share their relative distances. We show how it is possible to build a multilateration attack able to identify the actual user position with arbitrary precision.

Multilateration is a navigation technique, often used in radio navigation systems, based on the measurement of the difference in distance to two or more stations, whose locations are known. The stations also produce a certain signal at a known time.

In our scenario, the signal is replaced by the user distance from the attacker and time is given by the timestamp of the user latest activity. Please note that, multilateration is not concerned with measurements of absolute distance or angle between parties, but with measuring the difference in distance between two stations which results in an infinite number of locations that satisfy the measurement. All these possible locations form a hyperbolic curve. Multilateration therefore relies on multiple measurements to locate the exact location along that curve. In fact, a second measurement taken to a different pair of stations will produce a second curve, which intersects with the first. When the two curves are compared, a small number of possible locations are revealed.

If the attacker is able to retrieve an arbitrary number of samples of the user distance, either by changing their relative location or by sampling their distance with the victim with a number of malicious mobile client infiltrating the platform, the multilateration attack can be made arbitrary precise.

Our location probe method uses a simple multilateration algorithm. At the first step, locations expressed as longitude and latitude coordinates are translated to cartesian coordinates. We then calculate the estimated distance and minimise the linear norm between calculated distance and estimated distance by sensing the total error. We could have considered the total squared error between the estimated and actual distance, however in this contest we have concentrated on demonstrating that the attack is actually feasible, rather then on accuracy or performance of

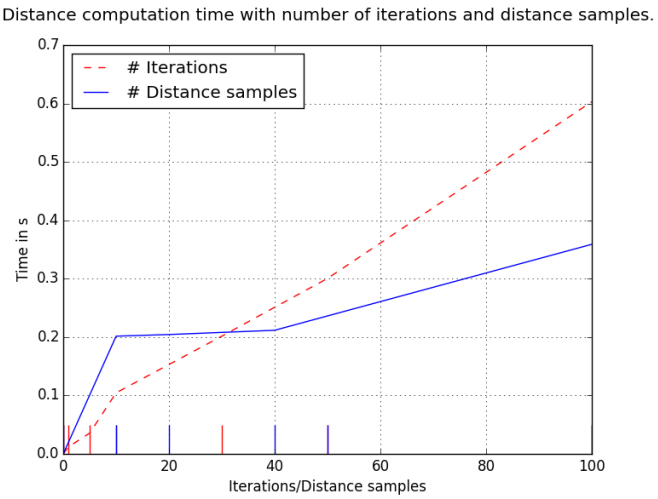



**Figure 4.2.1:** The image illustrates the time needed to compute a user position estimation based on the number of distance samples and the number of iterations of the algorithm. It is important to note how the number of distance samples does not affect the algorithm performances. The example was executed on a Apple Computer with 3 GHz Intel Core i7 Processor.

the algorithm (Fig. 4.2.1).

## 4.3 Modelling the user activity profile

We model the user's activity as series of events belonging to a certain identity. Each event is a document containing different information. We can formally define this a hypermedia document i.e. an object possibly containing graphics, audio, video, plain text and hyperlinks. We call the hyperlinks selectors and we use these to build the connections between the user's different identities or events. Each identity is a profile that the user has created onto a service or platform. This can be an application account or a social network account, such as their LinkedIn or Facebook unique IDs.

Each event is the result of the user performing an action. For the purpose of this study we have consider an action as resulting using an application or a service. An action is the activity of interacting with a mobile application or *liking* a resource

on a social network, i.e. directly expressing an interest, or the fact that a user has updated their location at a certain time.

Formally it is possible to model the graph of the events pertaining to a user as an hypergraph, where each edge can connect any number of vertices, and the root is first event in the series. A hypergraph $H$ is a pair $H = (X, E)$ where $X$ is a set of nodes (the events in the model), and $E$ is a set of non-empty subsets of $X$ called hyperedges or edges. Hypergraphs are a generalisation of a graph structure and provide a reasonable representation of the connections between the different events resulting of the actions performed by the user.

We find that this model is able to express the user's online footprint as a collection of traces left across different services. Furthermore, by using a hypergraph model we are able to grasp the connections between the different profiles and features.

This results in the possibility to profile users based on chosen selectors. For example, we might want to trace all users who have been in the radius of 500 meters to a certain location, or all the users in a certain neighbourhood who *like* a selected Facebook page.

### 4.3.1 Adversary model

In view of the assumptions described in the previous section, our privacy attacker boils down to an entity that aims to identify users and link their online profile to their physical identity. To achieve this objective, the attacker possess a Facebook profile. This profile is used in the first place to register to the application analysed in this study since all three use Facebook login as a personalised way for user to register and sign in.

## 4.4 Experimental results

We have analysed 250 users from a set of social proximity applications (Table: 4.4.2). All applications examined are matchmaking mobile platforms which uses geolocation technology. Users can use their location and preferences to search

for interesting people in a specific radius. All applications use Facebook profiles to allow their users to login but also to gather basic information and analyse users' social graph. The information collected are then used to match candidates who are most likely to be compatible based on geographical location, number of mutual friends, and common interests.

**Table 4.4.1:** Information regarding active users per application.

| Application | Users |
|---|---|
| Tinder [**?** ] | 10 Million active [**?** ] |
| Happn [**?** ] | 700.000 [**?** ] |
| Lovoo [**?** ] | 24 Million registered [**?** ] |
| Grindr [**?** ] | 2,35 Million active [**?** ] |
| Badoo [**?** ] | 200 million registered [**?** ] |

These applications present the user with the possibility to interact with other users by starting conversation or expressing their interests in them.

### 4.4.1 Information collection

Information collection is possible on these applications through different techniques. For the purpose of this study we have intercepted APIs call from mobile devices through Men In The Middle (MITM) attack in some occasions, and interacted with the APIs directly in other occasions. It is important to note that even when the application prevents an attacker from exploiting their APIs, a malicious entity could still use a multitude of profiles to cross gather information about users on the platforms.

**Table 4.4.2:** Information regarding the applications analysed

| Application | Fb ID | Loc. | Distance | User Pref. | Full Name | Birth-date | User tracking |
|---|---|---|---|---|---|---|---|
| Tinder [? ] | ✗ (1) | ✗ | ✓ | ✓ | ✗ (2) | ✗ (3) | ✓ |
| Happn [? ] | ✓ (1) | ✗ | ✓ | ✓ | ✗ (2) | ✗ | ✓ |
| Lovoo [? ] | ✗ (1) | ✗ | ✓ | ✓ | ✗ (2) | ✗ | ✓ |
| Grindr [? ] | ✗ (1) | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ |
| Badoo [? ] | ✗ (1) | ✗ | ✓(4) | ✓ | ✗ (2) | ✗ | ✓ (6) |

(1) Facebook ID is not exposed directly but it can be identified by crossing information like the user Facebook's likes, first name and year of birth.
(2) Only first name is shared.
(3) A fuzzy birthdate randomised in a range of two weeks is used. Real birthdate can be inferred by using Facebook Graph Search, depending on the victim's Facebook privacy settings.
(4) Offers option not to share distance.
(5) Asks for zodiac sign.
(6) Distance is shared for some users so it is theoretically possible.

### 4.4.2 Information processing

We have performed two types of attack on the set of application examined, namely a multilateration attack and a social graph attack.

#### Multilateration attack

Once we posses the user's id on the specific application we are able to query their APIs and update our information about the user constantly. Furthermore we are also able to change our own location on the platform to a certain extent. By measuring the relative distance to the victim we were able to identify their actual posi-

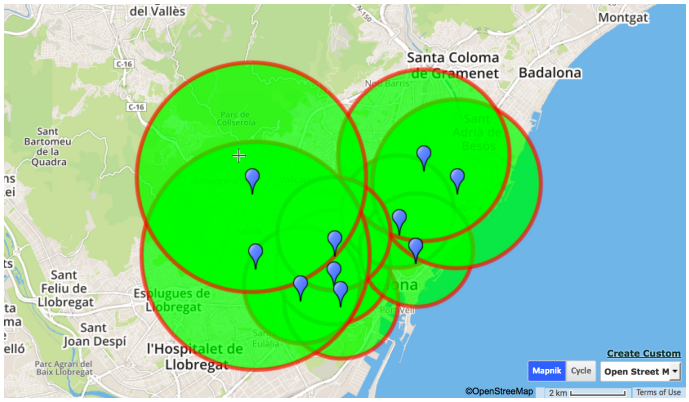



**Figure 4.4.1:** The image illustrates location samples with radii used to compute actual position estimation for one user across the city of Barcelona, Spain.

tion with arbitrary precision. Furthermore, the same technique was used to *follow* users across a specific amount of time by retrieving their profile information at regular interval. This type of attacks can be easily overlooked in densely populated cities but might become a serious privacy breach in rural areas.

### Hyper graph attack

The application examined for the scope of this study use the user's Facebook token to authenticate and/or authorise the application to request and obtain certain information about the user. An attacker could then use their own Facebook profile token to make request to the application server through their APIs, pretending to send the request from the app installed in a mobile device. This allows the attacker to receive all the information that users have shared with the platform and that are constantly exchanged with the application.

When the victim's Facebook id is shared through the application, the attacker can directly access and potentially use information publicly shared through the Facebook profile. In this situation the attacker could easily construct a complete graph of the user's preferences and social connections through the information that are public available through Facebook APIs.

When the victim's Facebook id is not directly shared, the application still disclose some information about the victim. This information include: the user first name and a set of photos, birthdate, randomised in a range of 15 days, and the Facebook pages that both the victim and the attacker have liked.

The victim preferences could then be used to identifies their Facebook profile. It is in fact estimated that Facebook posses 1.35 billion active users, of these, between 10% and 7% like one of the top 10 Facebook pages with most likes [**?** ]. We have collected a set of 250 Tinder users only in the city of Barcelona, of these 20% where sharing at least one interest with the attacker profile (Fig. 4.4.2).

Furthermore Facebook graph search allows any users to answer certain information about Facebook profiles. An example of a graph search on Facebook could be: *People who like Shakira and are named "John" and like Manchester United and been born in 1979*. This will create a pool of potential candidates. The list can be reduced by using Facebook reverse graph search, i.e. search for *Interests liked by people who like Shakira and are named "John" and like Manchester United and been born in 1979*. This will instead return a list of interests that the attacker can like on Facebook. Therefore, the attacker will return to query Tinder and find out if the number of interests in common with the victim has grown and which pages they now have in common. The attacker can therefore use the new information to further identify the victim profile on Facebook and potentially their friends (Fig. 4.4.3).

It is important to note that some applications might request information outside of Facebook public profile. Therefore, even if the victim has tailored their privacy settings to prevent some information to be leaked, the application can be used to access data that would be otherwise be kept private.

### 4.4.3 Information dissemination

Proximity-based social applications, in their current implementation, represent a gateway to access data about individuals. Information dissemination can therefore be accomplished both for large group of people with the purpose of targeting them,

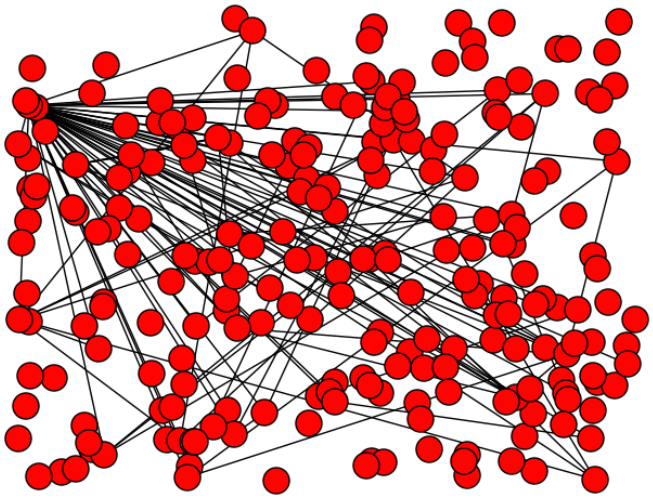


**Figure 4.4.2:** The image shows how it is possible to show connections for the population of users on Tinder for a certain area. Here we have collected Facebook pages liked by users in Barcelona and connected users or group of users, if they like the same page.

as well as for specific victims. Identifying and disclosing the presence of a certain person on a match making application could be enough to influence the opinion of that individual among their social relationships.

### 4.4.4 Invasion

Once a user location has being inferred, we can continue tracking the same users and their preferences for an unlimited amount of fetches. This could easily lead to identification of the user habit and where-about at different moment of the day, possibly uncovering their home and work locations and more information about the user.

## 4.5 Mitigation possibilities

Application developers could implement a number of techniques that would mitigate the actions of a possible attacker. Firstly, in their current implementation,

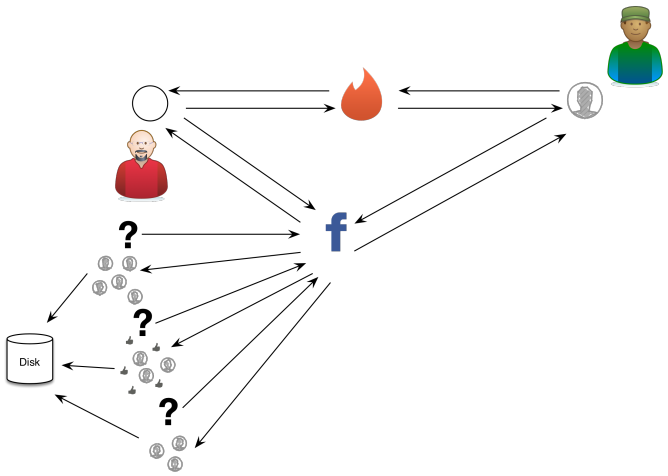



**Figure 4.4.3:** The image represents a Social Graph attack where an attacker sends queries Facebook asking question about a Tinder profile. The attacker is able to restrict the pool of potential candidates and eventually identify the victim's actual Facebook id. Furthermore the attacker is able to store information about the user that can be updated at a later time by querying the third party application.

the applications examined probe the user device for location information with the maximum precision possible. This information is then transferred to the server and the relative distance between users is returned to be displayed. Yet, for most of the application functionality this precision is not needed, and a lower precision could be used and sent to the server. This would make position attacks more difficult to perform.

Secondly, to sparkle interest between users, social proximity application often share common Facebook pages between parties involved. This information can then be used to easily identify unique Facebook accounts. Instead, the app could opt to display only the category of interest to which the Facebook page belongs. This way a possible attacker would not know what actual pages the user has liked.

Thirdly, an individual birth-date if combined with their location and first and/or last name can be used to infer sensitive information about them. Therefore even sharing the user's zodiac sign with passive observer need to be considered poten-

tially dangerous for the final user's privacy.

To conclude, to avoid exposing users to direct threats of *collection* and *processing* of private information, mobile apps should have the option not to supply any personal details to the platform. Users should not obliged to disclose their personal data. To avoid *dissemination* and *invasion*, user data collected by mobile applications should be communicated encrypted to the server.

## 4.6 Discussion

A new class of social application uses the users' actual location to provide personalised recommendation and allow for new interactions especially in urban settings. We have shown how these applications can expose their users to different privacy attacks that can be easily overlooked.

We have analysed a set of popular dating application, and observed how proximity-based social applications have access to certain identity information that could lead a possible privacy attacker to easily identify users on Facebook and link their online profiles to physical identities.

Furthermore we have shown how users constantly sharing their relative distance to other users can be *followed* by an attacker in their movement without their knowledge. We have demonstrated how this information can be used for a multilateration attack with arbitrary precision. There is in fact no restriction to the number of distance samples that a possible attacker might be able to measure.

We followed a formal framework to identify the classes of privacy violation to which users are subjected to without being aware of it and we have shown how these violations can all be carried out for the applications examined.

This shows how using third party profiles to provide access to a specific applications may cause a security *honey pot* for a possible attacker.

We have also stressed how In order to make the registration process easier, these applications often leverage on third party services to provide a login mechanism, while at the same time acquiring certain private information about their new users. The third parties used are often services such as Facebook or Google, and the in-

formation accessed concern the public profile of the users on such platforms.

While this technique certainly allows people to quickly sign up to an application and create a new profile, it also creates different privacy threats for users of such services. Primarily, it concerns who can gain access to such data and how information shared with third parties can also be stored and eventually transferred without the user explicit consent.

We have then used Facebook graph search to build a hyper graph of the user identity starting from few information that were shared through a third application. This shows how each information can be used as a selector to further identify a different piece of the whole user identity and can be used to target the user in real life.

*There will come a time when it isn't 'They're spying on me through my phone' anymore. Eventually, it will be 'My phone is spying on me'.*

Philip K. Dick

# 5

# Web tracking: how advertising networks collect users' browsing patterns

In the early age of the Internet users enjoyed a large level of anonymity. At the time web pages were just hypertext documents; almost no personalisation of the user experience was offered. The Web today has evolved as a world-wide distributed system following specific architectural paradigms. On the web now, an enormous quantity of user generated data is shared and consumed by a network of applications and services, reasoning upon users expressed preferences and their social and physical connections.

This chapter is focused on web users tracking and advertising networks, extending work presented in [**? ? ?**].

Advertising networks follow users' browsing habits while they surf the web, con-

tinuously collecting their traces and surfing patterns, since advertising sustains the business model of many websites and applications. Efficient and successful advertising relies on predicting users' actions and tastes to suggest a range of products to buy. Both service providers and advertisers try to track users' behaviour across their product network. For application providers this means tracking users' actions within their platform. For third-party services *following* users, means being able to track them across different websites and applications. It is well known how, while surfing the Web, users leave traces regarding their identity in the form of activity patterns and unstructured data. These data constitute what is called the user's online footprint. We analyse how advertising networks build and collect users footprints and how the suggested advertising reacts to changes in the user behaviour.

## 5.1 Background

Websites use *personalisation services* to provide a tailored experience to their visitors. In order to make their product more personal to the single users they need to keep profiles of their users, collect their in page reading activities and eventually their preferences. This data is then shared to third-party services, accessed and analysed without users' direct consent. Furthermore, records of users' activities are used for different purposes, most unknown to the end user, such as marketing or to provide analytics services back to the original website or application. Among the data analysed by websites are also included user preferences and social connections. These can be obtained by tracking users across different applications and sites through cookies or open web sessions. Even if the user does not accept cookies or is not logged into a service account, such as their Google, Twitter or Facebook accounts, the web page and third-party services can still try to profile them by using third-party HTTP requests, among other techniques. Within the HTTP request various selectors can be included to communicate user preferences or particular features, in the form of URL variables. Features that might be used by advertising networks and malicious trackers include personalised language or fonts settings, browser extensions, in page keywords, battery charge and status, and

so on. These features are then used to identify individual users by restricting the pool of possible candidates among all the visitors in a certain time frame, location, profile of interests. Unique users can then be distinguished across multiple devices or sessions.

In this chapter, have observed how users are tracked across the Web and how the displayed advertising is tailored even after they have visited a few websites with a certain interest bias. In previous work [**?** ] [**?** ] we analysed how third-party advertising services are able to profile users on a short series of websites visited and how these are able to *follow* users while they surf the web. In our study we analyse how the user profile detected by advertising services can be used to estimate the user privacy risk on a certain network. We analyse how advertising networks identify a user and start tracking them, by considering keywords contained in the web-page and understanding the underlying network structure of tracked domains. We measure the distance between the observed user profile and the actual user profile, by categorising the set of keywords contained in web pages and by capturing third-party HTTP requests. We introduce a set of metrics to express this distance between the two profiles.

It is important to note that we have considered the case for which users are not registering, neither connecting any external account, as it could be the case with services like: Facebook, Google+, Twitter, and so on. In such scenario we have measured how these networks still attempt to track the user by sending user information through HTTP requests to their services.

We present a model of the user profile that is able to capture how each website and tracking network categorise their activities in terms of interests and interactions.

Therefore, we analyse how much information is sent by each page visited to third-party services by measuring the partial user profile and the actual user profile. The partial user profile is what the website and third-party services know about the user. The actual user profile is instead the full profile measured at the end of the series of page visited.

We then, introduce a set of metrics to express the relationship between the par-

tial and the actual user profile.

Hence, we profile third-party HTTP calls sent by Facebook tracking services and compare this to the the user actual profile.

Finally, we model user online footprints as a graph of the actions generated by each user and analyse the resulting graph structure, identifying known malicious trackers.

## 5.2 Modelling the user profile

Each time the user visits a new page, we aggregate the page keywords and build what we consider the user's profile of interests (Fig. 5.3.3). We consider a tractable model of the user profile as a probability mass function (PMF), as proposed in [**?** **?** ], to express how each keyword contributes to expose how many times the user has indirectly expressed a preference toward a specific category. We consider that the user expresses a preference when they visit a webpage categorised with certain keywords. This model follows the intuitive assumption that a particular category is weighted according to the number of times this has been counted in the user profile.

We define the profile of a user as the PMF $p = (p_1, \ldots, p_L)$, conceptually a histogram of relative frequencies of tags across the set of tag categories $\mathcal{T}$. This means that we group tags around interests using top level categories as defined by the Open Directory Project (DMOZ) [**?** ]. The user profile is calculated at the end of the series of website visited by the user. Similarly we define the partial user profile at moment as this is known to the advertising network as $\hat{p} = (\hat{p}_1, \ldots, \hat{p}_L)$.

Note that, for the case when an advertising network is present on each and every page, $\hat{p} = p$ at the end of the series of sites visited. This means that the network was able to record each page visited by the user. This could easily be the page of advertising networks like Google that through different third-party services are ubiquitously present across the web.

We also define the profile of an ad, or third-party HTTP request as the PMF $q = (q_1, \ldots, q_L)$, where $q_l$ is the percentage of tags belonging to the category $l$

which have been assigned to this specific advertising item. You can think of the ad profile as the PMF of the tag contained in every HTTP request sent from the visited page to the advertising network (Listings: 5.1, 5.2, 5.3). This profile notes which tags the tracking network is using to identify the user and display some advertising content. Note that the ads profile, is calculated independently for each advertising network.

Both user and ads profiles can then be seen as normalised histograms of tags across categories of interest. Our profile model is in this extent equivalent to the tag clouds that numerous collaborative tagging services use to visualise which tags are being posted, collaboratively or individually by each user. A tag cloud, similarly to a histogram, is a visual representation in which tags are weighted according to their relevance.

In view of the assumptions described in the previous section, our privacy attacker boils down to an entity that aims to profile users by representing their interests in the form of normalised histograms, on the basis of a given categorisation.

We consider the third-party advertising network to operate like a recommendation system that suggest products or services that might be of interest for the user, based on their preferences. A recommendation system can be described as an information filtering system that seeks to predict if the user is interested or not in a particular resource. We assume that the ad server suggests advertising based on a measure of *similarity* between what the user *does* and what the network *knows*. Furthermore, we consider tracking service to work in a feedback loop (Fig. 5.2.1). When a user surfs the web each tracker on the visited pages communicates with the advertising service, sending a number of parameters through HTTP requests. These contain the user preferences and browsing history which will be taken into consideration when ads are returned to display on the page.

It is important to note that while it is safe to consider an advertising network as a recommendation system, we should also consider that a number of processes and interactions between the advertising networks, the website, and the ultimate advertiser, can influence the actual recommendation that is displayed to the user. Tracking services can in fact follow different strategies to recommend products

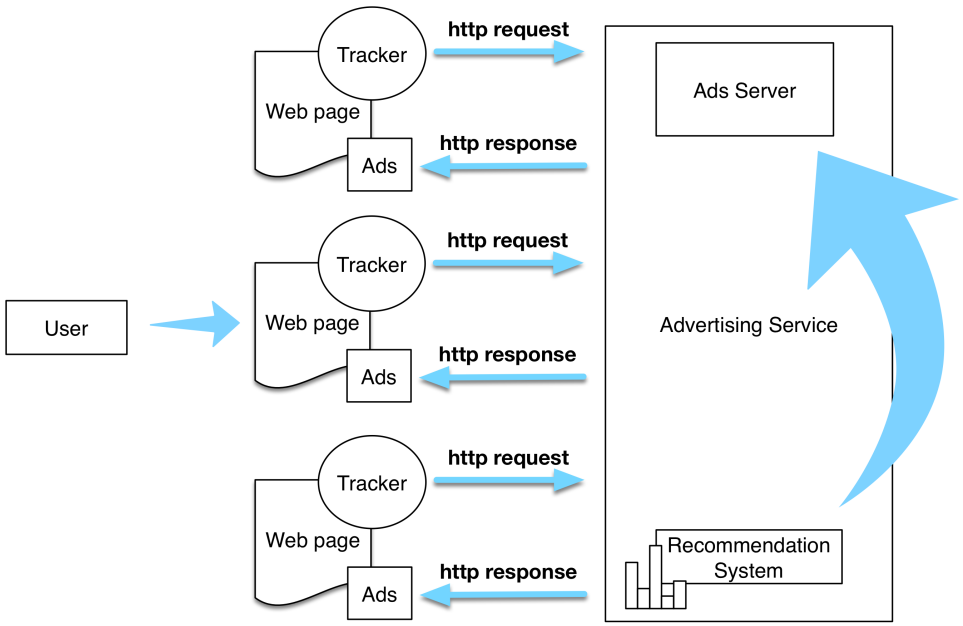



**Figure 5.2.1:** Advertising services work in a feedback loop. The image illustrate how while a user surf a number of web pages, the service record their profile and adapts the returned advertising.

to users. Some services display in page advertising where a bidding mechanism allow advertisers to compete for categories and spaces, other services might decide to target only specific categories, others might instead decide to target the visited page only.

We measure the user profile, as previously described, as a histogram of their recorded preferences, and the advertising profile as a histogram of the ads that the user has received. We have considered a set of metric to measure how the advertising network is tracking the user profile, and how a page sends information to a tracking service by transmitting a partial user profile.

In previous works[**?** ] [**?** ] we used the *1-norm*, *2-norm* as measures of how the advertising profile, or the partial user profile, approximates the user profile. Please recall that the partial user profile is calculated by a given advertising network at a given moment on a series of pages visited.

We now introduce the normalised *a-norm* as the generalised variation $\mathcal{GV}$ between two probability distributions, the partial and the genuine user profiles. Furthermore, we will introduce the *KL-divergence* as a measure of how the partial profile approaches the genuine user profile. Please note that while we are defining our metrics between the partial user profile and the genuine user profile, the same assumptions holds, without loss of generality, if we compare the user's and the advertising profiles.

### Norm and generalised variation

We define the $a-norm$ between the partial profile as observed by an advertising platform and the genuine user profile as:

$$\|p - \hat{p}\|_a = \sqrt[a]{\sum_l |p_l - \hat{p}_l|^a} \quad \text{with} \quad a \in [1, \infty]$$

The case for $a = \infty$ is defined in the limit:

$$\lim_{a\to\infty} \|p - \hat{p}\|_a = \lim_{a\to\infty} \sqrt[a]{\sum_l |p_l - \hat{p}_l|^a} = \max_l |p_l - \hat{p}_l|$$

The $a-norm$ is a distance in $\mathbb{R}^L$ with the following properties:

- Absolute homogeneity.
- Positive definite: $\|p - \hat{p}\|_a \geq 0$ and $\|p - \hat{p}\|_a = 0 \Leftrightarrow p = \hat{p}$
- $\|p - \hat{p}\|_a = \sqrt[a]{2} \Leftrightarrow p$ and $\hat{p}$ are orthogonal deltas.
- Triangle equality.

For $a = 1$ we define the *1-norm* between the partial and the genuine user profiles as:

$$\|p - \hat{p}\|_1 = \sum_l |p_l - \hat{p}_l|$$

The *1-norm* represent the average discrepancy between the two profiles. For $a = 2$ we define the *2-norm* as:

$$\|p - \hat{p}\|_2 = \sqrt{\sum_l |p_l - \hat{p}_l|^2}$$

The *2-norm* represents the Euclidean distance between the two distributions. When considering the *2-norm* it is possible to highlight larger discrepancies on the set of categories analysed. An interesting property is that *norms* are also nested. Hence:

$$\|p - \hat{p}\|_\infty \leqslant \|p - \hat{p}\|_2 \leqslant \|p - \hat{p}\|_1$$

We hence define the generalised variation $\mathrm{GV}(p, \hat{p})$, based on $a$-norm as:

$$\mathrm{GV}(p, \hat{p})_a = \frac{1}{\sqrt[a]{2}}\|p - \hat{p}\|_a = \frac{1}{\sqrt[a]{2}}\sqrt[a]{\sum_l (p_l - \hat{p}_l)^a}, \quad a \in [1, \infty]$$

The coefficient $\frac{1}{\sqrt[a]{2}}$ normalises the range of values of the $a-norm$ in $[0, 1]$. Therefore, $\mathrm{GV}(p, \hat{p})$ is a norm, is positive definite, absolutely homogeneous and satisfies the triangle inequality:

- $\mathrm{GV}(p, \hat{p}) \geq 0$ with equality if and only if $p = \hat{p}$
- $\mathrm{GV}(p, \hat{p}) \leq 1$ with equality if and only if $p$ and $\hat{p}$ are orthogonal canonical vectors (discrete deltas).

Note that for $a = 1$ the generalised variation $\mathrm{GV}(p, \hat{p})$ is equal to the total variation $\mathrm{TV}(p, \hat{p})$:

$$\mathrm{TV}(p, \hat{p}) = \frac{1}{2}\|p - \hat{p}\|_1 = \frac{1}{2}\sum_l |p_l - \hat{p}_l|$$

For $a = 2$ we have the normalised *2-norm*:

$$\mathrm{GV}_2 = \frac{1}{\sqrt{2}}\|p - \hat{p}\|_2 = \frac{1}{\sqrt{2}}\sqrt{\sum_l |p_l - \hat{p}_l|^2}$$

Finally, the case for $a = \infty$, $\lim_{a\to\infty} \mathrm{GV}(p, \hat{p})_a = \|p - \hat{p}\|_\infty$ between $p$ and $\hat{p}$. The reason is that for $a \gg 1$ the greatest difference dominates in the sum $\sum_l |p_l - \hat{p}_l|^a$. Therefore $\lim_{a\to\infty} \|p - \hat{p}\|_a \approx \max_l |p_l - \hat{p}_l|$.

One might interpret these norms as $a = 1$ been an average-case metric, $a = \infty$ being a worst-case scenario, and $a = 2$ a robust middle ground.

### KL-Divergence

Now we propose and justify an information-theoretic quantity as a measure of how the partial profile approaches the genuine user profile: the *KL-divergence*. Suppose that we might interpret the profile $\hat{p}$ observed by a third-party tracking service, as a sequence of $L$ independent, identically distributed, drawings of a user's genuine profile of interest $p$. Then in accordance with the rationale proposed in [**?** ] [**?** ], we may argue that the probability $p(\hat{p})$, of a given observed profile is related to the KL-divergence between the empirical observation $\hat{p}$ and the ideal one $p$, as follows:

$$-\frac{1}{L}\log p(\hat{p}) \xrightarrow[L\to\infty]{} \mathrm{D}(\hat{p}\|p)$$

Informally this means $p(\hat{p}) \approx 2^{-\mathrm{L}\,\mathrm{D}(\hat{p}\|p)}$. Note that small divergences will lead to likely outcomes, whereas large divergence associate with rare events.

Note also that $\hat{p}$ is absolutely continuous with respect to $p$: $p_l = 0 \Rightarrow \hat{p}_l = 0$. Also $\hat{p} \ll p \iff \mathrm{D}(\hat{p}\|p) < \infty$.

## 5.3 Modelling the user's online footprint

We model the user's activity as series of events belonging to a certain identity. Each event is a document containing different information. An event corresponds to an action generated by the user or one of their devices. When a user visits a website

or creates a post on a blog, an event is created. We can think of an event as a hypermedia document i.e. an object possibly containing graphics, audio, video, plain text and hyperlinks. We call the hyperlinks selectors and we use these to build the connections between the user's different identities or events. Each identity can be a profile or account that the user has created onto a service or platform, or just a collection of events, revealing something about the user. With account we mean an application account or a social network account, such as their LinkedIn or Facebook unique IDs. When the user visit a web page, or uses a web or mobile application, a series of events is generated and associated with the account. Some of these events are created by direct user's actions, others are created by code triggered indirectly by the user.

While the user visits a webpage and reads its content a series of snippets of code and client side scripts are executed and information is transmitted to the page backend or some third-party server. Among the information transferred are a number of user preferences. These can be their geographical location, battery level of their current used device, browser preferences, or just their browsing history captured up to that point. Some or all of the meta and in page keywords used to describe the page are also transferred. We build the user profile by collecting the meta keywords expressed in web pages. We consider this a subset of the possible set of preferences that third-party advertising networks might be interested in collecting.

### 5.3.1 Proposing a model of third-party requests on web pages

When a user visits a web page, the browser sends an HTTP request to the server to request a representation of the resource described through the URL. The server provides the resource representation in the form of a HTML document and the browser parses it. The HTML document contains a number of links to other resources, such as JavaScript code, videos, audios or images (Fig. 5.3.1). Some of these can be stored on the same domain as the requested page, some may be requested to third-party services. Such is the case of services like Google Analytics, share buttons from different social networks, or advertising banners. Together

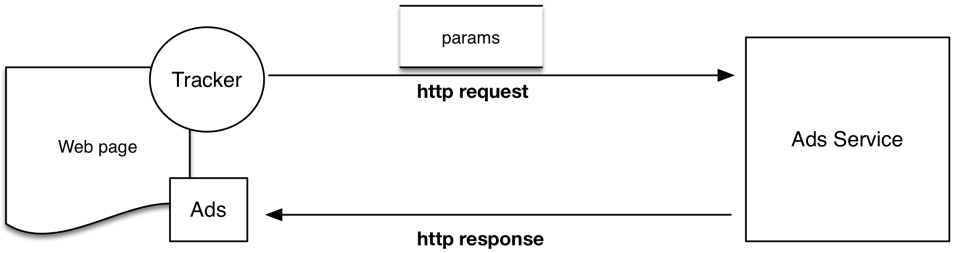



**Figure 5.3.1:** Trackers on web pages make third-party HTTP requests to advertising services. These return ads content tailored to the user web history or expressed preferences.

with the HTTP request, a number of parameters are included. These contains keywords, users? preferences, information regarding the user device and session, in page information sent to the third-party service from the website or application.

When a third-party request is performed by the visited page, we store the parameters passed and if the call belongs to a known tracking network we categorise the corresponding keywords. Also when a request is made, we store a direct link between the page and a tracking domain, such as *google.com*. This results in a graph model of tracking networks and how these are connected to pages (Fig. 5.3.2). The graph model allows us to understand the underling network structure of tracking networks and how these are pervasively following users across their visits. In fact, every time we discover which tracking services are active on a certain website, we can create an indirect link between the user and the tracker.

### 5.3.2 Network structure metrics

We said that advertising networks or privacy attackers need to be able to *follow* the user across as many websites as possible in order to profile their interests. This naturally translates onto a graph model where each page is directly connected to its active trackers (Fig. 5.3.2). We therefore considered a set of metrics that can uncover the underlying network structure of tracking service. The first of the metrics considered is the average degree of the neighbourhood. The average degree

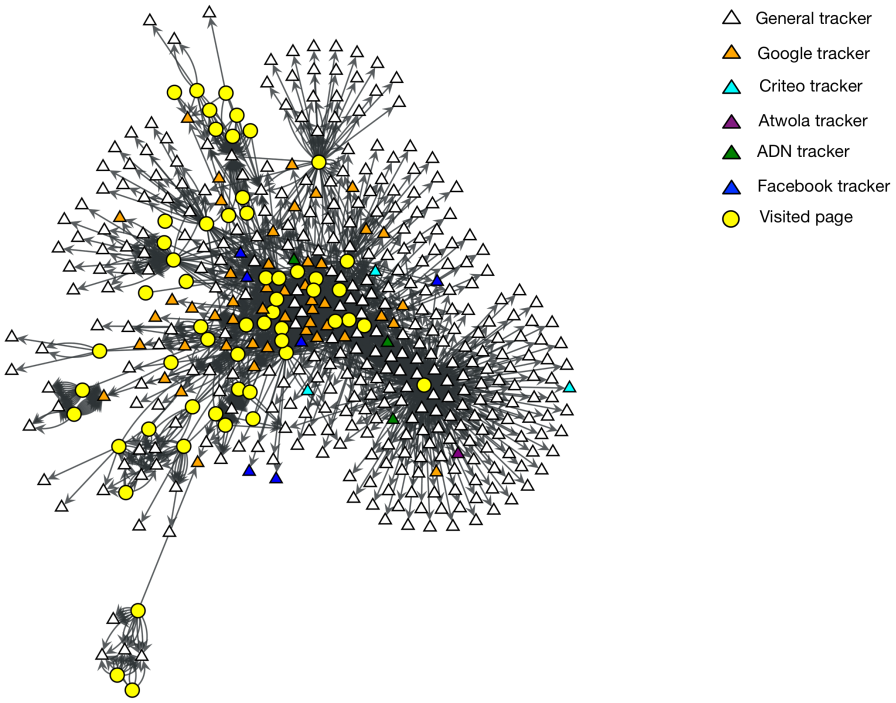



**Figure 5.3.2:** The graph shows how known trackers are connected to visited pages and therefore how these are able to follow users across different websites.

of the neighbourhood of each node is a good indication of how many pages are connected to a certain advertising service or tracking domain.

The average degree of the neighbourhood of a node $i$ is calculated as:

$$\langle k_{nn,i} \rangle = \frac{1}{|N(i)|} \sum_{j \in N(i)} k_j$$

Where $N(i)$ are the neighbours of node $i$ and $k_j$ is the degree of node $j$ which belongs to $N(i)$.

If a certain tracker domain is connected to the majority of the page visited by a certain user, this means that they have been able to collect the user's preferences and reading activities across a number of websites. The more a tracker domain is connected, the more the user might consider this a *risk* for their privacy. We used the average degree of the neighbourhood of a tracker to rank tracker domains.

To describe the resulting network structure, we also calculated the average scalar assortativity coefficient [**?** ] defined as:

$$r = \frac{\sum_{xy} xy(e_{xy} - a_x b_y)}{\sigma_a \sigma_b}$$

Where $a_x = \sum_y e_{xy}$ and $b_y = \sum_x e_{xy}$, and $e_{xy}$ is the fraction of edges from a vertex of type x to a vertex of type y.

We also generated a partition of SBM and nested SBM of the resulting graph employing an agglomerative multilevel Markov chain Monte Carlo (MCMC) algorithm as described in [**?** ] [**?** ][**?** ]. The idea behind using SBM to describe the network structure of identified trackers is to be able to identify similar trackers and to understand if trackers belonging to the same domain or that exhibit similar behaviour can be grouped based on network properties.

We profiled 50 users and each user visited a series of 100 pages. In total we analysed 5000 different pages (Table: 5.3.1). For each user we calculated how each page contributed to the user profile and also how third-party services adapted to the user profile by returning certain information in form of ads. Information

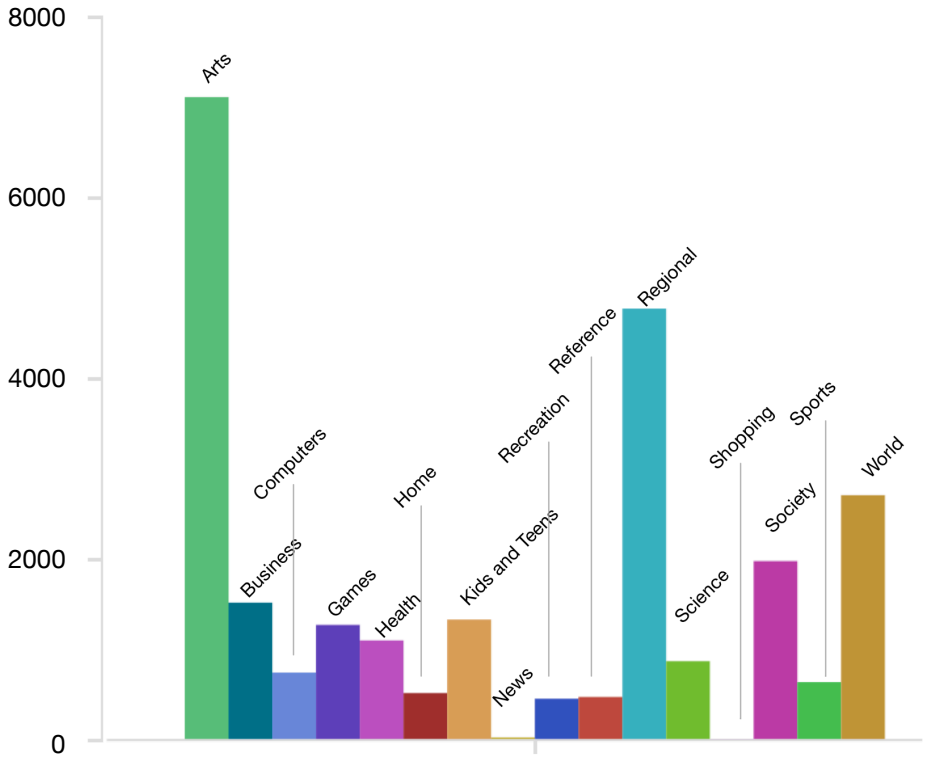



**Figure 5.3.3:** Here we show an example of user profile expressed in absolute terms by counting the number of keywords in each category for a browsing session. We model user and advertising profiles as histograms of tags keywords a set of predefined categories of interest.

**Table 5.3.1:** Statistics regarding collected users data

| Statistics about collected data | | | |
|---|---|---|---|
| Categories | 16 | Users | 50 |
| Pages per users | 100 | Total pages | 5000 |

that advertising services request from the visited page may vary in length and type (Listings 5.1, 5.2, 5.3). Some trackers might include only the referrer *url* and some devices information and user triggered parameters (Listings 5.1, 5.3) while other services might be more lengthily in what is sent from the page (Listing 5.2). Some of the information sent through third-party request cannot be categorised, since they include hashed users? ids and internal keywords and code belonging to the tracking service. Other information, like the keywords retrieved from the page (Listing 5.2) can be categorised into category of interest that we assume the tracking service uses to profile the user through their interests.

It is interesting to not how among the parameters sent to the third-party tracking services are not included just in page keywords regarding the topic of the page, but also specific browser information. Some of the device's preferences are included in the HTTP headers, like the user-agent identifying the browser and the Operative System. Other information regard how long the page took to load or how soon the content was ready (Listing 5.2).

```
Host: aax.amazon-adsystem.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.11; rv:48.0)
Gecko/20100101 Firefox/48.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Referer: HTTP://www.nytimes.com/2016/08/29/us/politics/donald-trump-congress-gop-voters.HTML?hp
Params:
  action: click
  pgtype: Homepage
  clickSource: story-heading
  module: first-column-region
  region: top-news
  WT.nav: top-news
  _r: 0
Cookie: ad-id=A8rOwZ2wOUK4gka1zjqyWNo; ad-privacy=0
Connection: keep-alive
```

**Listing 5.1:** A third-party request to Amazon Ads Service from the nytimes.com homepage. In this example keywords are sent directly as parameters in the HTTP request.

```
GET /pixel.gif?
Params:
  source: smarttag
  _kcp_s: nytimes
  _kcp_sc: us
  _kcp_ssc: politics
```

```
    _kcp_d: www.nytimes.com
    _kpref_: HTTP://www.nytimes.com/
    _kua_kx_lang: en-us
    _kua_kx_tech_browser_language: en-us
    _kpa_page_type=article
    _kpa_cg: us
    _kpa_scg: politics
    _kpa_pst: News
    _kpa_des: Presidential Election of 2016
    _kpa_per: Lujan Ben Ray
    _kpa_org: Republican Party
    _kpa_author: Alexander Burns and Jonathan Martin
    _kpa_keywords2: Presidential Election of 2016 Elections House of Representatives Politics Action Committees Elections Senate Republican
        Party Lujan Ben Ray Issa Darrell Trump Donald
    t_content_ready: 1792
    t_window_load: 12720
    ...
Host: beacon.krxd.net
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.11; rv:48.0) Gecko/20100101 Firefox/48.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Referer: HTTP://www.nytimes.com/2016/08/29/us/politics/donald-trump-congress-gop-voters.HTML?hp&
      action=click&pgtype=Homepage&clickSource=story-heading&module=first-column-region&region=top-
      news&WT.nav=top-news&_r=0
Cookie: ServedBy=beacon-a262-dub; _kuid_=DNT
Connection: keep-alive
```

**Listing 5.2:** A third-party request to krxd.net from a nytimes.com article. This request send different information regarding the article and the browser preferences through HTTP parameters. In addition to the keywords associated with the page, we can see how the request includes information regarding how long it took for the content to be ready *param* : $t_{c}ontent_{r}eady$ as well as how much it took for the browser window to load *param* : $t_{w}indow_{l}oad$.

```
Host: graph.facebook.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.11; rv:48.0) Gecko/20100101 Firefox/48.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
DNT: 1
Referer: HTTP://www.independent.co.uk/news/uk/politics/europe-could-go-down-the-drain-after-brexit-
      a7213976.HTML
Cookie: datr=TbHdVa-yyYq_3UHH_xYR6NGb; fr=0MuKlsg7QM3etJaWt.AWVJMdGky_V9X82TY03Y-wBtGqE.BV3bFx.XF.
      FfD.0.0.BXw9oU.AWVWPpAJ; lu=TggRyE6qvvdCystV9I2G-bow; _ga=GA1.2.1182524233.1441193978; sb=
      i14HV4ufa1WguCxPntCQagPo; c_user=100007394807876; xs=192%3AnWrYMasjLyLusw%3A2%3A1365011662%3
      A5189; csm=2; s=Aa4hJAWUyEoHSY_M.BXj1Ln; pl=n; p=-2; act=1472453154239%2Fo; presence=
      EDvF3EtimeF1472453144EuserFA21B07394807876A2EstateFDutF1472453144216Et2F_5b_5d
    Elm2FnullEuct2F1472410836BEtrFA2
    loadA2EtwF240195646EatF1472453143045CEchFDp_5f1B07394807876F2CC
Connection: keep-alive
```

**Listing 5.3:** A third-party request to facebook.com from a indipendent.co.uk article.

Once we were able to collect and profile readable keywords from HTTP requests, we wanted to know how each page contribute to *how much tracking services know* about our genuine user profile by observing a series of web pages visited. For each users we calculated the *TV*, the $GV_2$, the $\infty$-*norm* and the *KL-divergence* between the partial and the genuine user profile (Fig. 5.3.8). The metrics were calculated for 80 visited, while the genuine user profile was calculated over a series of 100 visits. Therefore, in our scenario, if a tracker is present in each visited page they would *know*, in the worst case scenario 80% of the visited pages. Note that the *TV* gives a measure of the average discrepancy between the probability distributions, while the $\infty$-*norm* gives the worst case scenario. From our results we see that the worst case scenario and the average one behave similarly.

We have then analysed the case of a tracker that is not present in each of the visited pages. We considered the facebook third-party requests to their services for this experiment. For each user we calculated the *TV*, the $GV_2$, the $\infty$-*norm* and the *KL-divergence* between the partial and the genuine user profile (Fig. 5.3.9) for pages where the tracker is present.

Finally, we profiled keywords in third-party HTTP requests to Facebook. We wanted to know what information were sent to Facebook for each page visited where the tracker was present. This is important to understand what trackers are able to capture about users' preferences if they are not able to *follow* the user across all the pages visited. We assumed that if a tracker is not present on a page, they have no knowledge the user visited it, therefore the partial profile as it is known to the tracker is not modified.

Note also that although none of the users considered in our experiment where logged into Facebook, web pages consistently send data to their third-party tracking services. This means that users are profiled by Facebook even if these are not logged in their platform, and individuals that have decided to opt out of Facebook continue to be targeted and known to their services. This is evident by the request shown on listings 5.3. A number of browser and device specific information is collected by the HTTP call although the user isn't connected to Facebook. For each users we calculated the *TV*, the $GV_2$, the $\infty$-*norm* and the *KL-divergence* between

the advertising profile $q$ and the genuine user profile $p$ (Fig. 5.3.10) for pages where the tracker is present. We considered a shorter series of pages (15 pages) following the intuition that advertising networks might try to form a profile of the user instantly given a small number of visits to similar pages. This was consistent with previous results obtained [**?** ].

We have also analysed network structure among the discovered trackers. By using our footprint model we also considered how tracker domains are linked to pages. In this case we calculated the average degree of the neighbourhood of each node, for nodes corresponding to advertising services. Our results show how it is possible to identify known tracker domains by measuring the average degree of the neighbourhood (Table: 5.3.2).

Considering the average degree of the neighbourhood of each node, we can also find out about some interesting properties of the network. We started considering the *in-degree distribution* of the network (Fig. 5.3.4). The degree distribution $P(k)$ of a network is defined as the fraction of nodes in the network with degree $k$. It is particularly interesting to note that the network *in-degree distribution* approximately follow a power law.

Another interesting property to consider is *assortativity*. Assortativity considers the conditional probability that a node of degree k is connected to a node with degree k'. If the probability function is increasing, the network is said to be assortative, showing that nodes of high degree are more likely to connect to nodes of high degree. If the function is decreasing the network is dissortative, meaning nodes of high degree are more likely to connect to nodes of lower degree. We found that the scalar assortative coefficient for the resulting graph is of -0.19 a value that is often found for internet systems [**?** ].

We also generated a partition of SBM and nested SBM of the resulting graph (Figs. 5.3.5 5.3.6). Here we identified how the resulting network partitions and communities corresponds to know tracker domains. This means tracking service exhibit similar properties across the same domain and its structure can be identified through statistical inference over the graph.

| Tracker domain | avg $k_{nn,i}$ |
|---|---|
| tacoda.at.atwola.com | 180.0 |
| bcp.crwdcntrl.net | 180.0 |
| match.prod.bidr.io | 180.0 |
| glitter.services.disqus.com | 180.0 |
| ad.afy11.net | 180.0 |
| idsync.rlcdn.com | 180.0 |
| mpp.vindicosuite.com | 180.0 |
| aka-cdn-ns.adtechus.com | 180.0 |
| clients6.google.com | 180.0 |
| i.simpli.fi | 180.0 |
| ads.p161.net' | 180.0 |
| dis.criteo.com | 180.0 |
| ads.stickyadstv.com | 180.0 |
| cms.quantserve.com | 180.0 |
| ads.yahoo.com | 129.0 |
| graph.facebook.com | 118.0 |
| ib.adnxs.com | 110.0 |
| rs.gwallet.com | 108.0 |
| bid.g.doubleclick.net | 98.333 |
| googleads4.g.doubleclick.net | 98.333 |

**Table 5.3.2:** The table shows the top 20 identified tracker domains based on the average degree of the neighbourhood.

Furthermore we also computed page-rank algorithm among the network and identified most connected tracked domains (Fig. 5.3.7). Again we were able to spot known tracker domains.

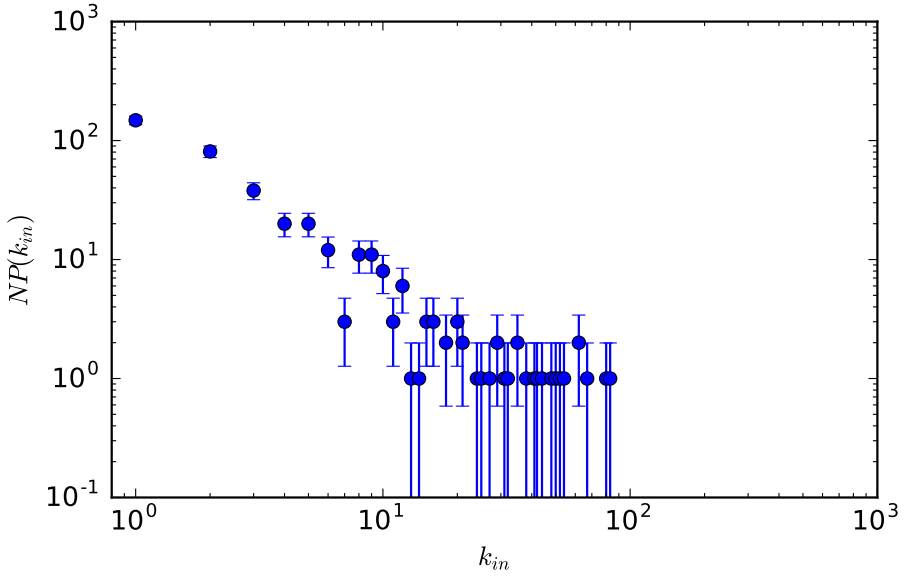



**Figure 5.3.4:** Degree distribution for the network resulting from the footprint model of users activity. We can see how the degree distribution follows a power law.

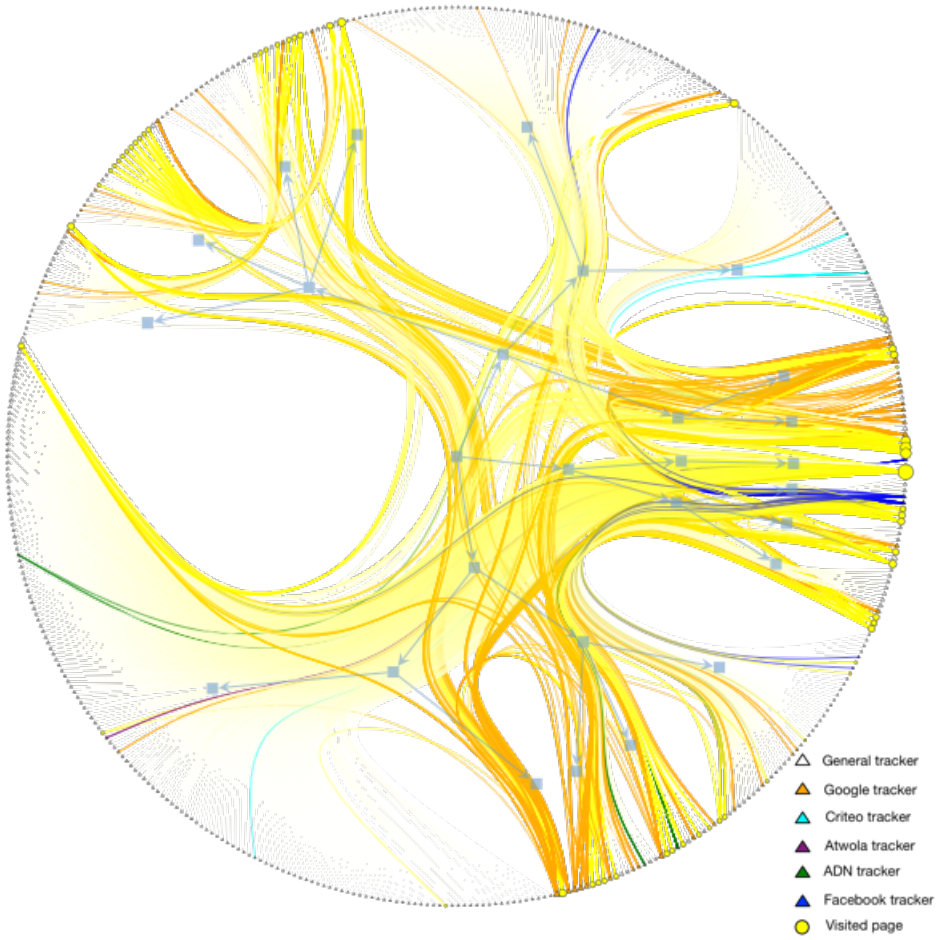



**Figure 5.3.5:** Block-model decomposition of the network. We can see how we can identify known tracker networks, and how trackers can be grouped into communities that exhibits similar network structure. The blue squares represent the block partition of the network. While the legend is referred to the original nodes, here represented by the shapes on the border.

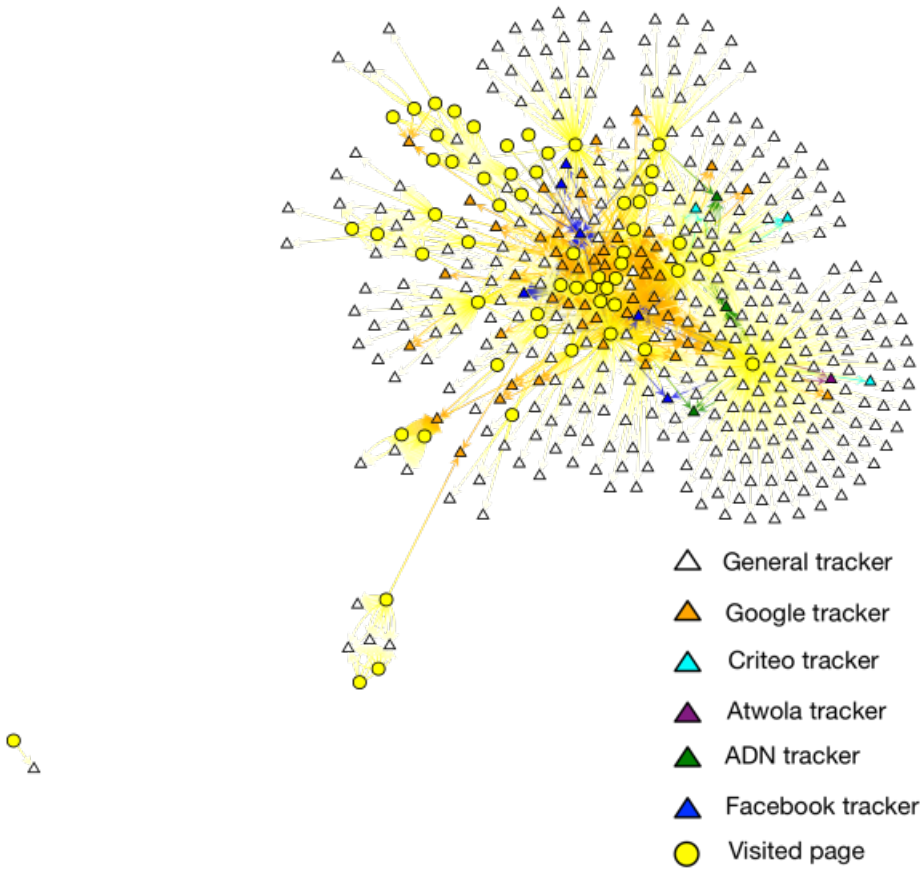



**Figure 5.3.6:** Blockstate representation of the network of tracking service resulting from our simulation. Here we highlight connections between known tracker networks and visited page.

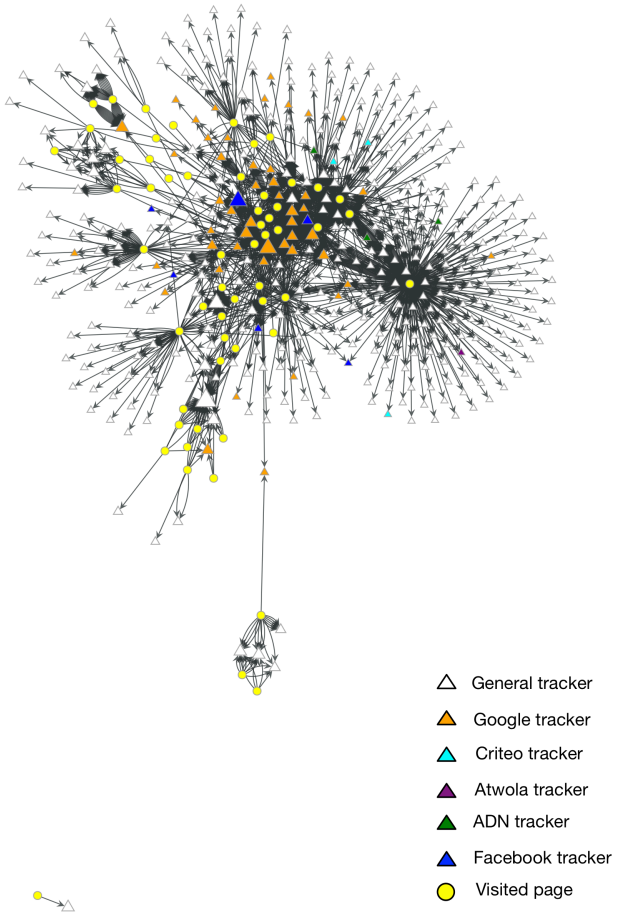



**Figure 5.3.7:** Pagerank computed over the tracking network. Known tracker domains that are *more connected* can be seen with a bigger node symbol compared to less connected ones.

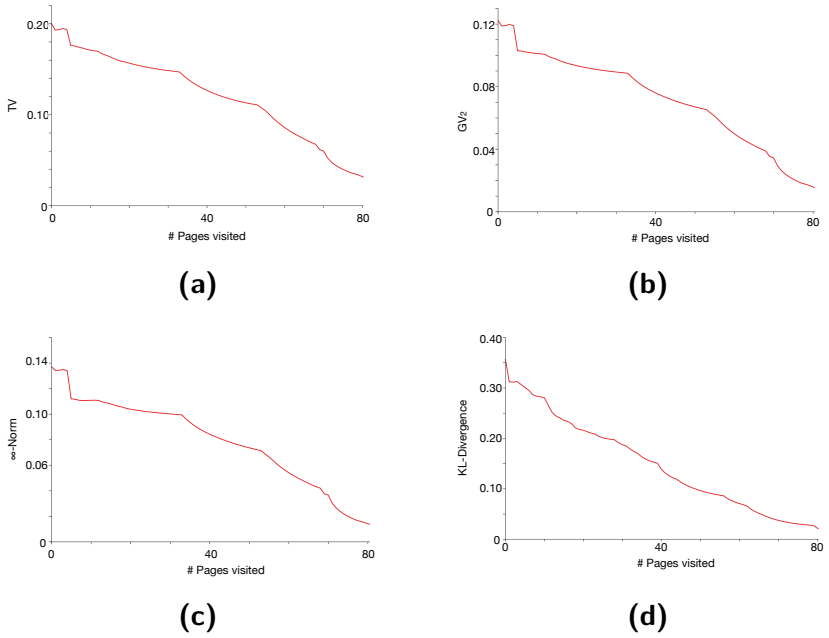



**Figure 5.3.8:** The figures show how each page visited contribute to the actual user profile. Please recall that we calculated the user profile at the end of the series of 100 web pages visited and we calculated the metrics for 80 visits, giving a 80% estimation. We therefore computed the $\mathcal{TV}$(a), the $\mathcal{GV}_2$(b), the $\infty-\backslash\wr\nabla\Updownarrow$(c) and the *KL-divergence*(d) for all pages and averaged among all users.

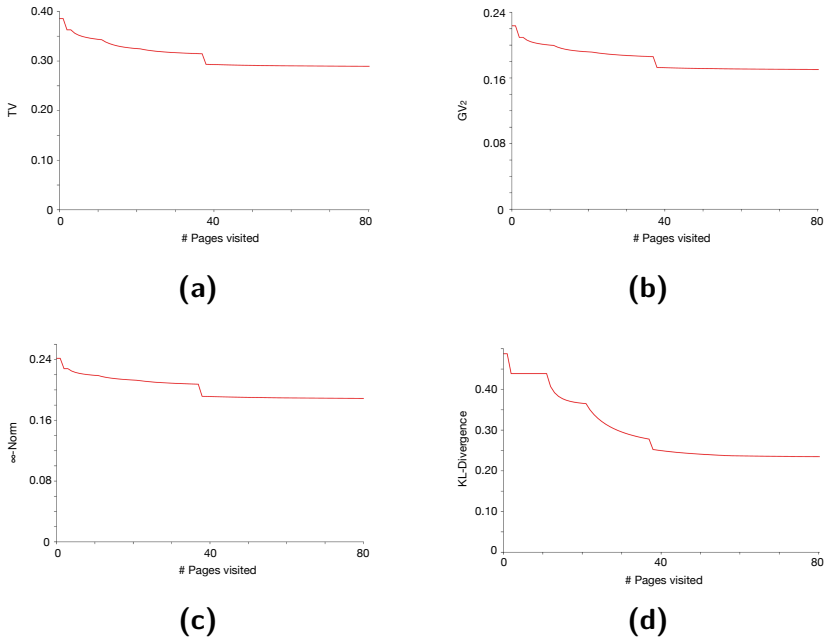



**Figure 5.3.9:** The figure show the relation between the profile captured by third-party requests to Facebook services and the actual user profile. Please recall that we calculated the user profile at the end of the series of 100 web pages visited and we calculated the metrics for 80 visits, giving a 80% estimation.We therefore computed the $\mathcal{TV}$(a), the $\mathcal{GV}_2$(b), the $\infty-\backslash\wr\nabla\Updownarrow$(c) and the *KL-divergence*(d) for all pages and averaged among all users.

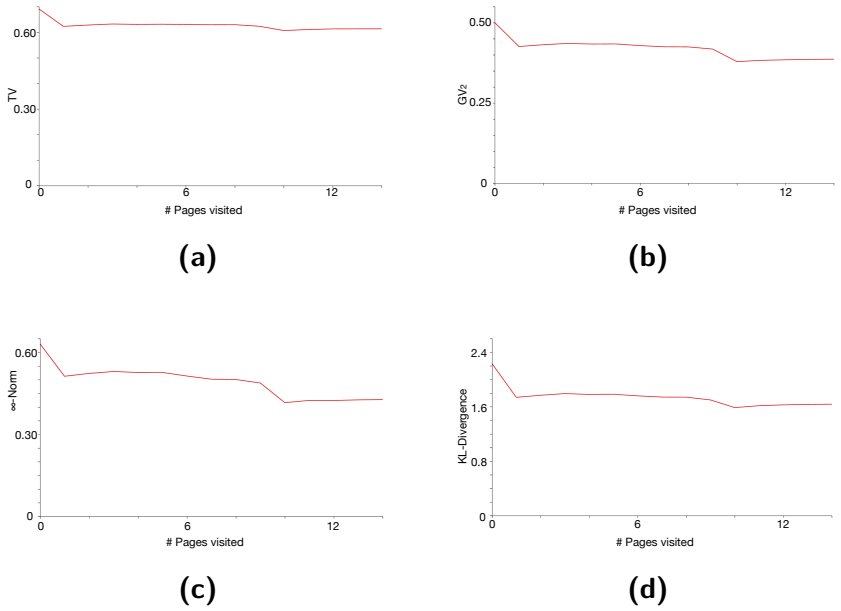



**Figure 5.3.10:** The figure show the relation between the profile sent by third-party requests ($q_n$ with $n \in [1, N]$) to Facebook services and the actual user profile. Please recall that we calculated the user profile at the end of the series of 15 web pages visited. We therefore computed the $\mathcal{TV}$(a), the $\mathcal{GV}_2$(b), the $\infty-norm$(c) and the *KL-divergence*(d) for all HTTP calls and averaged among all users.

## 5.4 Discussion

We introduced a set of metrics to show how information is sent to third-party tracking services when users surf the web. Because we considered users that were not logged into any identity account, such as Twitter, Google+ or Facebook, we show how third-party service were still able to collect valuable information. We computed the set of metrics for the partial user profile at each page visited. This shows how each page contribute to the actual user profile at the end of a series of websites visited. This means that an advertising network that is present on most of the pages visited possess a large amount of information regarding users and population of users. This information finally allows networks to predict fairly quickly user's preferences and behaviour. We also computed a set of network analysis on our graph model of the user online footprint. We were able to identify known trackers and isolate communities of similar trackers. This aspect is particularly interesting for the development of Privacy Enhancing Technologies for the web. Up to now, anti-tracking technologies have been built to simply stop third-party requests, alternative strategies might instead consider to send bogus information to certain over-connected tracker domains to masquerade the user real profile. At the same time a measurement of the average degree of the neighbourhood of a certain third-party domain can be used to evaluate how *dangerous* this can be considered for the user's privacy.

*Cyberspace. A consensual hallucination experienced daily by billions of legitimate operators, in every nation, by children being taught mathematical concepts... A graphic representation of data abstracted from banks of every computer in the human system. Unthinkable complexity. Lines of light ranged in the nonspace of the mind, clusters and constellations of data. Like city lights, receding...*

William Gibson, Neuromancer

# 6

# An information-theoretic model for measuring the anonymity risk in time-variant user profiles

Websites and applications use personalisation services to profile their users, collect their patterns and activities and eventually use this data to provide tailored suggestions. User preferences and social interactions are therefore aggregated and analysed. Every time a user publishes a new post or creates a link with another entity, either another user, or some online resource, new information is added to the user profile. Exposing private data does not only reveal information about single users' preferences, increasing their privacy risk, but can expose more about their network that single actors intended. This mechanism is self-evident in

*social networks* where users receive suggestions based on their friends' activities.

This chapter is centred on an information-theoretic approach to measure the differential update of the anonymity risk of time-varying user profiles, continuing on previous work published in [**?** ]. We are interested to measure how privacy is affected when new content is posted and how much third-party services *get to know* about the users when a new activity is shared. We use actual Facebook data to show how our model can be applied to a real-world scenario.

## 6.1 Background

Personalisation and advertising services collect user's activities to provide tailored suggestions. This data contributes to form over time what is considered the user online footprint. With the term online footprint we include every possible trace left by individuals when using communication services. It follows that the same notion of digital footprint spans all layers of the TCP/IP model, depending on the type of data taken into considerations. It is also important to note that the digital footprint of an individual is formed by their interaction with their social relationships, not only by their singular actions on a medium or platform.

We can therefore consider users' online footprints as linked data, where each event generated by a single user includes information regarding other users but also regarding other events and entities. This way of considering online footprints is very similar to the very structure of the Web, where web pages link to other pages when they reference a certain individual or object. This social and interconnected aspect of digital footprints is particularly evident for services like Facebook [**?** ], where users are suggested new pages and social connections based on their friends' network of relationships and expressed preferences, or *likes*.

Users' profiles also change over time, reflecting how real-world individuals change their tastes and preferences in comparison to, for example, a reference population. Every time new information is shared, the user is disclosing more about themselves or their social interactions, eventually changing their privacy risk.

More importantly, users tend to share their data and access to their identity ac-

counts, such as Google [ **?** ] or Facebook [ **?** ], when interacting with third-party applications. These applications use federated log in mechanisms through the user's identity account. To use the application, users grant it a certain level of access to their private data through their profile. This data includes details about their real *offline* identity, their whereabouts and in some situations even the company they work for. Once it has gained access, the application can now store user data and assume control over how it is further shared. The user will never be notified again about who is accessing their data, nor if these are transferred to third parties.

This aspect of privacy protection is particularly relevant since the right to privacy is commonly interpreted as the user's right to prevent information disclosure. When a user shares some content online, they are actively choosing to disclose some of their profile. At the same time, though, they might give away more that they intended, since no information is shared from app and service about how the profile is analysed or how the user's data is further shared.

Online services ask the user to access certain information, yet no concrete information is passed on how the data will be used or stored. Furthermore, these services are often designed as mobile applications where all the devices installing the app communicate with a centralised server and constantly exchange users' information, eventually allowing for unknown third parties, or potential attackers, to fetch and store this data. In addition, this information is often shared with insecure communication through the HTTP protocol, making it possible for a malicious entity to intercept these communications and steal user data.

In this model the management of privacy and trust of the platform to which users handle their data is highly centralised. The user entrusts the service with all their data, often as part of a service agreement. Generally a few services control the market and therefore can inevitably *know more* about the users. This is the case of popular email or messaging services, but also social networks, relationship apps and so on. These entities can easily know who is talking to whom and sometimes also the topic of their conversations.

In this chapter we analyse user online footprints as a series of events belonging to a certain individual. Each event is a document containing different pieces of in-

formation. An event correspond to an action generated by the user or one of their devices. When a user visits a website or creates a post on a blog, an event is created. We can think of an event as a hypermedia document, i.e., an object possibly containing graphics, audio, video, plain text, and hyperlinks. We call the hyperlinks selectors, and we use them to build the connections between the user's different identities or events. Each identity can be a profile or account that the user has created onto a service or platform, or just a collection of events, revealing something about the user. With account we mean an application account or a social network account, such as their LinkedIn or Facebook unique IDs.

When the user decides to share some new content, or subscribes a service by sharing part of their profile data, novel information is released. This information is either made public or shared to a group of people, like for a new social network post, or it is rather shared to a third party app.

We are interested to measure the differential update of the anonymity risk of user profiles due to a marginal release of novel information, based on an information-theoretic measure of anonymity risk, precisely, the Kullback-Leibler divergence between a user profile and the average population's profile.

We particularly considered real data shared by Facebook users as part of the Facebook-Tracking-Exposed project [**?** ]. For the purpose of this study, we considered categorised Facebook posts. We imagined that an attacker is interested in capturing users' preferences by looking at their posts and imagined a scenario where the information shared through a new event (i.e. sharing new content) increases or decreases the user's privacy risk, in other words, how much an attacker knows about them, once they have captured the new information.

In this work, we build upon a recent information-theoretic model for measuring the privacy risk incurred in the disclosure of a user's interests though online activity. Among other refinements, we incorporate an aspect of substantial practical importance in the aforementioned model, namely, the aspect of time-varying user profiles.

More precisely, we propose a series of refinements of a recent information-theoretic model characterising a user profile by means of a histogram of categories of in-

terest, and measuring the corresponding privacy risk as the Kullback-Leibler divergence with respect to the histogram accounting for the interests of the overall population. Loosely speaking, this risk may be interpreted as an anonymity risk, in the sense that the interests of a specific user may diverge from those of the general population. Our main contributions are as follows.

- We preface our main analysis with an argument to tackle populations in which the distribution of profiles of interest is multimodal, that is, user profiles concentrate around distinguishable clusters of archetypical interests. We suggest that said information-theoretic model be applied after segmentation of the overall population according to demographic factors, effectively extending the feasibility of the original, unimodal proposal.

- But the most important refinement and undoubtedly the main focus of this chapter consists in the extension of the aforementioned model to time-varying user profiles. Despite the practical significance of the aspect of time in the analysis of privacy risks derived from disclosed online activity, it is nevertheless an aspect all too often neglected, which we strive to remedy with this preliminary proposal. Here, the time variation addresses not only changes over time in the interests of a user, construed as a dynamic profile, but also novel activity of a possibly static profile, in practice known only in part.

- The changes in anonymity risk are formulated as a gradient of the Kullback-Leibler divergence of a user profile reflecting newly observed activity, with respect to a past history, and are inspired in the abstract formulation of Bregman projections onto convex sets, whose application to the field of privacy is, to the best of our knowledge, entirely novel.

- For a given activity and history, we investigate the profile updates leading to the best and worst overall anonymity risk, and connect the best case to the fairly recent information-theoretic framework of optimised query forgery and tag suppression for privacy protection.

- We contemplate certain special cases of interest. On the one hand, we provide a corollary of our analysis for the special case in which the anonymity risk is measured as the Shannon entropy of the user profile. On the other hand, we particularise our model in the extreme case in which the new observation consists in a single sample of categorised online activity.

- Last but not least, we verify and illustrate our model with a series of examples and experiments with both synthetic and real online activity.

## 6.2 An information-theoretic model for measuring anonymity risk

In this section, we build upon a recent information-theoretic model for measuring the privacy risk incurred in the disclosure of a user's interests though online activity. Among other refinements, we incorporate an aspect of substantial practical importance in the aforementioned model, namely, the aspect of time-varying user profiles.

Consider a user profile $p$, together with an average population profile $q$, both represented as histograms of relative frequencies of online activity along predefined categories of interest $i = 1, \ldots, m$. In the absence of a specific statistical model on the frequency distribution of user profiles, as argued extensively in [**? ? ?** ] on the basis of Jaynes' rationale for maximum entropy methods, we assume that *anonymity risk* may be adequately measured as the *Kullback-Leibler* (KL) *divergence* $\mathrm{D}(p\|q)$ between the user profile $p$ and the population's $q$. The idea is that user profiles become less common as they diverge from the average of the population. Precisely, we define anonymity risk as

$$\mathcal{R} \stackrel{\text{def}}{=} \mathrm{D}(p\|q) \stackrel{\text{def}}{=} \sum_{i=1}^{m} p_i \log \frac{p_i}{q_i}.$$

Usually, the basis of logarithm is 2 and the units of the divergence are bits.

Intuitively, the empirical histogram of relative frequencies (or type) $t$ of $n$ inde-

pendent, identically distributed drawings should approach the true distribution $\bar{t}$ as $n$ increases. Those drawings may be loosely interpreted as sequences of online queries according to some underlying user interests represented by $\bar{t}$. More technically, the extension of Jaynes' approximation to KL divergences for a sequence of independent events shows that the probability $p_T(t)$ of the empirical distribution $t$ is related to the KL divergence $\mathrm{D}(t\|\bar{t})$ with respect to the true distribution $\bar{t}$ by means of the limit

$$-\tfrac{1}{n}\log p_T(t) \xrightarrow[n\to\infty]{} \mathrm{D}(t\|\bar{t}).$$

According to this model, the user profile $p$ plays the role of the empirical distribution $t$, and the population's profile $q$, the role of the true distribution $\bar{t}$. In a way, we construe a user profile as an empirical instantiation of the population's profile. Concordantly, the divergence $\mathrm{D}(p\|q)$ between the user profile $p$ and the population's $q$ is a measure of how rare $p$ should be, which we regard in turn as a measure of *anonymity risk*. The argument that the rarity of a profile may also be understood as a measure of how sensitive a user profile may be considered, offers a measure of *privacy risk*. Admittedly, this model is limited to applications where the underlying assumptions may be deemed adequate, particularly when no specific, possibly multimodal distribution of the user profiles is available.

Another helpful interpretation of this measure stems from rewriting the user profile as a distribution $p_{I|J}$ of a random variable $I$ indexing online activity into predefined categories $i = 1, \ldots, m$, conditioned on the user identity $J$, defined on the user indexes $j = 1, \ldots, n$. Observing that the population profile is the expectation across all user profiles,

$$q_I = \mathrm{E}_J\, p_{I|J}(\cdot|J), \quad \text{(more explicitly,} \quad q_I(i) = \tfrac{1}{n}\sum_{j=1}^{n} p_{I|J}(i|j) \quad \text{for all } i\text{)},$$

we immediately conclude that the expected risk is

$$\mathrm{E}_J\, \mathcal{R}(J) = \mathrm{E}_J\, \mathrm{D}\left(p_{I|J}(\cdot|J) \,\middle\|\, q_I\right) = \mathrm{I}(I;J),$$

namely, the mutual information between the online activity $I$ and the user identity $J$.

### 6.2.1 Multimodality of the KL divergence model and conditioning on demography

Perhaps one of the major limitations of the direct application of the KL divergence model for characterising the anonymity of a profile is made clear when the distribution of profiles is concentrated around several predominant modes, contradicting the implicit unimodal assumption revolving around the population's profile $q$. Intuitively, one may expect several clusters in which profiles are concentrated, corresponding to various demographic groups, characterised by sex, age, cultural background, etc.

In order to work around this apparent limitation, we may simply partition the data into a number of meaningful demographic groups, indexed by $k$, and calculate the average population profile $q_{I|K}(\cdot|k)$ for each group $k$. Then, redefine the demographically contextualised anonymity risk as the KL divergence between the profile $p_{I|J}(\cdot|j)$ of user $j$, in group $k(j)$, and the corresponding reference $q_{I|K}(\cdot|k(j))$, that is,

$$\mathcal{R}_{\text{context}}(j) \stackrel{\text{def}}{=} \mathrm{D}\left(p_{I|J}(\cdot|j) \,\middle\|\, q_{I|K}(\cdot|k(j))\right).$$

Obviously, the model will be suitable as long as the profile distribution is unimodal within each demographic context, in the absence of a more specific model. Note that the measure of anonymity risk of the disclosed interests is now conditioned on demographic data potentially observable by a privacy attacker.

### 6.2.2 Gradient of the KL divergence and information projection

Before addressing the problem of the differential update per se, we quickly review an interesting result on the gradient of the KL divergence, and its application to convex projections with said divergence. Directly from the definition of the KL divergence between distributions $p$ and $q$ for a general logarithmic basis, compute

the gradient on the first argument

$$\nabla_p \mathrm{D}(p\|q) = \left(\log \frac{p_i}{q_i} + \log e\right)_i.$$

Swift algebraic manipulation shows that

$$\mathrm{D}(p\|q) = \mathrm{D}(p\|p^*) + \mathrm{D}(p^*\|q) + \nabla_{p^*}\mathrm{D}(p^*\|q)^{\mathrm{T}}(p - p^*), \tag{6.1}$$

for any additional distribution $p^*$, where the constant term $\log e$ in the gradient becomes superfluous, on account of the fact that $\sum_i p_i - p_i^* = 0$. Observe that part of the above expression may be readily interpreted as the Taylor expansion of $\mathrm{D}(p\|q)$ about $p^*$,

$$\mathrm{D}(p\|q) = \mathrm{D}(p^*\|q) + \nabla_{p^*}\mathrm{D}(p^*\|q)^{\mathrm{T}}(p - p^*) + O(\|p - p^*\|^2), \tag{6.2}$$

with error precisely $\mathrm{D}(p\|p^*)$.

In the context of convex projections, suppose that we wish to find the closest point $p^*$ inside a convex set $\mathscr{P}$ to a reference point $q$, in KL divergence, succinctly,

$$p^* = \operatorname*{arg\,min}_{p \in \mathscr{P}} \mathrm{D}(p\|q).$$

This problem is represented in Fig. 6.2.1. The solution $p^*$ is called the *information projection* of $q$ onto $\mathscr{P}$. Because for such $p^*$ the projection of the gradient of the objective onto the vector difference $p - p^*$ for any $p \in \mathscr{P}$ must be nonnegative, i.e.,

$$\nabla_{p^*}\mathrm{D}(p^*\|q)^{\mathrm{T}}(p - p^*) \geqslant 0,$$

we may conclude from the previous equality involving the gradient that

$$\mathrm{D}(p\|q) \geqslant \mathrm{D}(p\|p^*) + \mathrm{D}(p^*\|q).$$

This last inequality is, in fact, a known generalisation of the Pythagorean theorem

for projections onto convex sets, generally involving obtuse triangles[1].

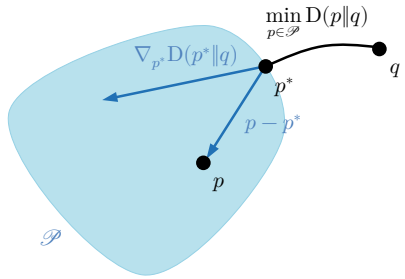



**Figure 6.2.1:** Information projection $p^*$ of a reference distribution $q$ onto a convex set $\mathscr{P}$.

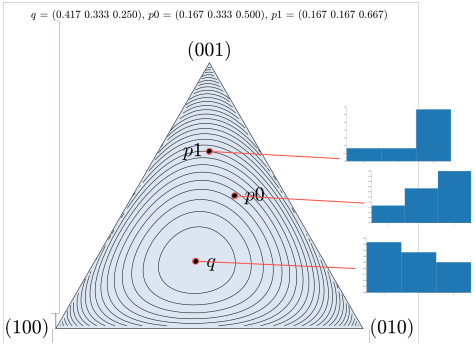



**Figure 6.2.2:** Probability simplices showing, the population distribution $q$, the user's profile $p_0$, the updated profile $p_1$.

[1]The expression relating the gradient with a set of divergences shown here may be readily generalise to prove an analogue of the Pythagorean theorem for Bregman projections. Recall that Bregman divergences encompass both squared Euclidean distances and KL divergences as a special case. An alternative proof of the Pythagorean theorem for KL divergences, which inspired a small part of the analysis in this manuscript, can be found in [**?** ] (Theor. 11.6.1).

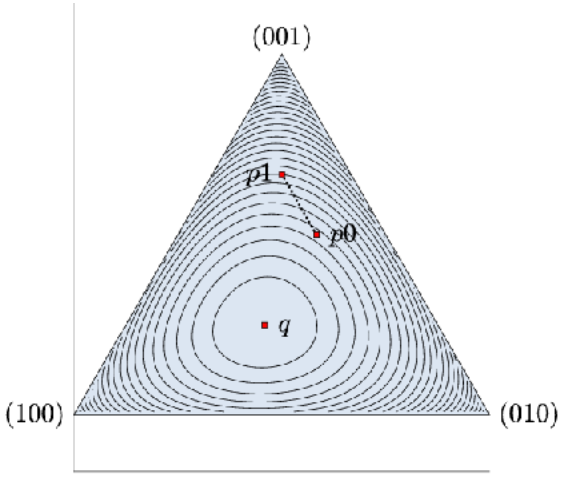



**Figure 6.2.3:** Probability simplices showing, the population distribution $q = (0.417, 0.333, 0.250)$, the user's profile $p_0 = (0.167, 0.333, 0.500)$, the updated profile $p_1 = (0.167, 0.167, 0.666)$. The intermediate points show the value of $p_a$ for different $a$.

### 6.2.3 Differential update of the anonymity risk due to revealing new information

Under this simple model, we consider the following problem. Suppose that the distribution $p_0$ represents a history of online activity of a given user up to this time, with associated anonymity risk $D(p_0\|q)$. Consider now a series of new queries, with interests matching a profile $p_1$ and associated risk $D(p_1\|q)$ (Fig. 6.2.2). If those new queries were observed, the overall user profile would be updated to

$$p_a = (1-a)p_0 + ap_1,$$

where the activity parameter $a \in (0,1)$ is the fraction of new queries with respect to the total amount of queries released. We investigate the updated anonymity risk (Fig. 6.2.3)

$$D((1-a)p_0 + ap_1\|q),$$

in terms of the risks associated with the past and current activity, for a marginal activity increment $a$. To this end, we analyse the first argument of the KL divergence, in the form of a convex combination, through a series of quick preliminary lemmas[2].

On the one hand, since the KL divergence is a convex function, we may bound the updated risk as

$$\mathrm{D}\left((1-a)p_0 + ap_1 \| q\right) \leqslant (1-a)\mathrm{D}(p_0\|q) + a\,\mathrm{D}(p_1\|q). \tag{6.3}$$

On the other hand, we may resort to our previous gradient analysis in §6.2.2, specifically to (6.1) and (6.2), to write the first-order Taylor approximation

$$\mathrm{D}\left((1-a)p_0 + ap_1 \| q\right) = (1-a)\mathrm{D}(p_0\|q) + a\,\mathrm{D}(p_1\|q) - a\,\mathrm{D}(p_1\|p_0) + O(a^2). \tag{6.4}$$

This last expression is consistent with the convexity bound (6.3), and quite intuitively, the term $-a\,\mathrm{D}(p_1\|p_0)$ in the Taylor approximation refining the convex bound vanishes for negligible activity $a$ or new activity profile $p_1$ similar to the history $p_0$ revealed thus far. We may alternatively write the updated risk as an increment with respect to that based on the user's online history, as

$$\mathrm{D}\left((1-a)p_0 + ap_1 \| q\right) - \mathrm{D}(p_0\|q) = a\left(\mathrm{D}(p_1\|q) - \mathrm{D}(p_0\|q) - \mathrm{D}(p_1\|p_0)\right) + O(a^2),$$

which we observe to be approximately proportional to the relative activity parameter $a$, and to an expression that only depends on the divergences between the profiles involved.

[2]The mathematical proofs and results developed here may be generalised in their entirety from KL divergences to Bregman divergences, and they are loosely inspired by a fundamental Pythagorean inequality for Bregman projections on convex sets.

### 6.2.4 Special cases of delta update and uniform reference

In the special case when the new activity contains a single query, the new profile $p_1$ is a Kronecker delta $\delta^i$ at some category $i$. In this case,

$$\mathrm{D}(p_1 \| q) = \mathrm{D}(\delta^i \| q) = -\log q_i, \text{ and}$$

$$\mathrm{D}\left((1-a)p_0 + a p_1 \| q\right) = (1-a)\mathrm{D}(p_0 \| q) + a \log \frac{p_{0\,i}}{q_i} + O(a^2).$$

A second corollary follows from taking the reference profile $q$ as the uniform distribution $u = \frac{1}{m}$, and replacing KL divergences in (6.3) and (6.4) with Shannon entropies according to

$$\mathrm{D}(p \| u) = \log m - \mathrm{H}(p). \tag{6.5}$$

Precisely,

$$\mathrm{H}\left((1-a)p_0 + a p_1\right) \geqslant (1-a)\mathrm{H}(p_0) + a\,\mathrm{H}(p_1). \tag{6.6}$$

consistently with the concavity of the entropy, and

$$\mathrm{H}\left((1-a)p_0 + a p_1\right) = (1-a)\mathrm{H}(p_0) + a\,\mathrm{H}(p_1) + a\,\mathrm{D}(p_1 \| p_0) + O(a^2). \tag{6.7}$$

Even more specifically, in the case of a delta update $p_1 = \delta^i$ and uniform reference profile,

$$\mathrm{H}\left((1-a)p_0 + a p_1\right) = (1-a)\mathrm{H}(p_0) - a \log p_{0\,i} + O(a^2).$$

### 6.2.5 Best and worst update

For a given activity $a$ and history $p_0$, we investigate the profile updates $p_1$ leading to the best and worst overall anonymity risk $\mathrm{D}\left((1-a)p_0 + a p_1 \| q\right)$. The problem of finding the best profile, yielding the smallest risk, is formally identical to that of optimal query forgery extensively analysed in [**?** ]. Note that this problem may also be interpreted as the information projection of the population profile $q$ onto

the convex set of possible forged profiles

$$\mathscr{P} = \{(1-a)p_0 + ap_1\},$$

with fixed $a$ and $p_0$, a scaled, translated probability simplex. In this case, the generalized Pythagorean theorem shown earlier guarantees

$$\mathrm{D}\left((1-a)p_0 + ap_1 \| q\right) \geqslant \mathrm{D}\left((1-a)p_0 + ap_1^* \| (1-a)p_0 + ap_1\right) + \mathrm{D}\left((1-a)p_0 + ap_1^* \| q\right).$$

We may now turn to the case of the worst profile update $p_1$, leading to the highest anonymity risk. Consider two distributions $p$ and $q$ on the discrete support alphabet $i = 1, \ldots, m$, representing predefined categories of interest in our context. Recall that $p$ is said to be *absolutely continuous* with respect to $q$, denoted $p \ll q$, whenever $q_i = 0$ implies $p_i = 0$ for each $i$. Otherwise, if for some $i$, we had $p_i > 0$ but $q_i = 0$, then $\mathrm{D}(p\|q) = \infty$. In the context at hand, we may assume that the population profile incorporates all categories of interest, so that $q_i > 0$, which ensures absolute continuity, i.e., $p \ll q$. Therefore, we would like to solve

$$\max_{p_1 \ll q} \mathrm{D}\left((1-a)p_0 + ap_1 \| q\right).$$

We shall distinguish two special cases, and leave the general maximisation problem for future investigation. Let us tackle first the simpler case $a = 1$, and call $p_1 = p$. Recall that the *cross-entropy* between two distributions $p$ and $q$ is defined as

$$\mathrm{H}(p\|q) = -\sum_{i=1}^{m} p_i \log q_i,$$

and is related to the (Shannon) entropy and the KL divergence via

$$\mathrm{H}(p\|q) = \mathrm{H}(p) + \mathrm{D}(p\|q).$$

Clearly,

$$\max_{p \ll q} \mathrm{H}(p\|q) = -\log q_{\min},$$

attained for $p = \delta^i$ corresponding to the category $i$ minimising $q$. It turns out that this is also the solution to the maximisation problem in the divergence, because

$$\mathrm{D}(p\|q) = \mathrm{H}(p\|q) - \mathrm{H}(p),$$

and $\mathrm{H}(\delta^i) = 0$, which means that $p = \delta^i$ simultaneously maximises the cross-entropy and minimises the entropy.

The second special case we aim to solve is that of a uniform reference $q = u$, discussed in §6.2.4. The corresponding problem is

$$\min_{p_1} \mathrm{H}\left((1-a)p_0 + ap_1\right).$$

We claim that the worst profile update $p_1$ is again a Kronecker delta, but this time at the category $i$ maximising $p_0$. Indeed, assume without loss of generality that $p_0$ is sorted in decreasing order, observe that $(1-a)p_0 + a\delta^1$ majorises any other convex combination $(1-a)p_0 + ap_1$, and recall that the entropy is Schur-concave.

As for the general case, the associated cross-entropy problem is fairly simple. We have

$$\max_{p_1 \ll q} \mathrm{H}\left((1-a)p_0 + ap_1\|q\right) = (1-a)\mathrm{H}(p_0\|q) - a\log q_{\min}, \tag{6.8}$$

for $p = \delta^i$ at the category minimising $q$. Unfortunately, the terms in the difference

$$\mathrm{D}\left((1-a)p_0 + ap_1\|q\right) = \mathrm{H}\left((1-a)p_0 + ap_1\|q\right) - \mathrm{H}\left((1-a)p_0 + ap_1\right),$$

are respectively maximised and minimised for deltas at different categories, in general, namely that minimising $q$, and that maximising $p_0$. We may however provide an upper bound on the anonymity risk based on these considerations; by virtue of the convexity of the divergence and the previous result on its maximisation,

$$\mathrm{D}\left((1-a)p_0 + ap_1\|q\right) \leqslant (1-a)\mathrm{D}(p_0\|q) - a\log q_{\min}. \tag{6.9}$$

## 6.3 Experimental results

In the previous section, we formulated the theoretical problem of the differential update of the anonymity risk of time-varying user profiles due to a marginal release of novel information, based on an information-theoretic measure of anonymity risk, specifically, the Kullback-Leibler (KL) divergence between a user profile and the average population's profile. In this section, we verify the theoretical conclusions drawn in the referred section with a series of numerical examples and experimental scenarios.

More precisely, we analyse the updated anonymity risk in terms of the profile's history and the current activity, for a given marginal increment $a$. Furthermore, we present how, fixed an activity parameter $a$, and given a certain initial profile, it is possible to identify the best and worst profile update leading to a new privacy risk. All of this is shown for the general case of anonymity risk measured as the KL divergence between a user profile and the overall profile of a population, and for the special case in which the population's profile is assumed uniform, in which divergences become Shannon entropies.

The examples simply resort to synthetic values of the reference profiles. As for the experimental scenario, we employ Facebook data. We consider a user sharing some new information through a series of posts on their timeline. We are interested to verify the theoretical analysis carried out in this work. All divergences and entropies are in bits.

### 6.3.1 Synthetic examples

In our first proposed example, we choose an initial profile $p_0 = (1/6, 1/3, 1/2)$, representing a user's past online history, an updated profile $p_1 = (1/6, 1/6, 2/3)$ containing more recent activity, and a population distribution $q = (5/12, 1/3, 1/4)$ of reference, across three hypothetical categories of interest. For different values of the recent activity parameter $a$, Fig. 6.3.1a plots the anonymity risk $\mathrm{D}(p_a \| q)$ of our synthetic example of updated user profile $p_a = (1-a)p_0 + ap_1$, with respect to the population's profile $q$, the user's history $p_0$, and the recent activity $p_1$. Specifically,

we verify the convexity bound (6.3) and the first-order Taylor approximation (6.4) in our theoretical analysis. In addition, we plot (b) the special case of uniform population profile, in which the anonymity risk becomes $\mathrm{H}(p_a)$. We should hasten to point out that the dually additive relationship (6.5) between KL divergence and entropy translates to vertically reflected versions of analogous plots, verifying the entropic properties (6.6) and (6.7).

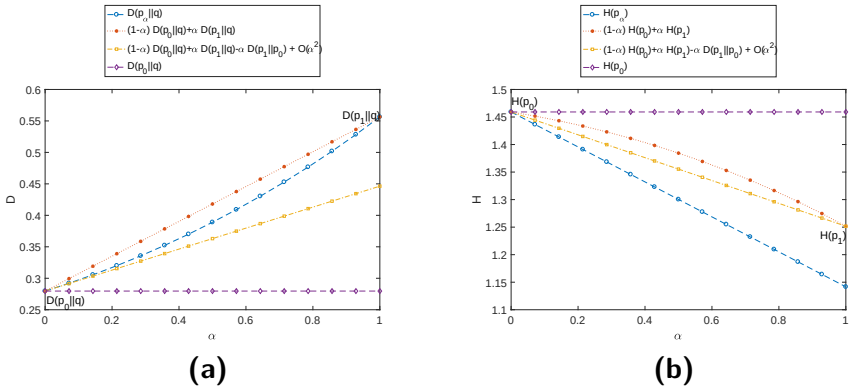
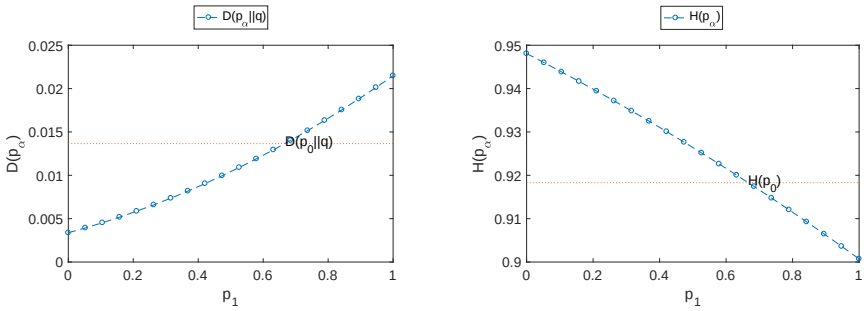



**Figure 6.3.1:** For different values of the recent activity parameter $a$, we plot (a) the anonymity risk $\mathrm{D}(p_a\|q)$ of a synthetic example of updated user profile $p_a = (1-a)p_0 + ap_1$, with respect to the population's profile $q = (5/12, 1/3, 1/4)$, across three hypothetical categories of interest, where $p_0 = (1/6, 1/3, 1/2)$ represents the user's online history, and $p_1 = (1/6, 1/6, 2/3)$ contains the recent activity in the form of a histogram. We verify the convexity bound (6.3) and the first-order Taylor approximation (6.4) in our theoretical analysis. In addition, we plot (b) the special case of uniform population profile, in which the anonymity risk becomes $\mathrm{H}(p_a)$.

In our second example we consider two categories of interest, so that profiles actually represent a binary preference. In this simple setting, profiles are completely determined by a single scalar $p$, corresponding to the relative frequency of one of the two categories, being $1-p$ the other frequency. We fix the activity parameter $a = 1/20$, set the historical profile to $p_0 = 2/3$, the reference profile to $q = 3/5$, and verify the analysis on the worst anonymity risk update of §6.2.5 plotting $D(p_a\|q)$ against profile updates $p_1$ ranging from 0 to 1, where, as usual,

$p_a = (1-a)p_0 + ap_1$. We illustrate this both for the privacy risk based on the KL divergence, in Fig. 6.3.2a, and for the special case of Shannon entropy, in Fig. 6.3.2b.




**(a)** **(b)**

**Figure 6.3.2:** In this example we consider two categories of interest, therefore profiles are completely determined by a single scalar $p$, being $1-p$ the other frequency. We fix the activity parameter $a = 1/20$, set the historical profile to $p_0 = 2/3$, the reference profile to $q = 3/5$, and verify the analysis on the worst anonymity risk update of §6.2.5 plotting $D(p_a\|q)$ against profile updates $p_1$ ranging from 0 to 1. In the entropy case we plot $H(p_a)$.

In the entropy case, our analysis, summarised in the minimisation problem (6.8), concluded that the worst update is a delta in the most frequent category. In this simple example with two categories, since $p_0 > 1/2$, the worst update corresponds to $p_1 = 1$, giving the lowest entropy. The reference line in the plot corresponds to $H(p_0) \approx 0.918$ bit. For the more general measure of risk as a divergence, since $q = 3/5$, we have $q_{\min} = 2/5$, and the bound (6.9) becomes

$$D(p_a\|q) \leqslant (1-a)D(p_0\|q) - a\log_2 q_{\min} \approx 0.0791,$$

fairly loose for the particular values of this example. The reference line in the plot indicates $D(p_0\|q) \approx 0.0137$.

These two examples confirm that new activity certainly has an impact on the overall anonymity risk, in accordance with the quantitative analysis in §6.2.5. This

can of course be regarded from the perspective of introducing dummy queries in order to alter the apparent profile of interests, for example, in line with the problem of optimized query forgering investigated in [**?** ].

### 6.3.2 Experiment based on Facebook data

We continue our verification of the theory presented, this time with experiments based on Facebook data, that is, a realistic scenario for which a population of users is sharing posts on Facebook. For the purpose of this study we have used data extracted from the Facebook-Tracking-Exposed project [**?** ], where users contribute their data to gain more insights on Facebook personalisation algorithm.

The extracted dataset contained 59 188 posts of 4 975 timelines, categorised over 10 categories of interest. We selected two users out of this dataset and considered the total of posts collected for each of them, i.e., their entire timelines. The population distribution for the users in the dataset is expressed by the following PMF:

$$q = (0.0401, 0.0870, 0.1485, 0.1691, 0.1025, 0.2081, 0.0435, 0.0525, 0.0558, 0.0924).$$

Note that $q$ is computed by taking into account not only the selected users, but the entire population of users across the dataset.

For each user we considered a historical profile comprising of the entirety of their posts minus a window of 15 posts. Over this window we consider a smaller sliding window for computing $p_1$, of 5 posts, hence we set the activity parameter $a = w/L$, where $L = len(timeline)$ is the total number of posts in the timeline, and $w$ represents the sliding window of 5 posts (Fig. 6.3.3). For *User A* $a_A = 0.0182$, while for *User B* $a_B = 0.0820$. This choice captures the idea that we want to simulate how the profile changes when the user shares $n$ new posts.

For User A we consider a series 376 shared posts, and for User B we consider a

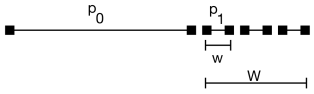



**Figure 6.3.3:** The image represents how the user initial profile was computed starting from the timeline data included in the dataset. Furthermore we show how the window $W$ of 15 posts is chosen from the last post of the series and how we considered a sliding window $w$ of 5 posts each time.

total of 61 posts. We can express the two users' profiles with the following PMFs:

$$p(A)_0 = (0.0146, 0.0036, 0.0810, 0.2311, 0.0397, 0.1931, 0.0156, 0.0324, 0.3705, 0.0179),$$

$$p(B)_0 = (0.0159, 0.0090, 0.0804, 0.2280, 0.0609, 0.1991, 0.0194, 0.0749, 0.2846, 0.0274).$$

For the set value of activity parameter $a$, Figs. 6.3.4a, 6.3.4c plot the anonymity risk $D(p_a\|q)$ between a user's updated profile $p_a = (1-a)p_0 + ap_1$, with respect to the population distribution $q$. Recall that $p_0$ is a user's profile in the Facebook dataset, built taking into consideration a long series of samples. This capture the idea that a user's profile is computed out of their history over a long series of actions.

These experiments confirm the theoretical analysis and examples presented, verifying in a real-world settings the convexity bound (6.3) and the first-order Taylor approximation (6.4) described in our theoretical analysis. In addition, we can computer the bound (6.9) for the general measure of the privacy risk as the KL divergence, which becomes, for User A,

$$D(p_a\|q) \leqslant (1-a)D(p_0\|q) - a\log_2 q_{\min} \approx 0.8870,$$

and for User B,

$$D(p_a\|q) \leqslant 0.7723.$$

Furthermore, we considered, in Figs. 6.3.4b and 6.3.4d, the privacy risk increments between the user profiles and an updated profile given by a certain activity over

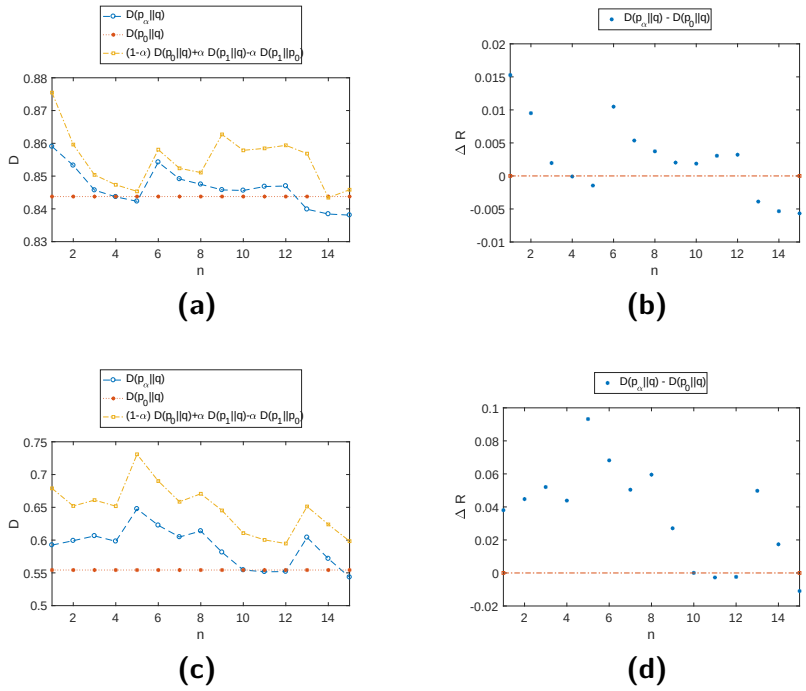



**Figure 6.3.4:** The figure considers the privacy risk between a user profile and a reference population distribution for two facebook users (Figs. 6.3.4b, 6.3.4d), and the risk increment $\Delta\mathcal{R} = \mathrm{D}(p_a\|q) - \mathrm{D}(p_o\|q)$ where $p_o$ is a user's profile in the Facebook dataset and $q$ is the reference population distribution calculated for all the posts in the dataset (Figs. 6.3.4b, 6.3.4d).

time. Recall that these deltas are computed as

$$\Delta\mathcal{R} = \mathrm{D}(p_a\|q) - \mathrm{D}(p_o\|q),$$

to show how a certain activity can theoretically result in an anonymity risk gain or loss.

Note that the theoretical analysis and results proposed in this article apply to dynamic profiles that change over time. This aspect is particularly interesting, since we are not simply considering profiles as a snapshot of the user's activity, over a small interval, but we are also taking into account changes in interests and general

behaviour that can impact the privacy risk.

As a result we can reach another interesting observation, consisting in the fact that profiles might have different privacy risk in different moments of time. This confirms the intuitive assumption that individuals might change their tastes and interests compared to a reference population, therefore having an impact on their overall privacy risk. In this case we reasonably assume that the profile of certain individuals might change more rapidly over time than that of the entire population.

## 6.4 Discussion

We proposed a series of refinements of a recent information-theoretic model of a user profile expressed through a histogram of categories of interest. The corresponding privacy risk is measured as the Kullback-Leibler divergence with respect to the histogram accounting for the interests of the overall population. Loosely speaking, this risk may be interpreted as an anonymity risk, in the sense that the interests of a specific user may diverge from those of the general population, extrapolating Jaynes' rationale on maximum-entropy methods.

We investigate the profile updates leading to the best and worst overall anonymity risk for a given activity and history. Thus, we connect the best case to the fairly recent information-theoretic framework of optimised query forgery and tag suppression for privacy protection.

Furthermore, the analysis of our model is applied to an experimental scenario, using Facebook timeline data. Our main objective was measuring how privacy is affected when new content is posted. Often, a user of some online service is unable to verify how much a possible privacy attacker can find out about them. We used real Facebook data to show how our model can be applied to a real world scenario. This aspect is particularly important for content filtering in Facebook. In fact, as users are profiled on Facebook, the very same activity is used to filter the information they are able to access, based on their interests. There is no transparency on Facebook's side about how this filtering and profiling happens. We hope that studies like this might encourage users to seek more transparency in the filtering

techniques used by online services in general.

With regard to future work, we would like to express the relationships between users as well as the people they communicate with, taking them all into consideration when calculating users' privacy risk.

*A new life awaits you in the Off-world colonies! A chance to begin again in a golden land of opportunity and adventure!*

Blade Runner

# 7

# Online identities as hypermedia documents

Web services use personal data to sustain their business by fuelling this stream of information to recommendation systems to generate tailored advertising. When users decide to subscribe a service, they automatically lose control over the data generated by their activity. This is especially true when third parties are authorised to access a user's profile and information. This chapter is focused on describing how user data is transferred from mobile and web application implementing their login flow through a larger authorisation provider, like Google or Facebook, and how this could be improved. Leveraging on work from [**?** ] on open web applications, and continuing on the study from [**?** ], we introduce a hypermedia model of the user online footprint. We show furthermore, how web and mobile app already

use hypermedia docuements to handle user data. Ultimately we use an example of a privacy broker to show how existing technologies can be used to allow users to retain control on their data. Our defined model can be both used to transfer data between services and clients and to express the value and risk of sharing certain information.

## 7.1 Background

The business model of many services and applications on today's web is based on having user authenticate and, completely or partially, generate data to *pay* for the possibility to use the service. Data is, in fact, used as a currency that can be resold for a variety of purposes. To target the users and suggest back product that they can buy, or to continuously track them across different visits, and in some cases different devices and websites.

Web services can furthermore dispose of the data collected as their property, as when users access a service they lose control and rights over their digital footprint. One of the problem associated with this is certainly linked to consent. Users give their implicit consent to the use of their data and subsequently their footprint, sometimes only in part, become property of the company providing the services. The same process of giving consent or granting permissions to access part of the user identity is not clear. This scenario is particularly evident when mobile applications request access to some resources on a user's phone, or when Facebook federated login mechanism is used. Facebook login provides both authentication and authorisation. The mechanism is used on the web as well as on iOS and Android, although on those platforms the primary mechanism uses the native Facebook application instead of the web API. When an application is connected to the user's Facebook profile using Facebook login, it can always access their *public profile* information. Facebook consider this information public and will not apply any restriction on it. Information that is shared with the public profile vary from user to user and depends on their privacy settings. By default, the Facebook public profile includes some basic attributes about the person such as the user's age range,

language and country, but also the name, gender, username and user ID (account number), profile picture, cover photo and networks. This is the minimum set of data disclosed by Facebook when the social network login mechanism is used to gain access to an app or service. This data is acquired by the app and user control over it is practically lost, as it will be disposed according to the app privacy policy. We propose an example scenario of a privacy broker implementing a privacy friendly authentication mechanism. We introduce two particular examples, the first one is based on the WebSocket protocol and shows a handshake based on the Socialist Millionaires protocol. The second example is based on the OAuth 2.0 flow which uses JWT to transmit private information to third parties. This examples builds upon previously published approaches and techniques, such as Crypto-Book [**?** ] and UnlimitID [**?** ], using the technique of blacklistable anonymous credentials [**?** ]. The idea behind these techniques is that a trusted (or semi trusted) Privacy Broker (PB) issues an anonymous credential $C(x)$ encoding a secret identity key x, unique and only known by the specific user. The user holding C(x) authenticates to a Service Provider (SP) by, given a cryptographically secure hash function and a secure elliptic curve group, producing a ticket $T$ depending on $x$, together with a zero-knowledge proof stating that they actually hold a valid credential from the PB, the key used to compute $T$ is the same as their secret identity key used to compute $C(x)$ and no previous ticket associated with past abuses used the identity key $x$. The examples proposed are presented to illustrate how a privacy broker could help the user maintain a desired footprint. Furthermore, the examples show how existing web protocols can be used to allow the user to authenticate anonymously to the SP and eventually disclose private information with the same SP or third-parties as they please. We would like to stress that we do not present novel cryptographic protocols, nor we propose modifications for existing web protocols or clients and that we make use of existing and widely adopted authorisation and authentication mechanism. Furthermore, the proposed hypermedia model allows the user to explore the shared preferences and properties shared with each service.

## 7.2 A hypermedia model of the user identity

Users interact with web services and applications with hypermedia protocols. Each time an action is completed onto an user's phone a call is performed to an APIs updating the user profile or sending some information to a service. These interactions are often completed over a REST protocol, such as HTTP, and consist of the client sending structured information to the server. This information can be anything regarding the user or the state of the used application, such as profile information or answers to specific queries initiated implicitly or explicitly by the user.

We define a model of the user identity to describe how a privacy policy can interact with data and which data or resource a service can access. Our model is based on the assumption that user data can be represented with a hypermedia object.

The whole concept of hypermedia is about making clients interact better with services, and making resources easily accessible and can be explored. A can be a record in a database, but it can also contain information from other records, or from other databases or applications. A resource is essentially an object acting as a gateway to some stream of information.

If we think for a moment about the evolution of the Internet and the Web, we are witnessing the transformation of a complex system from a platform of interconnected computers, to a platform of interconnected documents, to a platform of interconnected data streams.

The uniformity of web interfaces as defined by the RESTful architectural paradigms allows the usage of different types of identifiers to request resources in the same context, providing uniform semantics even when the access mechanism used may be different. As a matter of fact, we don't even have to be concerned with the access mechanism; we just need to ensure that our API replies consistently. The same principles permit us to introduce new types of resource identifiers without having to change the way existing identifiers work, while also allowing reuse of identifiers in a different context.

Building on the principles of RESTful resources, we begin by defining the identity model using JSONApi [**?** ] a specification for exchanging data between REST

interfaces. JSONApi can be used to define how a client should request that resources or their representations be fetched or modified, and how a server should respond to those requests. We envision that the same format can be used on the client side to represent identities and data associated with it and on the server side to request and exchange data.

```
{
    "data": {
        "type": "identity",
        "id": "john@johnsmith.com",
        "attributes":
            { // Attribute list },
        "relationships":
                { // Relationship list },
        "links": { // Third-party links }
    },
    "included": [ {
        "type": "resource",
        "id": "CC76598TDZKG9EEC3", // Device hardware id
        "attributes": { // resource attributes }
        "meta": { // Any meta information },
    }],
}
```

**Listing 7.1:** Some user identity data encoded with JSONApi.

We model the user's activity as series of events belonging to a certain identity. Each event is a document containing different information. We can formally define this an hypermedia document i.e. an object possibly containing graphics, audio, video, plain text and hyperlinks. We call the hyperlinks selectors and we use these to build the connections between the user's different identities or events. Each identity is a profile that the user has created onto a service or platform. This can be an application account or a social network account, such as their LinkedIn or Facebook unique IDs.

This is consistent with the versatility of RESTful interfaces. In fact, the interesting aspect of REST is that by combining relatively simple architectural elements it

is possible to build entire systems with complex functions.

In our hypermedia model, each event is the result of the user performing an action. For the purpose of this study we have consider an action as resulting using an application or a service. An action is the activity of interacting with a mobile application or *liking* a resource on a social network, i.e. directly expressing an interest, or the fact that a user has updated their location at a certain time.

We find that this model is able to express the user's online footprint as a collection of traces left across different services. Furthermore, by using a hypermedia approach we are able to grasp the connections between the different profiles and features. This results in the possibility to profile users based on chosen selectors. For example, we might want to trace all users who have been in the radius of 500 meters to a certain location, or all the users in a certain neighbourhood who *like* a selected Facebook page.

A service implementing the described model of the user identity can either be an identity provider or a client storing user's preferences and data. For example, a user might decide to login to third-party services through a trusted, or semi-trusted, identity provider, allowing them to disclose only a partial representation of their online footprint. The same user might store the full representation of their data locally on their devices or on different services.

The flexibility of this model allow the possibility to develop client applications that can retrieve different snippets of data from different identity providers and disclose information at the user control. We will know discuss how this can be achieved by showing how this can be implemented in well-used authentication protocols.

## 7.3 Data flow between identity providers and third-party services

At the time of writing, when third-party services use identity providers, like Facebook or Google, for federated login, a number of private data is transferred to the service at the time of login. If we observe a mobile app making a request to Face-

book to allow a user to login, we will see how the first call in the process is a OAuth request sending an authorization token as in Listing 7.2.

```
--3i2ndDfv2rTHiSisAbouNdArYfORhtTPEefj3q2f
Content-Disposition: form-data; name="batch_app_id"

464894686855067
--3i2ndDfv2rTHiSisAbouNdArYfORhtTPEefj3q2f
Content-Disposition: form-data; name="batch"

[{"relative_url":"oauth\/access_token?fields=&format=json&
    grant_type=fb_extend_sso_token&include_headers=false&sdk
    =ios","method":"GET"},{"relative_url":"me\/permissions?
    fields=&format=json&include_headers=false&sdk=ios","
    method":"GET"}]
```

**Listing 7.2:** OAuth request to Facebook.

If the request is successful, the user is granted permission to use the app and the service is transferred the user's data and authorised to refresh the user information unless this decides to prevent the app from doing so. This authorization is encoded with a *permission: property, status: value* JSON schema.

```
[{"code":200,"body":"{\"data\":[{\"permission\":\"
    user_birthday\",\"status\":\"granted\"},{\"permission\"
    :\"user_relationship_details\",\"status\":\"granted\"
    },{\"permission\":\"user_likes\",\"status\":\"granted\"
    },{\"permission\":\"user_education_history\",\"status\"
    :\"granted\"},{\"permission\":\"user_work_history\",\"
    status\":\"granted\"},{\"permission\":\"user_photos\",\"
    status\":\"granted\"},{\"permission\":\"user_friends\",\
    "status\":\"granted\"},{\"permission\":\"user_about_me\"
    ,\"status\":\"granted\"},{\"permission\":\"email\",\"
    status\":\"granted\"},{\"permission\":\"public_profile\"
    ,\"status\":\"granted\"}]}"}]
```

**Listing 7.3:** OAuth response from Facebook to third-party app.

At this point some services will use the information obtained to compute a user profile that reflects user generated data on their platform and their social identities. An example is a social app including the user's photo feed, as in Listing 7.4.

```
"user": {
    "_id": "531e1adefb3c64ad0500b2e5",
    "badges": [],
    "birth_date": "1985-04-27T08:36:13.695Z",
    "common_connections": [],
    "common_likes": [],
    "distance_mi": 2,
    "gender": 1,
    "group_matched": false,
    "instagram": {
        "completed_initial_fetch": true,
        "last_fetch_time": "2017-04-23T21:25:58.720Z",
        "media_count": 373,
        "photos": [
            {
                "image": "https://scontent.cdninstagram.com/
    t51.2885-15/34161997201408_n.jpg",
                "link": "https://www.instagram.com/p/BTPi20-
    AZEU/",
                "thumbnail": "https://scontent.cdninstagram.
    com/t51.2885-15/s150x150/e35/c18.jpg",
                "ts": "1492982755"
            },
        ...]
    }
}
```

**Listing 7.4:** Example of user profile including user's photos from other apps.

Even if the user deletes the app from their phone, the app will maintain the possibility to access their data on the identity provider, unless they will make the effort to delete it also from the list of authorised app. The process is not straightforward, and ultimately the app should request the user permission to access their data in-

stead of obtaining refreshed information from the identity provider.

## 7.4 Mitigation possibilities

We now discuss through two different examples, a mechanism that would be able to give users more control over their data and desired footprint, given the hypermedia model proposed above. We propose two possible solutions. We believe that the main problem with uncontrolled data flow between identity providers and third parties app resides in the way data is transferred when the user accesses the third party service. The first example uses WebSocket [**?** ] to perform a handshake protocol based on the Socialist Millionaires as implemented in Off The Record (OTR) messaging protocol [**?** ]. The second example uses a known non-interactive Zero-Knowledge proof and is based on the widely adopted Oauth 2.0 protocol [**?** ] for authorisation. We describe how JSON Web Token (JWT) [**?** ] Bearer can be used for anonymous authorisation and authentication.

### 7.4.1 Describing the OAuth 2.0 example

- Client
- Resource Owner
- Authorisation Server
- Resource Server

The six steps in the abstract OAuth 2.0 flow are defined as follows:

1. The client sends an authorisation request to the resource owner, either directly or via the authorisation server as intermediary.
2. The client receives an authorisation grant.
3. The client requests an access token by presenting the authorisation grant.

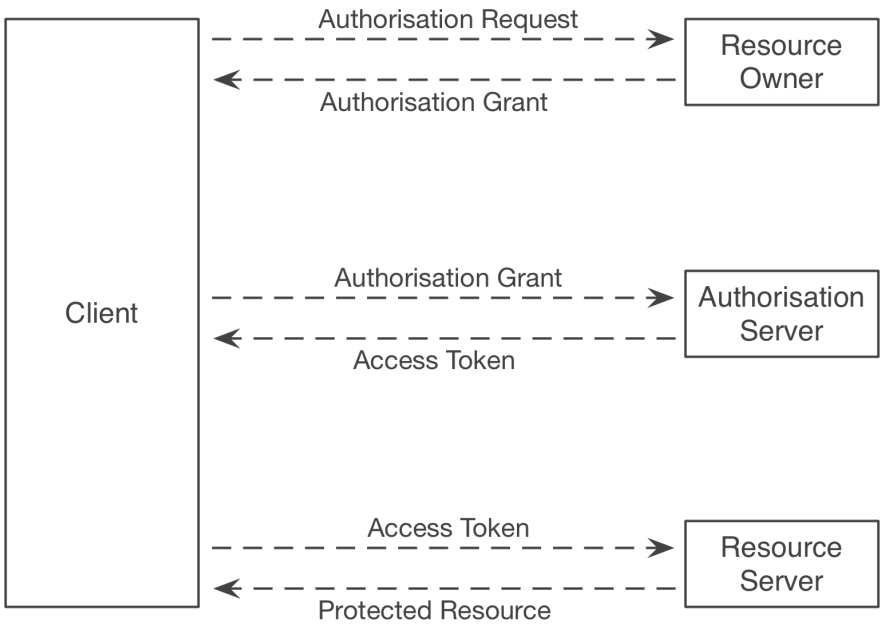



**Figure 7.4.1:** The abstract OAuth 2.0 flow describes the interaction between the four roles within the defined steps.

4. The client is authenticated by the authorisation server which issues and access token.

5. The client now requests the protected resource presenting the access token.

6. The resource server validates the token and replies with the resource representation if valid.

JSON Web Token (JWT) [**?** ] is a compact, URL-safe mechanism to represent and transfer claims between two parties. The claims are encoded as a JSON object. This can be used as the payload of a JSON Web Signature (JWS) [**?** ] structure, or as the plaintext in a JSON Web Encryption [**?** ] structure. Therefore, JWT allows for the claims to be digitally signed or integrity protected integrity protected with a Message Authentication Code (MAC) and/or encrypted.

### 7.4.2 Describing the WebSocket example

The WebSocket is a protocol created for web application to establish a bi-directional connection between a client running untrusted code and a remote host that has opted-in to communications from that code. WebSocket use the common origin-based security model implemented by web browsers, and consists in opening a handshake followed by basic messages framing, layered over TCP.

WebSocket provides a mechanism for browser-based applications to establish a two-way communication with a server without opening multiple HTTP connections based on a handshake protocol, where a WebSocket client or server uses a first a connection upgrade request. If the handshake is successful data can be transferred, by either the client or the server, at will.

Our WebSocket example implements the Socialist Millionaires protocol as used by OTR-chat after the WebSocket handshake and it is used to authenticate the client with the server without having to disclose the user identity.

### 7.4.3 Scenarios

We consider a scenario in which a user wants to login to a SP without having to directly expose their identity, by leveraging with a chosen privacy broker PB. Both the user and the SP trust the PB to an extent. In this scenario we have three parties:

- client
- service provider
- privacy broker

Before starting an authentication and subsequent authorisation process the client needs to create an account with the PB. This step can be consider as an *initialisation* phase.

### 7.4.4 Account creation on the PB

To create an account on the PB we assume that as the client is initialised it will generate a number of cryptographic keys to authenticate the various user identities. The keys can be saved in a wallet and used to register identities on the PB. A use might decide to use different keys for each service or the same key for a group of services. Note that each key doesn't correspond to a service account Id. A key is actually an account identification for the PB which is stored and used by the client.

To create an account on a service the client prepares a request for the PB performing a set of cryptographic operations that allows the client to register an anonymous identifier for the service. To implement such algorithm, we need a cryptographically secure hash function and a secure elliptic curve group. JSON Web Algorithms (JWA) [**?** ], provides Hash-based Message Authentication Codes (HMACs). HMACs enable the client to compute a MAC from a secret (the user's key) plus a cryptographic hash function. The algorithm for implementing and validating HMACs is provided in RFC 2104 [**?** ].

The account can be completely anonymous on the PB side as long as the SP is able to eventually blacklist the account in case they perform some malicious action.

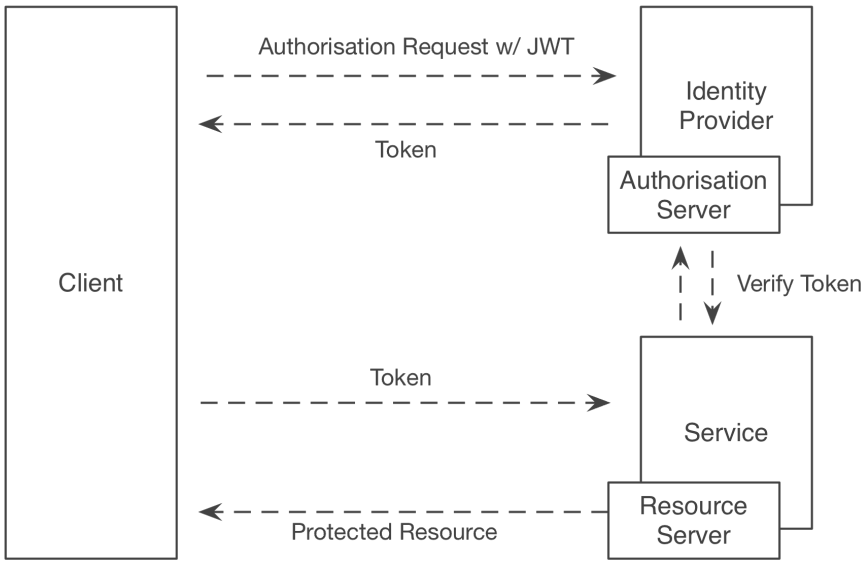



**Figure 7.4.2:** The privacy preserving OAuth 2 flow uses JWT to transmit user information.

### 7.4.5 Authorisation request on OAuth

A possible OAuth flow can be described as follows (Fig. 7.4.2):

1. The client sends an authorisation request to the PB. This request is signed and encrypted and packed into a JWT.
2. If the request is valid the PB issues a token.
3. The client uses the token to authenticate with the service. Which is verified separately with the PB.
4. The service returns access to the protected resource through its representation.

Once the client is issued a *token* from the PB it can authenticate with the *service*. Hence, the issued *token* which encodes the claims and service id can be transmitted

to the service encrypted with the JWE protocol. The token can then be verified and the client authorised to request the protected resource. An account is created on the service if the authentication request is initiated for the first time.

The OAuth example was implemented as follows:

1. The client sends an authorisation request to the PB.
2. The client receives an authorisation grant via a *JWT* encoding the o-knowledge proof to present to the SP.
3. The client engages the SP in a zero-knowledge proof that it holds a valid credential from the PB.
4. The client is authenticated (or not).
5. The client can now request protected resources, or the SP can request further information from the client.

The client can prove to know the secret pseudonym by solving a cryptographic challenge in the request over OAuth to the server and once it is authenticated through the PB can use the obtained OAuth token to communicate with the SP.

In our example we have considered the client sends the answer to a cryptographic challenge to the server, that can verify the proposed solution and issue a JWT with an authorisation ticket.

The client can use the ticket to disclose information with the PB or the SP. The PB can use the disclosed information to sing a statement to the SP regarding the client without disclosing the information or can authorise the client to access the SP by vouching for them.

### 7.4.6 Authorisation request on WebSocket

We imagine that a client has already established a handshake request with a server over WebSocket and is waiting to establish a communication. In this case the client will initiate an authorization process with the server where it wants to prove that

it knows the secret pseudonym registered with the server. To prove this, it can initiate a zero-knowledge proof like the Socialist Millionaires protocol to prove the server they know the pseudonym. We have used in out example a well-known protocol to show how this could be implemented in practice without having to modify existing web applications logic. Clearly, more efficient protocols and proofs can be used.

An interesting aspect of the WebSocket protocol is that it does provide the possibility to define custom protocols once the first handshake is success full. Here is where the OTR protocol, or another zero-knoledge proof protocol, can be used to preserve client anonymity.

Once the secure protocol is successful, the client can also initiate the data transfer to disclose information or receive an authorisation ticket that can be used with a third party.

The objective of this example is to show how it is possible to implement a secure protocol on top of existing infrastructure and how this wouldn't affect the mechanism performances nor present a big overhead on the transferred data.

## 7.5 Evaluation

The examples introduced described up to this point allows the client to granularly disclose information to the SP. More importantly in the current federated log in mechanism, implemented by IdPs like Facebook, third-party apps can always request to the IdP updated information regarding the user. With the proposed mechanism, based on the PB intermediary model, third-party apps will have to request the information to the client, that can refuse to provide them or can provide different values depending on the context.

A typical scenario is the case for which a user wouldn't like to disclose some of their preferences, and will instead send to the SP a forged profile, possibly similar to their actual interest profile but hiding some sensitive information. In this case the user will construct an hypermedia representation of their identity and transmit it to the SP. This representation can simply contain an anonymous id, while the

security ticket generated by the PB will be included as authentication token via JWT, ensuring that the request is valid. In addition the user client can chose to disclose some properties, if requested 7.5.

```
{
    "data": {
        "type": "identity",
        "id": "12fsdaGACYDSAG",
    },
}
```

**Listing 7.5:** Some user identity data encoded with JSONApi only an anonymous id.

We believe this aspect is fundamental for implementing privacy management technologies without having to change the way users interact with web apps, using hypermedia resource representations and without having to change existing web protocols that are so widely adopted. Up to now, privacy enhancement and privacy management solutions have been focusing with the network level of the user's communications. This approach instead shifts the focus to client interactions and existing web protocols that can be used and eventually improved to support such techniques.

Implementation of the examples described are available in various programming languages, as are considered industry standards. We tested the protocol described with the EdDSA implementation provided by ECPy [**?** ], a pure python Elliptic Curve library providing ECDSA, EDDSA, ECSchnorr, Borromean signatures as well as Point operations.

The implementation for the Socialist Millionaire Protocol takes 187 lines of codes in total and takes 0.0562 seconds to run on a Intel Core i7-6600U CPU @ 2.60GHz × 4 cores. On the other hand, the implementation with the same libraries for the challenge implemented in the OAuth example takes only 20 line of code and takes 0.0312 seconds to run on the same machine.

## 7.6 Discussion

The example illustrated above show how web applications can be easily patched to provide users with a more privacy friendly experience. Up to now, in fact, users have been given the choice to either accept the terms of SPs and share all the data requested or not use the service. The model proposed instead would give users control on how they can share the data and present SPs with certain information regarding the users. Furthermore, SPs will not be given the possibility to access user information from the PB as they please, but will have to request users to disclose certain information to the PB. This way users will know exactly which information has been disclosed and in which interaction with SPs, hence giving them the choice to request for the data to be deleted if they wished to do so. We believe this to be an important paradigm change in the way authentication and authorisation with SPs is implemented and a much needed improvement for privacy management on the client side.

*A Jedi uses the Force for knowledge and defense, never for attack.*

Yoda - The Empire Strikes Back

# 8

# Conclusions and discussion

This dissertation examined a class of privacy issues for online communication, proposing a model for the user identity and a possible new approach to information privacy management. This work focused on the analysis of privacy violation that can be found in different scenarios, on web and mobile applications and services. The goal was to convince the reader that, as the web is shifting towards hypermedia data models and protocols, also privacy analysis and protection have to adopt the same mindset.

The motivation behind this work was understanding how data, created by users, flows between applications and services. A very powerful example in this field is the use of federated log in mechanisms. To register to a new social application, users grant them a certain level of access to their identity data, through, for example, their Facebook, Twitter or Google accounts. This data includes details about

their identity, their whereabouts and in some situations even the company they work for. Third parties, like Facebook or Google, offer log in technologies, allowing the application to identify the user and receive precise information about them. Once the user grant access to their data, the application stores it and assumes control over how it is further shared. The user will never be notified again on who is accessing their data, nor if these are transferred to third parties. We showed how this mechanism can be modified to mitigate or avoid this.

We believe that an important aspect of privacy protection is giving web users the possibility to control their digital footprints. More specifically, we are aware that privacy issues involve a plurality of complexities. This is especially true nowadays that privacy has acquired a completely different meaning because people conduct part of their existence through and on communication platforms. Privacy rights need to consider the implication of *information privacy*, given that a person shares parts of their activities, interests and even thoughts with online service providers. As a consequence, the philosophical definition of privacy has evolved, while laws protecting individual privacy rights have tried to follow.

Up to now, in an online context, the right to privacy has commonly been interpreted as a right to *information self-determination*. Acts typically claimed to breach online privacy concern the collection of personal information without consent, the selling of personal information and the further processing of that information. This definition of privacy breach can be considered valid until the user has direct control of the data they have created.

This work started by analysing information filtering systems. These systems have been developed to predict users' preferences, and eventually use the resulting predictions for different services, depend on users revealing their personal preferences by annotating items that are relevant to them. At the same time, by revealing their preferences online users are exposed to possible privacy attacks and all sorts of profiling activities by legitimate and less legitimate entities.

We showed how query forgery arises, among different possible PETs, as a simple strategy in terms of infrastructure requirements, as no third parties or external entities need to be trusted by the user in order to be implemented. However,

query forgery poses a trade-off between privacy and utility. Measuring utility by computing the list of useful results that a user would receive from a recommendation system, we have evaluated how three possible tag forgery techniques would perform in a social tag application. With this in mind a dataset for a real world application, rich in collaborative tagging information has been considered.

It was calculated how the performance of a recommendation system would be affected if all the users implemented a tag forgery strategy. We hence considered an adversary model where a passive privacy attacker is trying to profile a certain user. The user in response, adopts a privacy strategy aiming at concealing their actual preferences, minimising the divergence with the average population profile. The results presented show a compelling outcome regarding how implementing different PETs can affect both user privacy risk, as well as the overall recommendation utility. It was showed how in a simple experimental evaluation, of a real world application scenario, the performances degradation of a recommendation system, is small if compared to the privacy risk reduction offered by the application of these techniques.

Furthermore, we focused on a class of social application that uses the users' actual location to provide personalised recommendation and allow for new interactions especially in urban settings. We have shown how these applications can expose their users to different privacy attacks that can be easily overlooked. We followed a formal framework to identify the classes of privacy violation to which users are subjected to without being aware of it and we have shown how these violations can all be carried out for the applications examined. This shows how using third party profiles to provide access to a specific applications may cause a security *honey pot* for a possible attacker.

We also analysed web users tracking and introduced a set of metrics to show how information is sent to third-party tracking services when users surf the web. We also computed a set of network analysis on our graph model of the user online footprint. We were able to identify known trackers and isolate communities of similar trackers. This aspect is particularly interesting for the development of Privacy Enhancing Technologies for the web. Up to now, anti-tracking technolo-

gies have been built to simply stop third-party requests, alternative strategies might instead consider to send bogus information to certain over-connected tracker domains to masquerade the user real profile. At the same time a measurement of the average degree of the neighbourhood of a certain third-party domain can be used to evaluate how *dangerous* this can be considered for the user's privacy.

We investigated the profile updates leading to the best and worst overall anonymity risk for a given activity and history. The analysis of our model was applied to an experimental scenario, using Facebook timeline data. Our main objective was measuring how privacy is affected when new content is posted. Often, a user of some online service is unable to verify how much a possible privacy attacker can find out about them. We used real Facebook data to show how our model can be applied to a real world scenario. This aspect is particularly important for content filtering in Facebook. In fact, as users are profiled on Facebook, the very same activity is used to filter the information they are able to access, based on their interests. There is no transparency on Facebook's side about how this filtering and profiling happens. We hope that studies like this might encourage users to seek more transparency in the filtering techniques used by online services in general.

We also introduced an example showing how it is possible to modify federated login mechanism to preserve users' privacy, to a certain extent. The example illustrated how web applications can be easily patched to provide users with a more privacy friendly experience. Up to now, in fact, users have been given the choice to either accept the terms of SPs and share all the data requested or not use the service. The model proposed instead would give users control on how they can share the data and present SPs with certain information regarding the users. Furthermore, SPs will not be given the possibility to access user information from the IdP as they please, but will have to request users to disclose certain information to the IdP. This way users will know exactly which information has been disclosed and in which interaction with SPs, hence giving them the choice to request for the data to be deleted if they wished to do so.

Given the extent of privacy issues and violations that are ignored by application developers and service providers, the author believes that the analysis, solutions

and results presented in this dissertation provide the basis to understand these and possibly address them. The author also hopes these results will motivate and provide a solid theoretical basis for additional analysis and privacy management techniques, and, ultimately, have a direct impact over users' privacy by eliminating or reducing barriers to the development of new and existing privacy aware protocols and services.

# Appendix

## 8.1 How the Socialist millionaires protocol works

Imagine that a *Prover* (the client) wants to prove the *Verifier* (the server) that they know a certain pseudonym. Given a known elliptic curve $C$, a generator for such curve $G$ and the curve order the protocol can be outlined as follows:

The Prover initiate the process and sends an authorisation request by transmitting the result of a certain computation:

1. Prover picks two random numbers: $a2$ and $a3$.
2. Prover computes $G2a = a2 * G$ and $G3a = a3 * G$.
3. Prover transmits $G2a$ and $G3a$ to the Verifier.

The Verifier received the two values initiate the authorisation process on their side:

1. Verifier picks two random numbers: $b2$ and $b3$.
2. Verifier computes $G2b = b2 * G$ and $G3b = b3 * G$.
3. Verifier computes $G2B = b2 * G2a$ and $G3B = b3 * G3a$.
4. Verifier picks random number $r$.
5. Verifier computes $Pb = r * G3B$ and $Qb = r * G + y * G2B$.
6. Verifier transmits $G2b$, $G3b$, $Pb$ and $Qb$ to Prover.

The Prover receives the first answer from the Verifier and continues the protocol:

1. Prover computes $G2A = a2 * G2b$ and $G3A = a3 * G3b$.

2. Prover picks random number $s$.

3. Prover computes $Pa = s * G3A$ and $Qa = s * G + x * G2A$.

4. Prover computes $Ra = a3 * (Qa - Qb)$.

5. Prover transmits $Pa$, $Qa$, $Ra$ to Verifier.

Now the Verifier will compute the proof with the values received and ultimately transmit the results of their computation to the Prover authorising the data transfer:

1. Verifier computes $Rb = b3 * (Qa - Qb)$.

2. Verifier computes $Rba = b3 * Ra$.

3. Verifier checks that $Rab == Pa - Pb$.

Now the Prover will verify the result on their side by performing the following computations:

1. Prover computes $Rab = a3 * Rb$.

2. Prover checks whether $Rab == Pa - Pb$.

## Socialist millionaires implemented with EdDSA

```
from ecpy.curves      import Curve,Point
from ecpy.keys        import ECPublicKey, ECPrivateKey
from ecpy.eddsa       import EDDSA
from ecpy.formatters import decode_sig, encode_sig
import time
start_time = time.time()

C = Curve.get_curve('Ed25519')
G = C.generator
q = C.order

import random
import hashlib

x = random.randint(1,q-1)
y = x
```

```
# Alice
# picks two random numbers: a2 and a3

a2 = random.randint(1,q-1)
a3 = random.randint(1,q-1)

# computes G2a = a2 * G and G3a = a3 * G

G2a = C.mul_point(a2, G)
G3a = C.mul_point(a3, G)

# encodes G2a and G3a

eG2a = C.encode_point(G2a)
eG3a = C.encode_point(G3a)

# sends eG2a and eG3a to Bob

# Bob
# picks two random numbers: b2 and b3

b2 = random.randint(1,q-1)
b3 = random.randint(1,q-1)

# computes G2b = b2 * G and G3b = b3 * G

G2b = C.mul_point(b2, G)
G3b = C.mul_point(b3, G)

# encodes G2a and G3a

eG2b = C.encode_point(G2b)
eG3b = C.encode_point(G3b)

# computes G2B = b2 * G2a and G3B = b3 * G3a

G2B = C.mul_point(b2, G2a)
G3B = C.mul_point(b3, G3a)

# encodes G2B and G3B

eG2B = C.encode_point(G2B)
eG3B = C.encode_point(G3B)

# picks random number r
r = random.randint(1,q-1)
```

```
64
   # computes Pb = r * G3B and Qb = r * G + y * G2B
66
   Pb = C.mul_point(r, G3B)
68
   q1b = C.mul_point(r, G)
70 q2b = C.mul_point(y, G2B)
   Qb = C.add_point(q1b, q2b)
72
   # encodes Pb and Qb
74
   ePb = C.encode_point(Pb)
76 eQb = C.encode_point(Qb)

78 # sends eG2b, eG3b, ePb and eQb to Alice
   # Alice
80
   # computes G2A = a2 * G2b and G3A = a3 * G3b
82
   G2A = C.mul_point(a2, G2b)
84 G3A = C.mul_point(a3, G3b)

86 # encodes G2B and G3B

88 eG2A = C.encode_point(G2A)
   eG3A = C.encode_point(G3A)
90
   # picks random number s
92 s = random.randint(1,q-1)

94 # computes Pa = s * G3A and Qa = s * G + x * G2A

96 Pa = C.mul_point(s, G3A)

98 q1a = C.mul_point(s, G)
   q2a = C.mul_point(x, G2A)
100 Qa = C.add_point(q1a, q2a)

102 # encodes Pa and Qa

104 ePa = C.encode_point(Pa)
   eQa = C.encode_point(Qa)
106
   # computes Ra = a3 * (Qa - Qb)
108
   Qab = C.sub_point(Qa, Qb)
```

```
Ra = C.mul_point(a3, Qab)

# encodes Ra
eRa = C.encode_point(Ra)

# sends ePa, eQa, eRa to Bob

# Bob

# computes Rb = b3 * (Qa - Qb)

Qba = C.sub_point(Qa, Qb)
Rb = C.mul_point(b3, Qba)

# encodes Rb
eRb = C.encode_point(Rb)

# computes Rba = b3 * Ra

Rba = C.mul_point(b3, Ra)
Pba = C.sub_point(Pa, Pb)

# encodes Pba and Rba

ePba = C.encode_point(Pba)
eRba = C.encode_point(Rba)

# checks whether Rab == Pa - Pb

if (eRba == ePba):
    print("Yolo")
else:
    print("Rba = " + str(eRba))
    print("Pba = " + str(ePba))

# sends Rb to Alice

# Alice
# computes Rab = a3 * Rb

Rab = C.mul_point(a3, Rb)
Pab = C.sub_point(Pa, Pb)

# encodes Pab and Rab

ePab = C.encode_point(Pab)
```

```
156 eRab = C.encode_point(Rab)

158 # checks whether Rab == Pa - Pb

160 if (eRab == ePab):
        print("Yolo")
162 else:
        print("Rab = " + str(eRab))
164     print("Pab = " + str(ePab))

166 print("--- %s seconds ---" % (time.time() - start_time))
```

**Listing 8.1:** Socialist Millionairs Proof implemented with EdDSA.

## OAuth 2.0 challenge with EdDSA

```
start_time = time.time()
2
def sha512(s):
4       return hashlib.sha512(s).digest()

6 # Bob knows Q = x * G
# Given a curve with generator G and its order q Alice can
    prove to Bob
8 # she knows 0 < x < n such that Q = x * G by:

10 Q = x * G
r = random.randint(1,q-1)
12 W = r * G
eW = C.encode_point(W)
14 ec = sha512(eW)
c = int.from_bytes(ec,'big')
16 d = r - x*c%q

18 # presenting (c, d) to Bob Bob verifies the proof by
    checking:
pA = C.mul_point(d,G)
20 pB = C.mul_point(c, Q)

22 pnt = C.add_point(pA, pB)
epnt = sha512(C.encode_point(pnt))
24 if  (ec == epnt):
        print("Yolo")
26 else:
```

```
    print("Nope")

print("--- %s seconds ---" % (time.time() - start_time))
```

**Listing 8.2:** Alice wants to prove their identity to Bob without revealing their pseudonym. The proof has been implemented with EdDSA.

# Colophon

This thesis was typeset using LaTeX, originally developed by Leslie Lamport and based on Donald Knuth's TeX. The body text is set in 11 point Arno Pro, designed by Robert Slimbach in the style of book types from the Aldine Press in Venice, and issued by Adobe in 2007. A template, which can be used to format a PhD thesis with this look and feel, has been released under the permissive MIT (X11) license, and can be found online at github.com/suchow/ or from the author at suchow@post.harvard.edu.