KU LEUVEN

ARENBERG DOCTORAL SCHOOL
Faculty of Engineering Science

Private Information Sharing in Online Communities

Filipe Beato

Supervisor:
Prof. dr. ir. Bart Preneel

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor in Engineering

May 2015

Private Information Sharing in Online Communities

Filipe BEATO

Examination committee:
Prof. dr. Adhemar Bultheel, chair
Prof. dr. ir. Bart Preneel, supervisor
Prof. dr. Claudia Diaz
Prof. dr. Bruno Crispo
Prof. dr. ir. Vincent Rijmen
Prof. dr. Emiliano De Cristofaro
(University College of London, London, UK)
Dr. Andreas Pashalidis
(Bundesamt für Sicherheit in der Informationstechnik, Germany)

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor in Engineering

May 2015

to my beloved parents . . .

Acknowledgments

"Porque é tamanha bem-aventurança
O dar-vos quanto tenho e quanto posso,
Que quanto mais vos pago, mais vos devo."
– Luís Vaz de Camões, *Rimas (1595)*

I am deeply indebted to all who have supported and encouraged me during my doctoral journey. I would like to take this opportunity to express my gratitude to the people from whose advice and influence this work significantly benefited.

First and foremost, I would like to thank my promotor Prof. Bart Preneel, for giving me the opportunity to pursue a Ph.D. in COSIC, as well as for his guidance, time, and trust to allow me to freely conduct my research.

I would like to thank Prof. Claudia Diaz and Prof. Dake Clarke for being my assessors and for all the insightful comments, feedback, and milestone approvals throughout the Ph.D. roadmap of the Arenberg doctoral school.

I want to express my gratitude to my jury members: Prof. Vincent Rijmen, Prof. Bruno Crispo, Prof. Emiliano de Cristofaro, and Dr. Andreas Pashalidis for their effort in reading this thesis along with their wise feedback, questions, and precious comments; and Prof. Adhemar Bultheel for chairing the jury.

I am indebted to my co-authors for the many fruitful discussions, valuable and helpful comments, and patience during deadlines that considerably contributed to this thesis: Emiliano, Ero, Fabio, Iulia, Karel, Kasper, Koen, Mauro, Markulf, Roel, Seda, and Stijn. I would like to thank Prof. Gene Tsudik for welcoming me to UCI for six months, and for introducing me to a number of exceptional people and researchers: Cesar, Kasper, Mish, Naveen, Paolo, and Sky. I am also grateful to Prof. Mauro Conti for the extensive collaboration work, research advise, and for inviting me to visit Padova and experiencing Spritz.

Working at COSIC has been a honorable and interesting adventure that made these years of research particularly enjoyable. Thank you all (past and current) for your wisdom, alma lunches, karting events, COSIC events, and annual sports days. I am extremely grateful to Jens, Saartje, and Svetla for welcoming me at their (tiny) office and for providing the friendliest and nicest office environment during most of my journey. Bedankt Roel for continuously taking over my chair every time I would leave the office, keeping it warm. Special thanks to Andreas, Anthony, Atul, Bart (M.), Begül, Elmar, Iraklis, Josep, Karel, Kimmo_0, Laura, Nikos, Oscar, Stefan, and Tomer for the lunches, coffee breaks, and interesting discussions. Thanks to Ruan and Pieter for allowing me to crash their office and use their couch, and Gunes and Marc for the sponge ball challenges. Bedankt Karel for initial advise, motivational pushes, and motorbike discussions. Bedankt Fre for the dutch translation of my beknopte samenvatting, and Kimmo_1 for continuously proof-reading this thesis. Thanks to all the members of the privacy group for the endless meetings, discussions, and feedback that continuously enlightened my research. A very special thanks goes to Péla Nöe for her patience, kindness, and continuous help dealing with administrative bureaucracy and many other problems I kept having.

I am forever grateful to all my great friends in Brussels for their friendship, support, and patience that made my time in Brussels an unforgettable (continuous) experience. Merci mille fois, mille fois merci!!

I would like to extend my gratitude to all my closest friends from Portugal and all the others spread around the globe for there reliable advice and unsurpassed moral support which made the distance from home almost negligible. Obrigadão malta!!

Last, but definitely not least, I want to thank my parents, my partner and her family for the continuous love, unlimited patience, and support. I am truly lucky to have you all! Obrigado! Sem vocês nada disto seria possível! Je ne sais pas comment vous remercier! Julie merci d'être toujours là pour moi quand rien ne va et pour tous les moments magiques.

Thank you! Obrigado! Merci!
Dankjewel! Danke! Grazie! Gracias!

I would like to acknowledge the Fundação para a Ciência e Tecnologia (FCT) for funding my research with the FRH/BD/70311/2010 grant.

FILIPE BEATO
Leuven, May 2015

Abstract

"As far as I'm concerned, if something is so complicated that you can't explain it in 10 seconds, then it's probably not worth knowing anyway."
– Bill Watterson, *The Indispensable Calvin and Hobbes (1992)*

In a modern society highly focused on digital services, online communities, such as Online Social Networks (OSNs) have taken the world by storm, boasting users in the hundreds of millions, mainly by providing easy and reliable channels for dissemination of information, as well as seamless coordination of social activities. At the same time, OSNs create treasure troves of sensitive information, collecting and processing large amounts of data about the users and their activities, leading to several privacy concerns. Although traditionally motivated by the targeted advertisement based business model, OSNs have also become primary targets of cyberbullying, security breaches, and government (mass) surveillance actions. The users' lack of awareness and little to no control over the content published on OSNs, aligned with the importance of privacy as a human right, makes privacy a crucial problem to be addressed.

In this thesis, we propose privacy-enhancing solutions that provide users with more control over the shared content on OSNs, while enforcing privacy by means of practical and efficient cryptographic primitives. Henceforth, we categorize the general privacy problems and define access control based on group definitions. Then, we devise a collaborative sharing scheme that allows to define access control rights on content that is made available on OSNs and that is related to multiple users.

Furthermore, we provide information sharing schemes for OSNs, focused on delivering and enforcing privacy as content confidentiality for multiple recipient

and group scenarios, such that OSN providers are kept oblivious of the shared content and its intended recipients. In addition, we model the notion of undetectable communication in the context of OSNs, and subsequently design a general covert information scheme that builds on top of any privacy sharing scheme delivering provable undetectability.

Finally, we develop a system for browsing OSNs anonymously, while taking advantage of the high-availability storage and communication tools from modern OSNs, while private communication is performed through an external network built upon the social trust delivered by users' connections. For each solution proposed in this thesis we develop practical tools demonstrating its efficiency and practical impact.

Beknopte samenvatting

In onze moderne maatschappij spelen digitale diensten een belangrijke rol en zijn het vooral de online gemeenschappen, zoals Online Sociale Netwerken (OSN), die een exponentiële groei gekend hebben resulterend in miljoenen gebruikers. Het succes van deze OSNs is te danken aan de makkelijk bruikbare en vooral betrouwbare manier waarop informatie kan verspreid worden en waarmee sociale activiteiten op elkaar kunnen worden afgestemd. OSNs hebben echter ook te kampen met privacyproblemen omdat ze gevoelige informatie over de gebruikers en hun activiteiten verzamelen en bewerken. Alhoewel hun oorspronkelijke bestaansreden vooral moet gezocht worden in gerichte reclame, worden OSNs nu ook misbruikt voor cyberpesten, beveiligingsinbreuken en spionage door de overheid. Het feit dat de meeste gebruikers zich niet van deze gevaren bewust zijn en in veel gevallen weinig controle hebben over wat er op OSNs gepubliceerd word, gecombineerd met het feit dat privacy een basisrecht is, zijn de hoofdredenen waarom we deze problemen bestuderen.

In deze thesis stellen we oplossingen voor die de privacy van gebruikers verhoogt en hen meer controle geeft over de informatie die op OSNs gedeeld wordt; onze oplossingen garanderen de privacy door middel van praktische en efficiënte cryptografische bouwblokken. Daarnaast stellen we een indeling van privacyproblemen voor en definiëren we toeganscontrolemechanismen op basis van groepsdefinities. Bovendien stellen we ook een schema op dat gebruikers van OSNs toelaat om informatie gerelateerd aan meerdere gebruikers te delen door middel van toegangscontrolerechten.

We leiden ook schemas af om informatie op OSNs te delen, waarbij de nadruk ligt op privacy voor gebruikers. Hierbij maken we gebruik van encryptietechnieken voor meerdere ontvangers of gebruikersgroepen, waarbij de OSN providers niets kunnen afleiden over de informatie die gedeeld wordt en met wie deze informatie gedeeld wordt. Bovendien modelleren we ook het begrip “niet-detecteerbare communicatie” voor OSNs en ontwerpen we een algemeen verdoken

informatieschema dat werkt bovenop ieder bestaand privacy deelschema en bewijsbaar niet-detecteerbaar is.

Tenslotte ontwikkelen we ook een systeem om anoniem gebruik te maken van OSNs, dat de gekende voordelen van een OSN blijft behouden zoals hoge beschikbaarheid, opslag en communicatiemogelijkheden. Hierbij gebeurt privé communicatie via een extern netwerk gebouwd op het sociale vertrouwen dat afgeleid wordt uit de verbindingen tussen de verschilllende gebruikers. Voor iedere oplossing die we in deze thesis voorstellen tonen we ook de efficiëntie en praktische bruikbaarheid aan door middel van demonstrators.

Contents

List of Figures

List of Tables

List of Abbreviations

AES	–	Advanced Encryption Standard
ANOBE	–	Anonymous Broadcast Encryption
API	–	Application Programming Interface
ATE	–	ATE pairing
BC	–	BouncyCastle Library
BDH	–	Bilinear Diffie-Hellman Problem
BE	–	Broadcast Encryption
BLS	–	Barreto Lynn Scott curves
BN	–	Barreto Naehrig curves
BTP	–	Biometric Template Protection
CBC	–	Cipher Block Chaining mode
CCA	–	Chosen-Ciphertext Adversary
CCM	–	Counter with CBC-MAC mode
CDH	–	Computational Diffie-Hellman Problem
CPA	–	Chosen-Plaintext Adversary
DDH	–	Decision Diffie-Hellman Problem
DLP	–	Discrete Logarithm Problem
DKG	–	Distributed Key Generation
DPD	–	Data Protection Directive 95/46/EC
ECC	–	Elliptic Curve Cryptography

FIPP	–	Fair Information Practice Principles
FQL	–	Facebook Query Language
GCM	–	Galois Counter Mode
GCHQ	–	Government Communications Headquarters (UK)
HTTP	–	Hypertext Transfer Protocol
HTTPS	–	Secure Hypertext Transfer Protocol
IBE	–	Identity-Based Encryption
IND	–	Indistinguishable
IP	–	Internet Protocol
JSON	–	JavaScript Object Notation
MAC	–	Message Authentication Code
MB	–	Megabyte
NSA	–	National Security Agency (US)
OAuth	–	Open Standard for Authorization
OTR	–	Off-The-Record messaging
OSN	–	Online Social Network
PETs	–	Privacy-Enhancing Technologies
PGP	–	Pretty Good Privacy
PK	–	Public Key
PKG	–	Private Key Generator
PRF	–	Pseudo-Random Function
PS	–	Private-Sharing Scheme
Tor	–	The Onion Routing protocol
SHA	–	Secure Hash Algorithm
SJCL	–	Standford Javascript Cryptographic Library
SK	–	Symmetric Key
SOCKS	–	Socket Secure
SSL	–	Secure Socket Layer

QR	–	Quick Response code
TLS	–	Transport Layer Security
VSS	–	Verifiable Secret Sharing

List of Symbols

$\mathtt{m}$	–	Plaintext message
$\mathtt{C}$	–	Ciphertext message
λ	–	Security parameter
$\{0,1\}^n$	–	Set of all bit strings of size n
$\lvert x\rvert$	–	Length of bit string x
$\oplus$	–	Bitwise XOR, e.g., $x \oplus y$
$\lVert$	–	Concatenation of two or more variables, e.g., $x \lVert y$
$\cup$	–	Set union, $X \cup Y = \{x : x \in X \vee x \in Y\}$
$\cap$	–	Set intersection, $X \cap Y = \{x : x \in X \wedge x \in Y\}$
$\triangle$	–	Set symmetric difference, $X \triangle Y = \{x : (x \in X) \oplus (x \in Y)\}$
$\mathtt{k}$	–	Symmetric Key
(pk,sk)	–	Asymmetric Public-Private Key Pair
(vk,sgk)	–	Digital Signing-Verifying Key Pair
(mpk,msk)	–	Master Public-Private Key Pair
$\mathsf{E}(\cdot),\mathsf{D}(\cdot)$	–	Symmetric Encryption and Decryption Algorithms
$\mathtt{Enc}_{pk}(\cdot),\mathtt{Dec}_{sk}(\cdot)$	–	Asymmetric Encryption and Decryption Algorithms
$\mathtt{Sign}_{sgk}(\cdot),\mathtt{Ver}_{vk}(\cdot)$	–	Digital Signature and Verification Algorithms
$\mathtt{H}(\cdot)$	–	Hash Function, possibly modeled as a random oracle
$\mathtt{PRF}(\cdot)$	–	Pseudo-Random Function
$\mathtt{MAC}(\cdot)$	–	Message Authentication Code
$\mathtt{Pr}[x]$	–	Probability Distribution of the random variable x

$H(x)$	–	Shannon Entropy of x
$\mathcal{P}$	–	User OSN Profile
$\mathcal{R}$	–	User OSN Connections, i.e., Friendship Connections
$\mathcal{L}$	–	Group of Connections
$\mathcal{S}$	–	Set of Intended Recipients
$\mathcal{F}$	–	Routing Friend

1

Introduction

"All human beings have three lives: public, private, and secret."
– Gabriel García Márquez, *A Life (2010)*

Modern society has experienced a considerable transformation, with people moving towards a digital communication era. Aligned with the vast impact of the Internet, online communities, such as online social networks, blogs, and wikis, have taken the world by storm and conquered society. Online Social Networks (OSNs), such as Facebook, Twitter, Google+, and Linkedin, represent the most popular type of online community; experiencing an enormous growth in the past years by attracting large amounts of users and are becoming integrated into the daily routines of many. OSNs provide users with a panoply of options and reliable communication channels for sharing information, as well as seamless coordination of social activities in an ubiquitous, simple, and convenient manner.

Boyd and Ellison [40] define OSNs as a web service that allows users to perform three main actions, as follows.

> *"... (1) construct a public or semi-public profile within a bounded system, (2) articulate a list of other users with whom they share a connection, and (3) view and traverse their list of connections and those made by others within the system."*

Besides all their advantages, OSNs represent also repositories of vast amounts of sensitive information, by storing content related to the users and their activities. The most popular OSNs provide "privacy settings," so that users can select which other users (or groups) can access their content. However, this process relies not only on the diligence of the users but also on the trustworthiness of the providers in enforcing access control and protecting stored content from possible adversaries. While the risk of negative publicity and lawsuits deliver providers the incentives to safeguard user information, end-user license agreements often include legal clauses allowing data mining or information selling to third-party services [171]. In particular, OSNs frequently employ large-scale mining to improve the quality of service and deliver targeted advertisement, as part of their business model. Consequently, all the exposed information on OSNs allows the entities in control to infer sensitive information about users, such as interests, whereabouts, social circles, or even political and sexual orientations. Moreover, the concentration of personal information in a single system also exacerbates the dangers of data leaks [38] and insider attacks [157, 161].

OSNs have also become primary sources of obtaining private information, as well as targets of government surveillance and Internet censorship. For instance, the number of subpoenas issued by U.S. law enforcement agencies on OSN data has increased steadily over the past few years [147], whereas the existence of government surveillance has been demonstrated by the documents leaked by Edward Snowden reporting the NSA PRISM project [204]. In addition, Twitter has also demonstrated their ability to censor content on a country basis, complying with governments' requests to remove or block certain content [37]. Many countries worldwide are reported to block, selectively filter, or perform censorship on OSNs, as demonstrated by the last three yearly reports of the Freedom House association [91, 92, 93].

Privacy and surveillance has been a concern and topic of research addressed by Privacy-Enhancing Technologies (PETs) with cryptographic mechanisms playing a key role in producing a number of effective privacy-preserving protocols [102, 103, 114]. However, the novel nature of OSNs introduced different communication changes and thus new privacy challenges, as demonstrated by Acquisti and Gross [3] and later supported by Chew *et al.* [54]. In fact, the complex and ever changing OSN environment presents various new and hard to classify privacy problems, only recently categorized by Gürses and Diaz [113] into three distinct but entangled classes: surveillance, social privacy, and institutional privacy; they stress the fact that shared content on OSNs should be shared (privately) and accessed only by its indented recipients, and not by any extra entity.

With the significant communication changes introduced by OSNs, the information exchanged and shared by users on OSNs can be accessed and

collected by multiple (unwanted) parties, directly affecting users' privacy. Hence, this motivates the need for user-controlled (private) sharing schemes that intend to provide protecting OSN user's privacy, while translating the offline social practices into the OSN sphere [198]. This thesis studies and proposes privacy-enhancing solutions aiming at providing users with more control over the shared content on OSNs, while enforcing privacy by means of practical and efficient cryptographic primitives. In short, this thesis presents privacy-enhancing solutions while addressing the following main technical challenges:

1. How to enable private sharing of information on OSNs, so that unwanted parties learn as limited information as possible regarding the content and the intended recipients during the communication.
2. How to implement those solutions efficiently for the current OSN design.

1.1 Thesis Contributions

As part of a user-centric web, it is fundamental that users are empowered with control of their shared information. In particular, we aim at designing cryptographic privacy-preserving protocols for protecting information shared on OSNs against non-intended recipients, while disclosing as limited information as possible regarding the content, the user identity, and the intended recipients.

Therefore, this thesis investigates privacy and security mechanisms for protecting information shared in Online Communities, while taking into account their current limitations. We detail the contributions of this thesis as follows:

- We propose a Collaborative joint protocol based on secret sharing that achieves confidentiality and allows collaborative joint access control definitions for OSNs (Chapter 5).
- We design privacy-enhancing schemes for privately sharing information among multiple recipients on OSNs based on cryptographic primitives, that keep any unauthorized user oblivious of the content and the identity of the intended recipients (Chapter 6).
- We model undetectability in the context of OSNs and suggest a general covert sharing scheme achieving undetectable communication (Chapter 7).
- We also devise a system that allows users to browse OSNs while keeping their traces anonymous towards the provider, by relying on friendship connections (Chapter 8).

At the same time, we developed practical tools, such as Scramble (Appendix A) and VirtualFriendship (Chapter 8), that demonstrate the feasibility and practical impact of the proposed schemes on modern OSNs.

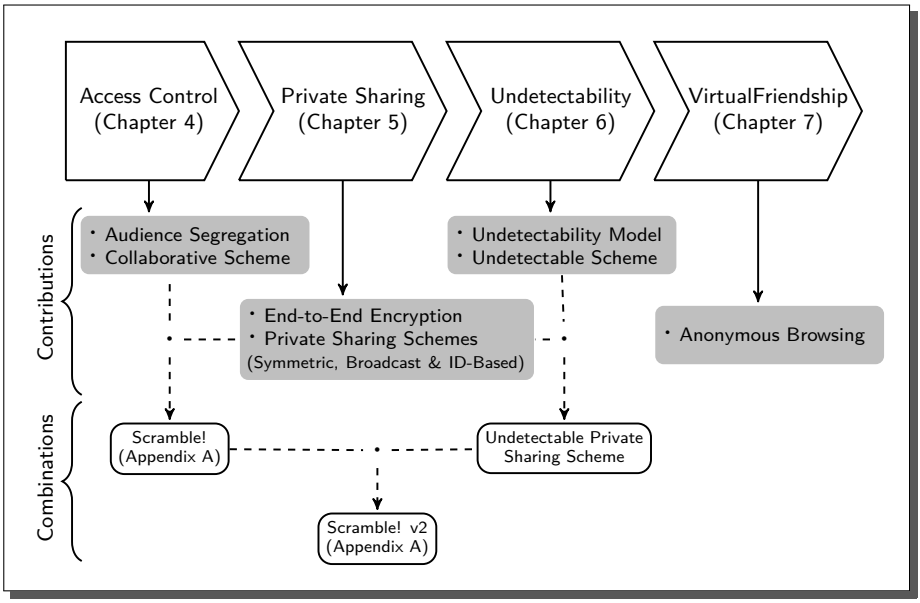


Figure 1.1: Overview of the individual contributions per chapter along with the chapter combinations contributions.

1.2 Thesis Structure

This thesis investigates privacy and security mechanisms for protecting information sharing in Online Communities; it is divided into two main parts: Preliminaries (Part I) and Private Information Sharing in Online Communities (Part II). We begin in Part I by enumerating and classifying the privacy problems present in OSNs, along with the associated literature review, followed by the theoretical cryptographic and privacy background and notation used throughout this thesis. In Part II we present our contributions in the form of self-contained chapters each derived from one or a combination of several scientific publications. Each chapter (Chapter 5–8) delivers a single or multiple contributions offering extra privacy properties (contributions) when combined, as depicted in Figure 1.1. In particular, we address the different privacy issues from different perspectives, starting from access control based solutions followed by end-to-end encryption, key management, the formalization of undetectability, and we attend to the anonymity problem within OSNs.

A brief summary of each of the chapters and the associated contributions is presented as follows.

Part I – Preliminaries

Chapter 2 – Privacy in OSNs. This chapter gives a brief introduction to the topic of privacy, as well as an in-depth motivation of the needs of privacy-enhancing technologies in OSNs, and the solutions developed throughout this thesis. Aligned with the current privacy definitions in computer science, we first discuss several privacy threats existent in OSNs and the current approaches taken by OSNs. Then, we emphasize the categorization of privacy problems in OSNs and the motivation for the solutions throughout this thesis. Further, we discuss the currently available privacy-preserving solutions for OSNs.

Chapter 4 – Background and Notation. In this chapter we survey the cryptographic and privacy background, alongside the general notation used throughout this thesis. Readers familiar with cryptography concepts, such as identity-based encryption, broadcast encryption, and secret sharing, as well as anonymity definitions, can skip this chapter without loss of continuity.

Chapter 3 – Literature Review. In this chapter we review the associated related work with respect to each of the different chapters. At the same time, we compare the work proposed with our contributions, highlighting the differences and advantages of our solutions.

Part II – Private Information Sharing in Online Communities

Chapter 5 – Audience Segregation. In this chapter, we bridge the offline notions of audience segregation based on groups and collaboration with the OSN ecosystem. In addition, we present a construction of a collaborative sharing scheme for joint content in OSNs, such that multiple content-related users can collaboratively exercise access control right choices.

Chapter 6 – Information Sharing. In this chapter, we model the notion of *end-to-end encryption* on OSNs, alongside with three private information sharing protocols designed to secure covert information in OSNs, by means of different cryptographic primitives. For each, we study the security, key management, and implementation challenges. Then, we compare the proposed protocols based on key management issues, storage, and efficiency properties.

Chapter 7 – Undetectability. In this chapter, we explore the notion of *undetectable communication* in OSNs. As a result, we introduce formal definitions alongside system and adversarial models, that complement better understood notions of anonymity and confidentiality. We present a novel scheme, as extension to the ones of Chapter 6 achieving undetectable communication in OSNs. We also discuss, via an open-source implementation, the overhead and demonstrate that it is acceptable.

Chapter 8 – Hiding Interactions. Even though encryption is employed, it is still possible to infer sensitive information from browsing behavior. In this chapter, we devise a system allowing users to anonymously browse content available on OSNs, while profiting from the high-availability, storage, and communication tools of current OSNs. The system relies on the social trust delivered by the user social connections to relay traffic on the OSN.

Chapter 9 – Conclusions. Finally, in this chapter, we outline the results of the research from this thesis, along with the respective implications. At the same time, we discuss challenges for future research on related topics.

1.3 Associated Publications

This thesis incorporates material and concepts from different publications in conferences, workshops, and journal articles, co-authored with other researchers. Chapter 5 represents the result of the initial work presented at HotPets, merged with the work developed with Roel Peeters, and published in the proceedings of the IEEE PerCom workshop SESOC 2014 [24]. Chapter 6 presents the results of joint work with Markulf Kohlweiss, Karel Wouters, Iulia Ion, and Stijn Meul, published in the proceedings of the PETs 2011 [22], ACM CODASPY [20] conference, and as the COSIC technical report [23],[1] respectively. The material of Chapter 7 exhibits the published work at IEEE PST 2014 [19], from the joint work with Emiliano De Cristofaro and Kasper B. Rasmussen. Chapter 8 is the outcome of the collaborative work with Mauro Conti published in the IEEE SESOC [17] workshop 2013 and in the IEEE CNS 2014 [18] proceedings. All authors contributed equally to the above publications.

[1] Currently under submission.

1.4 Further Contributions

Alongside the materials published in this thesis, we made further contributions with other researchers. Even if these are not closely aligned with the research questions of this thesis, these contributions have played an important role in developing the results presented in this thesis. Therefore, we now describe the extra contributions, as follows.

Secure and Privacy-Friendly Public Key Generation and Certification, Borges *et al.* [34]. This paper addresses the bootstrap problem of key generation and certification that highly relies on the twofold requirements: an out-of-band verification for certifying keys generated by clients, or a trusted server generating both the public and secret parameters. It devises a novel constrained key agreement protocol, built upon a constrained Diffie-Hellman, to generate a secure public-private key pair and set up a certification environment without disclosing any private parameter from the client to the server. In this way, servers can guarantee safe parameter generation as well as direct key certification, while not learning any secret parameter from clients.

· [34] Borges, F., Martucci, L. A., Beato, F., and Mühlhäuser, M., Secure and privacy-friendly public key generation and certification. In *IEEE TrustCom 2014* (Sep. 2014), Y. Liu, Ed., IEEE, pp. 114–121.

Criteria Towards Metrics for Benchmarking Template Protection Algorithms, Simoens *et al.* [182]. This paper provides the first holistic approach to the evaluation of Biometric Template Protection (BTP) that can cover a whole range of methods. We present a selection of well-defined criteria and some metrics that are compliant with the reference architecture for template protection as defined in the recently adopted standard ISO/IEC 24745 (2011), which is applicable to nearly all known BTP methods. The criteria have been grouped in three categories of performance: technical, protection, and operational.

· [182] Simoens, K., Yang, B., Zhou, X., Beato, F., Busch, C., Newton, E., and Preneel, B., Criteria Towards Metrics for Benchmarking Template Protection Algorithms. In *IAPR ICB 2012* (Mar.-Apr. 2012), A. K. Jain, A. Ross, S. Prabhakar, and J. Kim, Eds., IEEE, pp. 498–505.

Privacy in Social Software, van den Berg *et al.* [199]. This book chapter deals with social software addressing the lack of control users hold over their

own personal data as well as the awareness of its disclosure. Therefore, it presents specific requirements along with practical software solutions addressing these problems for two types of online communities: online social networking sites and web forums. This book chapter served as an early introduction to the research questions addressed in this thesis.

· [199] Beato, F., Borcea-Pfitzmann, K., Leenes, R., Potzsch, S., and Van den Berg, B., Privacy in Social Software. In *Privacy and Identity Management for Life* (2011), J. Camenisch, S. Fischer-Huebner, and K. Rannenberg, Eds., Springer-Verlag, pp. 33-60.

Part I

Preliminaries

2

Privacy in Online Social Networks

"... a cegueira é uma questäo privada entre a pessoa e os olhos com que nasceu."
– José Saramago, *Ensaio sobre a Cegueira (1995)*

Entangled with the enormous popularity of Online Communities, such as Online Social Networks (OSNs), privacy has become a topic of interest within a broad range of disciplines in today's digital society. However, due to the multiple perspectives in varying disciplines and contexts, a single definition of privacy is a complex and ever evolving task [58, 184, 185]. Although it is not the goal of this chapter to present a concrete definition of privacy, it introduces the different privacy definitions within computer science and OSNs alongside the privacy approach followed in this thesis.

After giving a short overview on the history and evolution of privacy, we study in this chapter the notion of privacy in OSNs as used throughout this thesis. In particular, we start by presenting privacy threats and associated motivational examples. Afterwards, we enumerate the different privacy definitions used in computer science introduced by Gürses [111] along with the standard approaches taken by OSNs for the different privacy definitions. Then, following the categorization from Gürses and Diaz [113], we emphasize privacy-enhancing technologies and, specifically, the surveillance problem as the main focus of this thesis.

2.1 Privacy: Short Overview

Privacy is a concept widely used in a broad range of disciplines, such as philosophy, law, and political science, yet there is no single common definition. In fact, each discipline suggests different definitions, views, and classifications while demonstrating the importance of privacy as a meaningful and valuable concept.

Historically *privacy* was defined in 1890 by the legal scholars Warren and Brandeis as "the right to be left alone" [203], emphasizing the right to privacy as the importance of controlling the dissemination of information about oneself with respect to a person's private life. The legal right to privacy was constantly used in the early sixties comprising four distinct intrusions: intrusion into private affairs, public disclosure of embarrassing facts, bad publicity, and identity theft, as categorized by Prosser [167]. Westin [205] presented later the concept of "self-determination" and the ability of individuals to control information as a key requirement for a free society, as well as an instrument to avoid exploitation, discrimination, and false judgment [184]. Gerstein in 1978 lined privacy to the usual cultural borders placed by people's intimacy and relationships [98]. Since then legislators and governments promulgated laws to protect users private. However, the constant evolution of digital society, the easiness of collecting a large amount of data, and the processing of digital data by organizations, amplified the importance for digital privacy. Recently, legislators have been struggling to produce regulations, able to balance the needs of data collection towards a more transparent and controlled perspective, such as the Data Protection Directive (DPD)[1] from the European Union, and the Fair Information Practice Principles (FIPPs)[2] from the Federal Trade Commission in the United States. However, the definition and enforcement of such regulations generally stumbles on the constant changing and complex digital ecosystem, along with changing cultural attitudes, and different discipline views.

In computer science the concept of privacy was introduced in 1981 by Chaum [52], envisaging the support of anonymous communications in a digital world under surveillance, as the protection of the content and the identity of the entities involved in a digital communication. Chaum initial efforts were enabled by public-key cryptography and became influential to development of the privacy-enhancing technologies (PETs) research, leading to the development of several research branches tackling different privacy problems. For instance,

[1] Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 *on the protection of individuals with regard to the processing of personal data and on the free movement of such data.* Official Journal L 281, 23/11/1995 P.0031-0050.

[2] Federal Trade Commission, Privacy Online: Fair Information Practices in the Electronic Marketplace 36–37, May 2000.

communication anonymity tools, such as Chaum Mixes [52, 53] and later Tor [79] provide anonymity (see. Definition 8). In addition, identity management systems, such as anonymous credentials [41, 47] provide privacy during entity authentication revealing solely a proof that the users hold the specific attributes required for authentication without disclosing any extra information.

2.2 Privacy vs. Online Social Networks

In the past decade, OSNs achieved enormous popularity impacting people's behavior, as shown by Boyd [40]. At the same time, OSNs constitute a vast source of centralized sensitive information, leading to several privacy threats [54] and social inconvenience [170]. Although usually stemmed from the OSN design [115] and business model [171], the extensively available social tools and users' (careless) actions amplify the privacy impacts, as demonstrated by Wang *et al.* [202]. The broad impact of privacy issues has been reported by researchers [54] and demonstrated by several events reported by popular media on disclosure of embarrassing information [139], social scams [170], and user retaliations against new features [14, 165], increasing the poor privacy reputation of OSNs [115].

Initially, a few efforts analyzed the multitude of privacy issues existent in popular OSNs, such as Facebook, Twitter, and MySpace, e.g., [3, 54, 39, 81]. The early findings of Acquisti and Gross [3] show that the careless attitude towards privacy of younger generations when using OSNs was due to their lack of awareness. More recent studies by Boyd [39] and later by Dwyer [81], demonstrate that most OSN users are in fact concerned about privacy, and often adapt their actions and minimize the data shared on OSNs. However, one can deduce that the resultant (successful) privacy protest events gathering a vast number of OSN users, such as Facebook Beacon [165], Newsfeed [170], and, more recently, the surveillance Prism project [204] indicates a wide user-base support for the need of more privacy-friendly solutions.

In this section we survey specific privacy threats, the associated consequences, and examples supporting the need for privacy in OSNs. Then, we sketch the current paradigms categorizing privacy within Computer Science research, as well as, the privacy problems classes existent in OSNs. For each, we demonstrate the importance along with the approach used by the solutions described in this thesis.

2.2.1 Privacy Threats: Motivational Examples

The following privacy threats serve as a motivational reference on the importance of privacy-enhancing technologies in the context of OSNs, and thus, supporting the goals of this thesis.

Censorship and Surveillance. Aligned with the large proliferation of data, and the fundamental role played in coordinating and amplifying grassroots movements demonstrated during the recent Arab Spring uprising [191], OSNs have become primary targets of government surveillance [70, 177], and censorship [65]. Even if those are commonly used measures employed by oppressive regimes, as showed by Abdelberi *et al.* [1], other "democratic" countries also employ similar means. In particular, the number of subpoenas issued by United States law enforcement agencies on OSN data has increased steadily over the past few years [147]. Further, the documents leaked by Snowden report more extreme surveillance with the GCHQ and NSA, e.g., the Prism project [204]. In light of the analysis performed by the independent organization Freedom House, many countries worldwide are reported to block, selectively filter, and perform censorship on OSNs [93]. Consequently, large collections of information of users is assembled, known as "surveillant assemblage" [116], which alongside mining tools may lead to the retrieval of sensitive information [136, 177].

Data leakage and collection. OSNs allow users to openly share, and exchange large amounts of data with their friends and with other users. Data leakage is considered to be the process of data disclosure to, or collection by, any non-intended recipient without previous consent of the user sharing the content. Although most shared data on OSNs already leaks unintentionally some sensitive data [136], OSN intentionally grant third party companies, such as advertisement and applications companies, for economical reasons additional access without the user's consent [171], amplifying the leakage of sensitive information. Several cases on data leakage and collection have been reported by the news media. For instance, Facebook has already been sued for reading private messages of their users [161], while Twitter admitted to extract address books from user's phones in order to obtain more information related to the users [157]. Despite the fact that data leakage is often associated to security breaches (e.g., the iCloud breach [139]), and software bugs (e.g., Facebook bug leaks 6 Million people's data [38]), other events and studies demonstrate that OSNs also leak information without users' consent [170]. With the current technical means and the OSN communication channels all the leaked and collected data can easily go viral, and have high impact on user's privacy.

Profiling. This process classifies users based on the collection and classification of data collected according to their actions, behaviors, and patterns. This is a commonly used tool in advertising, in order to screen and identify the "perfect target candidate" for specific advertisements. With the information troves presented by OSNs, all data shared and actions performed contain highly valuable information. In particular, Facebook is considered to earn between $5 - 10$ US dollars per user [64], boosting their revenue from 1.97 Billion dollars in 2010 to 12.47 Billion dollars in 2014.[3] Data mining in OSNs is considered to be a very powerful marketing and advertisement tool, making several companies pay large amounts of money [171]. Naturally, this leads to increasing privacy concerns, as it is possible to generate sensitive information from any shared data, as pointed by Krishnamurthy [134, 136], such as sexual orientations based on user connections, as proved by Jernigan and Mistree [124]. In addition, Mao *et al.* [146] demonstrate in practice how to derive sensitive information from users shared tweets. Profiling is traditionally performed obliviously from users and its results may guide to discrimination by granting or denying benefits and opportunities.

Ownership and control. All content shared online is persistent and can be easily conveyed and quickly spread. It is very hard to control any information that has been publicly shared in OSNs. At the same time, ownership of the shared content in OSNs is hard to define, in particular, when the use of OSNs is paid with the users' information [171]. Therefore, once information is spread, and shared with third parties, such as target advertisement companies, the user privacy is heavily affected. In fact, users not only lose control of the shared information but also ownership, as it is hard to exercise ownership of their own information.

Identity theft. The problem of identity theft involves users pretending to be someone else by using their identity, generally for personal benefit or to cause direct damage, i.e., a personal vendetta. Popular OSNs facilitate such attacks by providing a large source of information that can be used to derive sensitive information from the shared content, for example, social security numbers as demonstrated by Gross and Acquisti [107].

Availability. Users rely on OSNs to provide an efficient communication channel for the delivery and availability of information. However, several examples demonstrate that several OSNs, such as Facebook and Twitter, have the power to censor unwanted information. They gave already executed this power, when

[3] Annual Financials for Facebook Inc. Cl A: `http://www.marketwatch.com/investing/stock/fb/financials`, Accessed: Feb. 16, 2015.

they detected sharing of harmful or disrespectful content share [6], or when they are forced by government authorities [37].

2.2.2 Privacy Paradigms in Computer Science

To address the multitude of existent privacy threats researchers have proposed several solutions to protect user data, denoted as privacy-enhancing technologies. These technologies usually involve solutions providing protection to single and specific threats, users, and systems. To categorize the different types of privacy-enhancing technologies, Gürses [111] proposed a division into three distinct privacy paradigms: practice, control, and confidentiality. Although entangled in the context of OSNs, we overview each paradigm separately and describe the general approaches followed by OSN providers.

– **Privacy as Practice** *(Identity Construction).* This category focuses mainly on the aspects of privacy as transparency and awareness. In particular, aiming at a better understanding and possible mediation during the collection, aggregation, and analysis of data by providers. Such information is required legally by DTD and FIPP regulations, and should be delivered by OSNs through their *Terms of Service*. These are generally overcomplicated (by design) so users barely read them [33], and thus along with other OSN design choices users learn very little information regarding how the data is handled [144], what are the best practices [132], or consequences [60]. Users tend to disclose more personal information using OSNs than during general offline activities, as demonstrated by Christofides *et al.* [57]. Tools such as privacy mirrors [158] allow users to set preferences and verify how those preferences impact their privacy.

– **Privacy as Control** *(Information Self-determination).* This category follows the concept of "self-determination" expressed by Westin [205], and by the ability of users to control the disclosure and dissemination of the data in order to avoid leakages [89, 38]. Hence, technologies in this group help preventing information disclosure to any entity outside the user-defined privacy policies based on access control rules, that are managed and enforced by providers. To avoid unwanted disclosure, OSNs typically allow users to define access control policies in order to select the audience that can access to the shared content, for instance, *Only me*, *Friends*, *Friends-of-Friends*, and *Public*. However, these policies are (by design) hard and difficult to use [33], confiding trust to OSNs for the management and enforcement of those policies.

– **Privacy as Confidentiality** *(Hiding).* In contrast to the previously described categories, the technologies in this category deem no trust on service providers, usually driven by possible risks of any type of information leakage [89, 204, 135], undisclosed collection [157, 161], and misuses of data [171]. Thus they aim to hide information as well as to anonymize traces, while still benefiting from the online services functionalities. This approach is currently incompatible (by choice) with the OSN business model [99], thus not offered by them as a service. Although this paradigm can be divided into different groups of confidentiality solutions, we focus on the subset addressed in this thesis, and extend each throughout the next chapters along with our contributions.

  - Data Confidentiality. Encryption techniques conceal the information, providing secrecy, unlinkability, undetectability, and unobservability. As a result, the collection of information is minimized, reducing data leakages and knowledge of the shared content by providers.

  - Communication Anonymity. (see. Definition 8) Keeps the identity of users confidential; this is achieved when a user is not identifiable among a set when performing an online action [166] and also does not reveal any additional information [75]. These properties are hard to apply to OSNs either due to economical factors, such as target advertising, or bounded by Data Retention regulations requiring the storage of communication data. Also, OSNs are identity based, thus compliant at most with pseudonymous, however, the use of persistent pseudonyms does not provide anonymity.

  - Distributed Architecture. Mostly through decentralized architectures (i.e., peer-to-peer) in order to remove the central server considered as a single point of trust. In this way, data is separated among different clients, that share and participate collaboratively towards the functionality of the system. As a result of removing the single point of trust, it requires the re-design of most current OSNs.

In light of the contributions and goals of this thesis, we mainly address problems following the privacy as control, and confidentiality paradigms; capturing both confidentiality and control together rather than as separate goal. Specifically, Chapter 5 provides a solution for privacy as control, whereas Chapters 6–8 sketch privacy as control and confidentiality solutions. However, we consider privacy as practice beyond the scope of this thesis.

2.2.3 Privacy Categories in OSNs

Multiple privacy threats occur concurrently in the complex and ever changing OSN ecosystem. This makes the process of providing solutions for the privacy problems in OSNs complex. For instance, even if access control is enforced by a user, software bugs, leaks, and the OSN economical mindset leaves users susceptible to data leakages towards other users [134] or third-party applications without user consent [171]. In fact, any user shared content, friendship lists, and actions can be exploited, and thus, negatively impact users' privacy. In this section, we extrapolate and review the different categories of privacy problems in OSNs as suggested by Gürses and Diaz [113]. Then, we review some of the general privacy threats alongside motivational examples, and survey existent privacy-enhancing solutions.

– **Institutional Privacy.** Composed of the issues related to the lack of control, and oversight over the data collection in OSNs. Such problems are addressed by legislators following DTD and FIPP regulations that are not specific for OSNs.

– **Social Privacy.** Classifies the problems leading to a direct social impact, as most of social interactions and friendship connections are mediated through OSNs. Such problems are usually linked to the lack of transparency of access control settings, hence any quick regrettable share may lead to catastrophic repercussions [202]. For instance, status updates or "controversial" picture sharing may lead to loss of employment [170], and social cyberbullying [127]. Solutions such as privacy nudges deliver enhanced privacy options for OSN users [11].

– **Surveillance.** This category considers threats related to content and behavior data in OSNs that is monitored, accessed, and processed by unwanted authorities, such as OSN providers or powerful governments agencies. All data shared using OSNs should be accessed by only the intended audience. The origin of surveillance problems is, usually, motivated by economical factors, such as the OSN business model towards target advertisement [171], or by government (mass) surveillance activities [70, 93].

In this thesis, we address predominantly the privacy problems with respect to surveillance in OSNs, by employing privacy-enhancing solutions. As a result, we mainly consider adversaries monitoring and eavesdropping the communications on OSNs with the objective to learn, process, and access the content shared on OSNs. For instance, the OSN provider motivated by economical purposes, or

pressured by governments (mass) surveillance actions. However, the problems of surveillance and social privacy are highly linked, hence the solutions proposed in the following chapters may also have a beneficial effect towards social privacy problems.

2.3 Summary

Motivated by the importance and varying definitions of privacy, this chapter sketched the current privacy definitions in computer science and translated them to context of OSNs. At the same time, we reviewed current privacy-enhancing technologies for OSNs, supporting the principles developed by Gürses and Diaz [113].

Throughout the remaining chapters we will focus on privacy solutions related to the surveillance problems on OSNs, as most of the online activities of individuals involve content data exchange and different recipient audiences. Hence, the privacy definition derived for this thesis aims at protecting user content from surveillance, so that users define the recipient audiences based on access control rules (Chapter 5) and enforce this by encryption (Chapter 6). In addition, our solutions make the communication undetectable (Chapter 7). In (Chapter 8) we present solutions for anonymous browsing. Specially, our contributions transport the offline social attitudes towards the OSN sphere with the aid of cryptographic mechanisms.

3

Literature Review

"Etudions les choses qui ne sont plus. Il est nécessaire de les connaître, ne fût-ce que pour les éviter."
– Victor Hugo, *Les Misérables (1862)*

Privacy in Online Social Networks (OSNs) has sparked a very large interest in the security and privacy the research community, resulting in a substantial amount of work aiming at protecting the privacy of the information shared and published on OSNs. In this chapter, we review different privacy-enhancing technologies proposed in the context of OSNs related to the contributions in this thesis.

3.1 Overview

Along with the increased popularity of OSNs several privacy concerns start to arise which have prompted a large interest within the research community. A number of studies have enumerated privacy issues and challenges in OSNs [40, 81] as described in Section 2.2 (Chapter 2). As a result different solutions from diverse technical research communities have been devised to mitigate privacy issues as the ones described in Chapter 2. In this chapter, we mainly review the privacy-preserving solutions that are most relevant for the proposals of Chapters 5–8, and refer the reader to other surveys for more details [2, 28, 210].

We begin by reviewing the access control solutions and models related to the collaborative access control model presented in Chapter 5. Next, we examine the solutions focusing on protecting the privacy of the content shared and published on OSNs, achieving properties such as confidentiality, integrity, and undetectability, related to Chapter 6–8. Then, we survey new OSN system designs, both centralized and decentralized. We finalize by highlighting the existent open challenges addressed by the solutions proposed throughout this thesis.

3.2 Access Control

The classical access control approach is the use of access control lists, such that each resources maps to an access list [190]. The access control design in popular OSNs is by most considered to be the bottleneck for privacy [115], mainly through the lack of user control towards a more fine grained access rights definition. Therefore, several models have been proposed to improve the current models and enhance privacy in OSNs. For example, Carminati and Ferrari [49] develop a rule based access control model using the topology of the OSN in combination with a distributed trust model. To achieve access to content, users are required to provide a proof based on the trust model. However, this solution requires full re-design of current OSNs. Squicciarini *et al.* [187] proposed PriMA a privacy system that automatically generates access rules policies according to the privacy preferences, the sensitivity, and the disclosure risk of the profile data. The policy is specified based on the disclosure risk value defined by the score of previous granted and denied accesses for each friendship relation. Although the complex access control models define policies that deliver efficient protection from unauthorized access, when applied to current OSNs they require OSN integration and access control design changes while relying on the OSN provider to enforce access control rules. These solutions do not consider content confidentiality protection against surveillance problems, as defined in Section 2.2.3.

Gürses [111] argues that in the cases that shared content is related to multiple users (e.g., Facebook tagging), access control rights should be collaboratively decided. For this purpose, Squicciarini *et al.* [188] tackled the collaborative enforcement of privacy policies using a voting system based on incentives to encourage honest behavior of the participant users. Also, Zhu *et al.* [212] suggested a new collaborative key management framework for a private OSN, focusing on distribution and delegation of keys and not on private sharing content.

3.3 Privacy Solutions

Several privacy-preserving solutions have been proposed to improve security and privacy in OSNs [2]. Most solutions address privacy problems related to the content shared and published by users on OSNs, such as profile information, status updates, comments, and images. In this section, we review the solutions based on cryptographic mechanisms aiming at protecting different types of shared content. Privacy-preserving solutions in OSNs are typically implemented on top of popular OSNs, such as Facebook, and use existing building cryptographic blocks to encrypt the content before publishing; that is analogous to Christodorescu's [56] position paper on how to privately use untrusted web servers.

Personal Details. Personal details, such as name, date of birth, and location, represent very sensitive content. Therefore, NOYB [108] proposed a technique based on substitution ciphers, used to encrypt each of the user's personal details and to encode resulting ciphertexts to look like fake, yet legitimate data. This is accomplished using uniformly distributed data from an external dictionary. Therefore, one could say that NOYB aims to (somewhat) limited undetectability in OSNs, although limited to personal details. In addition, VPSN [61] applies the concept of "virtual private networks" to OSNs, allowing users to replace the original personal details with pseudo-random information, and to establish private and confidential channels between friends to share sensitive data, similar to the concepts used in Virtual Private Networks [172].

Content Privacy. flyByNight [142] is a Facebook application designed to protect users posted content using a proxy-encryption mechanism [110], while relying on Facebook servers for key management. This scheme offers no protection against surveillance based problems as the OSN holds access to the decryption keys. In contrast, Scramble! [22] (described in Chapter 6, and Appendix A) uses broadcast encryption for improved access control on Facebook, allowing users to specify the recipients of shared information by using their public keys and group definitions similar to the concept of *circles* in Google+, thus hiding the content from the provider. Another cryptographic solution is Persona [10], that uses attribute-based cryptography [27] to limit the access rights to recipients holding specific attributes. For efficiency purposes Persona requires the recipients to be publicly known, revealing the identity of the recipients of the messages. EaSiER [123] extends the Persona attribute-based encryption approach by adding support for efficient access rights revocation based on re-encryption techniques from Naor and Pinkas [155]. Günther *et al.* [109] suggested a private profile management cryptographic model serving as

a building block for privacy in social interactions, and two private management schemes for sharing data alongside formal security definitions for confidentiality and unlinkability. The two profile management schemes are based on symmetric cryptography and broadcast encryption scheme of Gentry and Waters [97], achieving confidentiality and unlinkability. However, their construction requires users to hold full control and to manage their profile data as in decentralized networks minimizing the communication and storage overhead, resulting in an unrealistic approach for current OSNs designs. Although the previous solutions provide content privacy, achieving confidentiality by means of encryption, they all require the transfer of non-legitimate data through the OSN, and thus users risks to be easily identified and censored by a surveillance adversary.

Specifically for photo content cases, such as private photo sharing, Tootoonchian *et al.* [194] proposed Lockr to protect the privacy of shared images on OSNs by means of content separation, i.e., by storing the content in a different trusted storage server along with an access control list of the intended recipients, while uploading fake content to the OSN. Lockr assumes trust in the storage server, and the access control rights are expressed by policies based on the social relations between users. Tieney *et al.* [193] presented Cryptogram a system comprising an image encoding mechanism resilient to image compressing and transformation methods applied by OSNs. Cryptogram posts the encoded result on the OSN allowing viewing access solely to the authorized users holding the shared symmetric keys. Castiglione et al. [50] suggested using image filenames and tagging as cover objects to embed a secret in Online Photo Services, and subsequently circumvent image manipulation issues. However, this approach requires a large number of images, and an *a priori* shared knowledge of pictures.

Content Privacy and Censorship resilience. Besides the vast work addressing the privacy as confidentiality, limited work has been done to address detection and, subsequently, censorship when privately sharing information using OSNs. FaceCloak [143] and FSEO [20] have been proposed with the objective to circumvent censorship by allowing users to privately share any type of protected content (i.e., text or images) on the OSN without detection. Similarly, FaceCloak and FSEO encrypt the sensitive data, store it on third party servers, and post a short *fake* text on the OSN. Only authorized users holding decryption keys can produce the mapping index, allowing them to extract and decrypt sensitive data on the server. FaceCloak uses random sentences from Wikipedia to represent the fake text, whereas FSEO [20] allows publishers to choose their *fake* text. Despite the similarity between FaceCloak and FSEO, FaceCloak relies on a single (trusted) FaceCloak server and does not allow fine-grained access control definitions. While, FSEO (Chapter 6 and 7) extends Scramble! to allow a more fine-grained access control while keeping the recipients

identities secret to outsiders by using anonymous broadcast encryption [16, 140] and content storage to user-defined servers. Moreover, FaceCloak [143] fake text does not achieve undetectability (as defined in Chapter 7), and rather circumvents restrictions on the use of encryption on OSNs.

Friend match privacy. For protecting content privacy in the friend search and common friend finder scenarios, De Cristofaro *et al.* [72] introduced a private contact discovery protocol. The protocol enables two users of an OSN to learn their common contacts without learning any of the other friends. Later, Nagy *et al.* [154] extended [72] to the finding friends problem, using private set intersection techniques. The protocol allows users to privately generate and share their list of friends such that other friends can compare and find common friends in the honest-but-curious model.

3.4 New OSN designs

As a result of a privacy-unfriendly OSN business model [171], the faulty OSN access control design [115], and the concerns on the single point of trust [30], several privacy-friendly architectures have been suggested to replace existing platforms. Although decentralized architectures are often advocated, there exist centralized privacy-friendly solutions that rely on the server solely to support high-availability of content dissemination and simple storage to a large number of non-tech-savvy users. For example, Anderson *et al.* [5] propose a centralized system that stores all user generated content in encrypted format, such as content and friends list. Only the authorized users holding the associated cryptographic keys are able to access the content. Similarly, Feldman *et al.* [86] proposes the Frientegrity framework for protecting privacy and integrity of the information shared, while making it possible to detect misbehavior and benefiting from the availability and reliability benefits of a centralized environment. In addition, Hummingbird [73] presents a variant of Twitter that provably guarantees confidentiality of tweet contents, hashtags, and follower interests. Hummingbird bases its design on private set intersection [71] methods to match the authorized followers to the private tweets.

In turn, decentralized solutions, such as Safebook [63] and Diaspora [82] exclude central servers to eavesdrop, manage storage, and maintain communications, removing the single point of failure. In this way, users are able to keep their data, friendship connections, and actions performed on the OSN away from the prying eyes of the OSN provider. However, decentralized solutions do not deliver directly security and privacy towards malicious peers storing and processing data, striking problems with the metadata [106]. For this reason, Backes *et al.* [8]

develop a cryptographic API achieving improved access control, anonymity, and confidentiality. Vu *et al.* [201] mitigate the issue of data backup in decentralized OSNs by using secret sharing. In turn, decentralized architectures may hinder *real-time* availability of information or require users to buy cloud storage for their data [179].

In the context of OSNs, the decision of switching to either a different or a new privacy-friendly OSN is collective by nature, thus, creating obstacles to large-scale diffusion of new infrastructures. In other words, users may not be motivated to switch, unless the majority of their friends will switch as well.

3.5 Summary

A substantial research effort has been made by different research disciplines focusing on diverse aspects of security and privacy problems on OSNs, but most require changes to the OSN design and infrastructure. Throughout this thesis we mainly focus on building privacy-enhancing solutions that can be plugged on top of popular OSNs used today by hundreds of millions of people. In addition, most proposals for enforcing access definitions do not provide content confidentiality required for protection against mass surveillance issues, while the majority of content privacy solutions do not allow a fine-grained access control definitions per item and do not protect the identity of the recipients. These problems are addressed in Chapter 5 and Chapter 6. Moreover, very little work has been done on undetectability and anonymous browsing in OSNs. To the best of our knowledge, we present the first study of undetectable communication in OSNs in Chapter 7, and the the first system providing anonymous browsing in Chapter 8.

4

Background and Notation

"I do not define time, space, place, and motion, as being well known to all."
– Isaac Newton, *Philosophiae Naturalis Principia Mathematica (1687)*

This chapter introduces the relevant background and notation for understanding the remaining chapters throughout this thesis. After presenting a general definition of Online Social Networks (OSN), it focuses on relevant computational assumptions, concepts, and notions from the fields of cryptography and privacy.

4.1 Online Social Networks

We model an Online Social Network (OSN) as the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ whose vertices represent the users $u \in \mathcal{U}$ in the OSN (i.e., $\mathcal{V}$), and the edges the connections between users. Each u establishes a (a-)symmetric set of relationships $\mathcal{R}_u \in \mathcal{U}$ that contains all users to which u is connected with, such that, $(u, v) \in \mathcal{E}$ represents a valid connection, if $v \in \mathcal{R}_u$, and $u \in \mathcal{R}_v$ for undirected graphs. Henceforth, users are considered to hold a profile $\mathcal{P}$ in the OSN, and manage a list of friendship connections $\mathcal{R}$ with whom they share content $\mathtt{m}$.

Without loss of generality, throughout this thesis we consider two users of any OSN – u and v as Alice and Bob, respectively; they adhere to the following roles:

Publisher: user who publishes information $\mathtt{m}$ to a set of recipients $\mathcal{S}$, generally denoted by u or simply Alice.

Viewer/Recipient: user who accesses, and views the posted information $\mathtt{m}$, denoted as v or Bob, and commonly $v \in \mathcal{S} \vee v \in \mathcal{R}_u$.

For simplicity's sake, we consider Alice the user who initializes the communication with the recipient Bob. They exchange or post arbitrary size messages $\mathtt{m} \in \mathcal{M}$, such that, $\mathcal{M} \in \{0,1\}^*$. The size, however, may be limited depending on the OSN provider, for instance, Twitter applies a 140-character limitation (1120 *bits*).

4.2 Cryptography

In this section we briefly overview the cryptographic concepts, tools, and building blocks. For ease of explanation we omit the definitions and basic notions of the underlying cryptographic primitives, such as hash functions, number-theoretic assumptions as well as encryption, and signature schemes.[1]

4.2.1 General Notation and Definitions

For any $n \in \mathbb{N}$, let $\{0,1\}^n$ denote the set of bit strings of length n, and $\{0,1\}^*$ the set of bit strings of arbitrary length. For two strings x and y, $x \parallel y$ denotes their concatenation, and $x \oplus y$ their bitwise XOR. The notation $x \xleftarrow{r} X$ indicates that x is selected uniformly at random from the finite set X. For any two sets X and Y, we define the union by $X \cup Y = \{x : x \in X \vee x \in Y\}$, the intersection as $X \cap Y = \{x : x \in X \wedge x \in Y\}$, the symmetric difference as $X \triangle Y = \{x : (x \in X) \oplus (x \in Y)\}$, the subset of Y as $X \subset Y = \{y \in Y : y \in X\}$, and the empty set by $X = \emptyset$.

The security parameter λ is considered to be l bits, and $\mathcal{M} \in \{0,1\}^*$ the message space, such that the plaintext message $\mathtt{m} \in \mathcal{M}$.

[1] For extra details on basic notions of cryptography we point the reader to the Handbook of Applied Cryptography [148].

Negligible Function. A function $f(\cdot)$ is negligible in the security parameter λ if for any polynomial $p(\cdot)$, the function $f(\cdot)$ is bounded from above by $p(\lambda)^{-1}$. For simplicity reasons, we denote a negligible function $\epsilon(\lambda)$ by ϵ.

Public-Key Encryption. A public-key encryption scheme is composed of three algorithms: $\mathtt{PKE} = \{\mathtt{KeyGen}, \mathtt{Enc}, \mathtt{Dec}\}$. The $\mathtt{KeyGen}(\lambda)$ returns the public-private key-pair (pk,sk) on input of the security parameter λ. While $\mathtt{Enc}_{pk}(\mathtt{m})$ takes a message $\mathtt{m}$and the public key pk, and outputs the ciphertext $\mathtt{C}$. The $\mathtt{Dec}_{sk}(\mathtt{C})$ outputs $\mathtt{m}$ for the correct private key sk and $\mathtt{C}$, otherwise $\perp$.

Symmetric-key Encryption. A symmetric-key encryption is a function $\mathsf{E} : \mathcal{K} \times \mathcal{M} \rightarrow \mathcal{C}$, such that, for each secret key $\mathtt{k} \in \mathcal{K}$, and a message $\mathtt{m} \in \mathcal{M}$, $\mathsf{E}_{\mathtt{k}}(\mathtt{m})$ represents the encryption (invertible mapping) of $\mathtt{m}$ under $\mathtt{k}$. The invertible mapping is the decryption function denoted by $\mathsf{D}_{\mathtt{k}}(\mathtt{C})$. The authenticated version returns an extra authentication tag $\mathtt{T}$, with $\langle \mathtt{C}, \mathtt{T} \rangle \leftarrow \mathsf{E}_{\mathtt{k}}(\mathtt{m})$, such that, $\mathsf{D}_{\mathtt{k}}(\mathtt{C}, \mathtt{T})$ will output $\mathtt{m}$. Any tampering of $\mathtt{C}$ will result in a different tag $\mathtt{T}'$, such that $\mathtt{T}' \neq \mathtt{T}$, and $\mathsf{D}_{\mathtt{k}}(\mathtt{C}, \mathtt{T})$. Throughout this thesis we interchangeably use the same notation $(\mathsf{E}(\cdot), \mathsf{D}(\cdot))$ for semantically secure symmetric-key encryption, and authenticated symmetric encryption schemes.

Digital Signatures. A public-key digital signature scheme is composed of three algorithms: $\mathtt{DSig} = \{\mathtt{KeyGen}, \mathtt{Sign}, \mathtt{Ver}\}$, representing the key generation, signing, and verification algorithms. The $\mathtt{KeyGen}(\lambda)$, on input of the security parameter λ, outputs a public-private key-pair (vk,sgk). The algorithm $\mathtt{Sign}_{sgk}(\mathtt{m})$, takes a message $\mathtt{m}$ and the private signing key sgk, and outputs the signature σ. $\mathtt{Ver}_{vk}(\sigma, \mathtt{m})$ returns true if the signature σ on the message $\mathtt{m}$ using the public verification key vk is valid, otherwise false.

Hash Functions. A cryptographic hash function $\mathtt{H}(\cdot)$ is an algorithm that maps arbitrarily long bit strings to digests of a fixed length l, s.t., $\mathtt{H} : \{0,1\}^* \rightarrow \{0,1\}^l$. It is assumed that $\mathtt{H}$ is modeled as a random oracle.

Message Authentication Codes. A message authentication code ($\mathtt{MAC}$) is a keyed hash function, taking the secret key $\mathtt{k}$ and an arbitrary-length as input. The output is fixed-length and provides integrity and authenticity of message $\mathtt{m}$ under key $\mathtt{k}$, $\mathtt{MAC}_{\mathtt{k}}(\mathtt{m})$.

Pseudo-Random Functions. A pseudo-random function ($\mathtt{PRF}$) is an efficient (deterministic) algorithm which given a key $\mathtt{k}$ and an n-bit string $x \leftarrow \{0,1\}^n$, returns an n-bit string $y \leftarrow \mathtt{PRF}_{\mathtt{k}}(x)$, so that it is infeasible to distinguish y from

a truly random output. In short, $\mathtt{PRF} : \mathcal{K} \times \{0,1\}^n \rightarrow \{0,1\}^n$. We consider for the remainder of this thesis, message authentication codes $\mathtt{MAC_k}(\cdot)$ to constructed by $\mathtt{PRF_k}(\cdot)$, although the inverse is not necessarily true.

Bilinear Maps. Let $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ be three groups of prime order q. An admissible *asymmetric*[2] bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ is defined as a map from the gap groups $\mathbb{G}_1$ and $\mathbb{G}_2$ to the target group $\mathbb{G}_T$ that satisfies the following properties:

Bilinearity. For all values of $a, b \in \mathbb{Z}_q$, and for all $P \in \mathbb{G}_1$ and $Q \in \mathbb{G}_2$, $e(aP, bQ) = e(P, Q)^{ab}$.

Non-degeneracy. If P is a generator of $\mathbb{G}_1$, and Q is a generator of $\mathbb{G}_2$, $e(P, Q)$ is a generator of $\mathbb{G}_T$.

Computability. There is an efficient algorithm to compute $e(P, Q)$ for all $P \in \mathbb{G}_1$ and $Q \in \mathbb{G}_2$.

Throughout this thesis we only use asymmetric pairings, for efficiency, and security reasons, according to Galbraith [95] and Joux in [126].

4.2.2 Security Definitions

Information-theoretic arguments cannot be used to prove security of most cryptographic primitives. However, security can be bounded to the computational complexity of certain mathematical tasks, for which it is assume that no algorithm can solve this task in polynomial time with negligible probability. Hence, after introducing the general security definitions we describe some mathematical hardness assumptions known from the literature that are used in this thesis, as follows.

Semantic Security. Without loss of generality, throughout this thesis, we consider that a cryptographic scheme is semantically secure if the indistinguishability property holds for the message and ciphertext.

Definition 1 (Message Indistinguishability)**.** *The cryptographic scheme $T(\cdot)$, such that $\mathtt{C} \leftarrow T(\lambda, \mathtt{m})$, is message indistinguishable if for every two messages $\mathtt{m}$, $\mathtt{m}'$, any bounded adversary $\mathcal{A}$ cannot distinguish the output of $T(\mathtt{m})$ of $T(\mathtt{m}')$, with non-negligible probability.*

$$|\mathtt{Pr}[\mathcal{A}(\mathtt{C} \leftarrow T(\lambda, \mathtt{m})) = 1] - \mathtt{Pr}[\mathcal{A}(\mathtt{C}' \leftarrow T(\lambda, \mathtt{m}')) = 1]| \leq \epsilon.$$

[2]For the *symmetric* bilinear map, the mapping groups are of the same order, such that: $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$.

Definition 2 (Ciphertext Indistinguishability)**.** *The cryptographic scheme $T(\cdot)$, such that $\mathtt{C} \leftarrow T(\lambda)$, is ciphertext indistinguishable if no bounded adversary $\mathcal{A}$ is able to distinguish the output of T from any random value $r \leftarrow \{0,1\}^{\lambda}$, with non-negligible probability.*

$$|\mathtt{Pr}[\mathcal{A}(\mathtt{C} \leftarrow T(\lambda)) = 1] - \mathtt{Pr}[\mathcal{A}(r \xleftarrow{r} \{0,1\}^{\lambda}) = 1]| \leq \epsilon.$$

Key-Privacy. Public key anonymity, or Key-Privacy was defined by Bellare *et al.* [25] as the indistinguishability property of the public keys used for encryption. In particular, the hardness of identifying the public key used for the encryption, defined as follows.

Definition 3 (Key-Privacy)**.** *A public key encryption scheme $\mathtt{C} \leftarrow \mathtt{Enc}_{pk}(\mathtt{m})$, is key private if any bounded adversary $\mathcal{A}$, with access to the list $\{pk_0, pk_1\}$, is not able to distinguish the output $\mathtt{C}_b$ of $\mathtt{Enc}_{pk_b}(\mathtt{m})$ when using pk_0 and pk_1, s.t., $b = \{0,1\}$, with non-negligible probability.*

$$|\mathtt{Pr}[\mathcal{A}(pk_0, pk_1, \mathtt{m}, \mathtt{C}_0) = 1] - \mathtt{Pr}[\mathcal{A}(pk_0, pk_1, \mathtt{m}, \mathtt{C}_1) = 1]| \leq \epsilon.$$

Hardness Assumptions. Let $\langle \mathbb{G}, q, g \rangle \leftarrow \mathcal{G}(\lambda)$ denote the setup algorithm that generates a group $\mathbb{G}$ of order q, on input of the security parameter λ.

Definition 4 (Discrete Logarithm Problem (DLP))**.** *For $\langle \mathbb{G}, q, g \rangle \leftarrow \mathcal{G}(\lambda)$ and any random $x \xleftarrow{r} \mathbb{Z}_q$, the DLP problem states that is hard to find a value x, given g^x.*

$$\mathtt{Pr}[\mathcal{A}(g, g^x) = x \ \ | \langle \mathbb{G}, q, g \rangle \leftarrow \mathcal{G}(\lambda)] \leq \epsilon.$$

Definition 5 (Computational Diffie-Hellman Problem (CDH))**.** *Let $\langle \mathbb{G}, q, g \rangle \leftarrow \mathcal{G}(\lambda)$. For any random values $x, y \xleftarrow{r} \mathbb{Z}_q$, the CDH problem denotes the hardness for the adversary $\mathcal{A}$ to compute g^{xy} with non-negligible probability.*

$$\mathtt{Pr}[\mathcal{A}(g, g^x, g^y) = g^{xy} \ \ | \langle \mathbb{G}, q, g \rangle \leftarrow \mathcal{G}(\lambda)] \leq \epsilon.$$

Definition 6 (Decision Diffie-Hellman Problem (DDH))**.** *For $\langle \mathbb{G}, q, g \rangle \leftarrow \mathcal{G}(\lambda)$, let $x, y, z \xleftarrow{r} \mathbb{Z}_q$, DDH problem states the hardness for the adversary $\mathcal{A}$ to distinguish (g, g^x, g^y, g^z) from (g, g^x, g^y, g^{xy}).*

$$|\ \mathtt{Pr}[\mathcal{A}(g, g^x, g^y, g^z) = 1] - \mathtt{Pr}[\mathcal{A}(g, g^x, g^y, g^{xy}) = 1]\ | \leq \epsilon.$$

Definition 7 (Bilinear Diffie-Hellman Problem (BDH))**.** *Let $\mathbb{G}_1$, $\mathbb{G}_2$ be two groups of prime order p, q, and $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ an admissible bilinear map, such that P and Q are generators of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively. The BDH problem in $\langle \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e \rangle$ is as follows: given $\langle P, Q, aP, bQ, cPQ \rangle$ for some randomly chosen $a, b, c \in \mathbb{Z}_q$ compute $W = e(P, Q)^{abc} \in \mathbb{G}_T$.*

An algorithm $\mathcal{A}$ has negligible advantage ϵ in solving BDH in $\langle \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e \rangle$ if:

$$\mathtt{Pr}[\mathcal{A}(P, Q, aP, bQ, cPQ) = e(P, Q)^{abc} \mid \langle q, \mathbb{G}_1, \mathbb{G}_2, e \rangle \leftarrow \mathcal{G}(\lambda)] \leq \epsilon.$$

where the probability is over the random choice of $q, \mathbb{G}_1, \mathbb{G}_2, e$ according to the distribution induced by $\mathcal{G}(\lambda)$, the random choice of $a, b \in \mathbb{Z}_q$, and the random bits of the algorithm $\mathcal{A}$.

Random Oracles. A random oracle is a theoretical black box that for each unique input returns a uniformly random chosen result from its output domain. A random oracle is deterministic, i.e., given a particular input it will always produce the same output.

4.2.3 Building Blocks

Now we turn to overview the main cryptographic building blocks used in the remainder of this thesis.

Identity-Based Encryption. The concept of Identity-Based Encryption (IBE) was introduced by Shamir [181], with the main idea of using any string as the public key. IBE requires no certificates as users can rely on publicly known identifiers such as an e-mail address or a telephone number, thus, reducing the complexity of establishing and managing a public key infrastructure. Boneh and Franklin propose the first practical IBE using bilinear pairings [31], and modeled in in the random oracle model. Later Gentry [96] presented an extension to the standard model.

A generic IBE scheme is composed of four randomized algorithms.

`IBE.Setup:` On the input of a security parameter λ, outputs a master secret msk, and the master public parameters $mpk = params$.

`IBE.Extract:` Takes the public parameters $params$, the master secret msk plus an `id`, and returns the private key $sk_{\mathtt{id}}$.

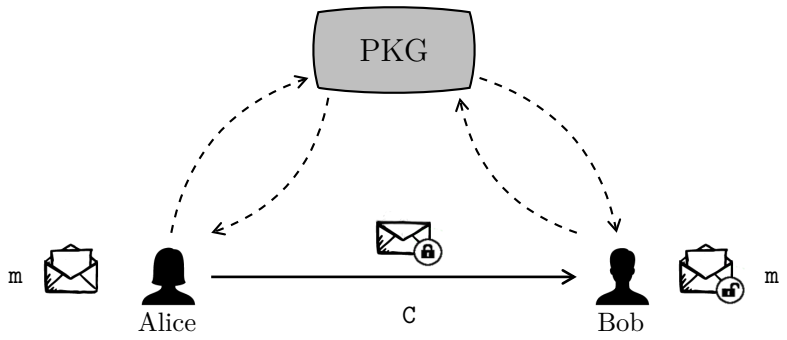


Figure 4.1: Identity-Based Encryption Model, where Alice sends an encrypted message to Bob using a valid string as his public key, such as Bob's email address.

`IBE.Encrypt:` Returns the encryption $\mathtt{C}$ of the message $\mathtt{m}$ on the input of the *params*, the $\mathtt{id}$, and the arbitrarily length message $\mathtt{m}$.
`IBE.Decrypt:` Reconstruct $\mathtt{m}$ from $\mathtt{C}$ by using the secret $sk_{\mathtt{id}}$ and the *params*.

Boneh and Franklin Identity-Based Encryption scheme [31]

IBE.Setup(λ)

- Let $\mathcal{G}(\lambda)$ be a BDH generator, output a prime q, two groups $\mathbb{G}_1$, $\mathbb{G}_2$, the bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$, and the message space $\mathcal{M} = \{0,1\}^n$.
- $s \xleftarrow{r} \mathbb{Z}_q$ and set $P_{pub} = sP$, such that, $P \in \mathbb{G}_1$.
- The hash functions $\mathtt{H}_1 : \{0,1\}^n \to \mathbb{G}_1$, $\mathtt{H}_2 : \mathbb{G}_2 \to \{0,1\}^n$ modeled as random oracles.

$$mpk = \langle q, \mathbb{G}_1, \mathbb{G}_2, e, n, P, P_{pub}, \mathtt{H}_1, \mathtt{H}_2 \rangle \qquad msk = s$$

IBE.Extract($\mathtt{id}$)	IBE.Encrypt($\mathtt{m}, \mathtt{id}$)	IBE.Decrypt($\mathtt{C}, sk_{\mathtt{id}}$)
$Q_{\mathtt{id}} = \mathtt{H}_1(\mathtt{id}) \in \mathbb{G}_1$	$Q_{\mathtt{id}} = \mathtt{H}_1(\mathtt{id}) \in \mathbb{G}_1$	$\mathtt{C} = \langle U, V \rangle$
$sk_{\mathtt{id}} = sQ_{\mathtt{id}}$	$r \xleftarrow{r} \mathbb{Z}_q$	
$pk = \mathtt{id}$	$x = e(Q_{\mathtt{id}}, P_{pub})) \in \mathbb{G}_2$	$X = \mathtt{H}_2(e(sk_{\mathtt{id}}, U)) \in \mathbb{G}_2$
return $(pk, sk_{\mathtt{id}})$	$\mathtt{C} = \langle rP,\ \mathtt{m} \oplus \mathtt{H}_2(x^r) \rangle$	$\mathtt{m} = V \oplus X$

Figure 4.2: Boneh and Franklin basic IBE scheme [31].

The `IBE.Setup` and `IBE.Extract` algorithms are executed by a trusted Private Key Generator (PKG) server, whereas `IBE.Encrypt` and `IBE.Decrypt` are performed by two players, e.g., Alice and Bob. Consequently, key escrow is performed implicitly in the classic IBE scheme as the PKG holds the master

secret key. Figure 4.1 depicts a general overview of the IBE model, whereas Figure 4.2 describes the Boneh and Franklin basic IBE scheme.

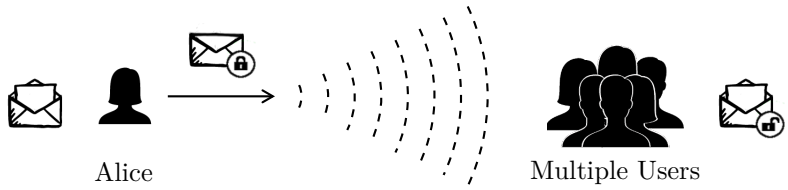


Figure 4.3: Broadcast Encryption Model, where Alice sends an encrypted message to multiple recipients.

Anonymous Broadcast Encryption. Broadcast encryption (BE) schemes were introduced by Fiat and Naor [88], to address the multi-user setting, as depicted in Figure 4.3. A BE scheme allows a user to encrypt a message to a set $\mathcal{S}$ of users, such that only the users in the set $\mathcal{S}$ are able to decrypt the message. The computational overhead of the BE is typically related to the length of the ciphertext and the number of recipients. To overcome this issue, the set $\mathcal{S}$ of recipients is generally known. Barth *et al.* [16] and Libert *et al.* [140] extended the notion of BE, and introduced the notion of Anonymous Broadcast Encryption (ANOBE) scheme, where the recipient set $\mathcal{S}$ remains private even to the members in the set. Later, Fazio and Perera [85] suggested the notion of outsider anonymous BE (oANOBE) representing a more relaxed notion of ANOBE that provides set privacy only to outside users.

A generic BE and ANOBE scheme consists of four randomized algorithms.

`BE.Setup:` On the input of a security parameter λ, generates the public parameters *params* of the system.
`BE.KeyGen:` Returns the public and private key (pk,sk) for each user according to the *params*.
`BE.Encrypt:` Takes the set of users composed by their pk $\mathcal{S} = \{pk_0 \dots pk_{|\mathcal{S}|}\}$ along with the secret message `m` and generates `C`.
`BE.Decrypt:` Reconstructs `m` from C using the private key sk_i if the corresponding public key $pk \in \mathcal{S}$. Otherwise, it returns $\perp$.

Figure 4.4 displays the Barth *et al.* [16] general anonymous BE scheme, secure under the random orable model. Note that the pk can be represented by the `id` value from the IBE scheme.

Barth, Boneh, and Waters Anonymous Broadcast Encryption scheme [16]

BE.Setup(λ)

- Let $\mathbb{G}$ be a group, with generator g, where CDH is hard, DDH easy, and the hash function $\mathtt{H} : \mathbb{G} \to \{0,1\}^\lambda$ modeled as a random oracle.
- Strongly correct, key private $\mathtt{PKE}$=$\{\mathtt{KeyGen}(\lambda), \mathtt{Enc}_{pk}(\cdot), \mathtt{Dec}_{sk}(\cdot)\}$ scheme.
- Strongly unforgeable $\mathtt{DSig} = \{\mathtt{KeyGen}(\lambda), \mathtt{Sign}_{sgk}(\cdot), \mathtt{Ver}_{vk}(\cdot)\}$ scheme.
- Semantically secure $\mathsf{E}(\cdot), \mathsf{D}(\cdot)$.

BE.KeyGen(λ)

$(pk, sk) \leftarrow \mathtt{PKE.KeyGen}(\lambda)$

$a \xleftarrow{r} \{0,1\}^\lambda$

$pk' = (pk, g^a)$
$sk' = (sk, a)$

return (pk', sk')

BE.Encrypt($\mathtt{m}, \mathcal{S}$)

$(sgk, vk) \leftarrow \mathtt{DSig.KeyGen}(\lambda)$
$\mathsf{k} \xleftarrow{r} \{0,1\}^\lambda$, $r \xleftarrow{r} \{0,1\}^\lambda$
$T = g^r$
For all $pk' \in \mathcal{S}$:
| $X = (vk \parallel g^{a \cdot r} \parallel \mathsf{k})$
| $c_{pk} = \mathtt{H}(g^{a \cdot r}) \parallel \mathtt{Enc}_{pk}(X)$

$C_1 = \{c_{pk_0} \parallel \ldots \parallel c_{pk_{|\mathcal{S}|}}\}$
$C_2 = \mathsf{E}_\mathsf{k}(\mathtt{m})$
$\sigma = \mathtt{Sign}_{sgk}(T \parallel C_1 \parallel C_2)$

$\mathtt{C} = \langle \sigma \parallel T \parallel C_1 \parallel C_2 \rangle$

BE.Decrypt($\mathtt{C}, sk$)

$\mathtt{C} = \langle \sigma \parallel T \parallel C_1 \parallel C_2 \rangle$
$Y = \mathtt{H}(T^a) = \mathtt{H}(g^{a \cdot r})$
Find $c_j = Y \parallel c_{pk}$
| otherwise, return $\perp$
$P \leftarrow \mathtt{Dec}_{sk}(c_{pk})$
| If $P = \perp$, return $\perp$
| otherwise, $P = \langle vk, x, \mathsf{k} \rangle$

If $x \neq T^a$, return $\perp$
If $\mathtt{Ver}_{vk}(\sigma \parallel T \parallel C_1 \parallel C_2)$
| $\mathtt{m} = \mathsf{D}_\mathsf{k}(C_2)$
| otherwise $\perp$

Figure 4.4: Barth, Boneh and Waters Anonymous Broadcast Encryption scheme [16].

Secret Sharing. The notion of secret sharing was introduced by Shamir [180] with the objective of dividing a secret k into n shares among n entities, such that, only a subset of size equal to or greater than a threshold t can reconstruct k, where $t \leq n$. In practice, a random secret k is generated along with a polynomial over $\mathbb{Z}_p$ of prime order p, $f(x)$ of degree $t-1$, such that $p > max(\mathsf{k}, n)$ and $f(0) = \mathsf{k}$, where the shares $s_i = f(i) \mod p$, are represented by different points on the polynomial. Any entity with t or more shares can reconstruct $f(x)$ using Lagrange interpolation, and subsequently find k. This is done by constructing the Lagrange multipliers a_i in t points of $f(x)$, as follows.

$$\mathsf{k} = \sum a_i s_i \qquad \text{for} \qquad a_i = \prod_{j \neq i} \frac{j}{j-i}.$$

Figure 4.5 exemplifies graphically the generic model of secret sharing. Further, Chor *et al.* [55] suggested a Verifiable Secret Sharing (VSS) scheme to allow anyone to verify that the right shares are used. The scheme was extended by Feldman [87] and later by Pedersen [163].

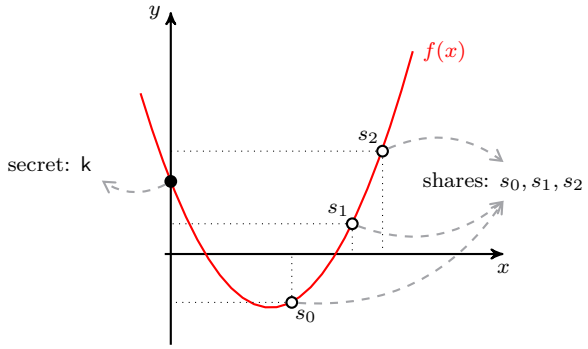


Figure 4.5: Secret Sharing Model representation whereby the polynomial for degree $t = 2$, so that knowledge of s_0, s_1, and s_2 allows computation of the main secret k.

Distributed Key Generation. Distributed Key Generation (DKG) was introduced by Pedersen [163, 164] to allow a group of entities to collaboratively setting a secret sharing environment over a public channel.

For multiple parties to jointly generate a secret sharing k, all entities are required to participate in a DKG scheme. Each entity i involved generates a different k_i and $f^i(x)$. Later the shares s_{ij} are distributed and verified. Hence, a generic DKG does not require a trusted party, since the master secret is computed as the sum of all the polynomials, and can only be retrieved by joining t shares. A generic DKG protocol consists of two phases:

`DKG.Setup:` Every entity i generates a random secret k_i, and computes a polynomial of degree $t-1$. The entity i distributes a valid share s_j^i over all the other j entities, along with the commitment to the share. Each entity j verifies the shares, and computes the new share $s_j = \sum_i s_j^i$. The master secret is unknown by each party, and composed of the origin point on the sum of all polynomials $f^i(x)$.

`DKG.Reconstruct:` Each entity i broadcasts its share s_i, and with $t \leq n$ shares, one can reconstruct the master secret k.

The DKG protocol is secure assuming that no adversary is able to corrupt t parties or more. Figure 4.7 summarizes the Pedersen DKG scheme [164], while Figure 4.6 demonstrates a simple DKG scheme among three different players with the associated polynomials $f(x), g(x), h(x)$, respectively. Although, all parties hold the share s_{ij} of the final polynomial, represented by the aggregation of the polynomials of all the parties, the master secret is kept oblivious to any party with less than t shares.

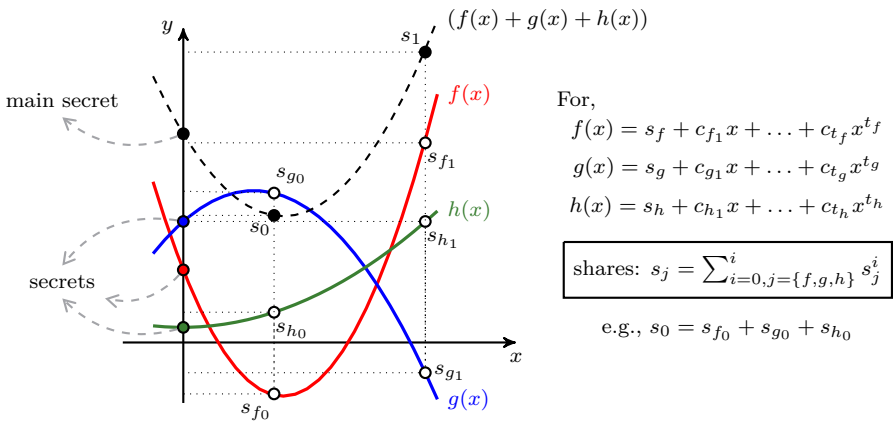


Figure 4.6: Distributed Key Generation Model with three parties.

Pedersen Distributed Key Generation scheme [164]

DKG.Setup(i, t, n)

Each player i :
- Pick a polynomial $f^i(x) = c_0 + c_1 x + \ldots + c_t x^t$, s.t., $c_0 = \mathsf{k}_i, c_1, \ldots, c_t \in \mathbb{Z}_q$.
- Compute n shares $s_j^i \leftarrow f^i(x)$
- Compute t-commitments $\alpha_v^i = g^{c_v} (\text{mod } \mathsf{q})$ for $v = 0, \ldots, t$.
- Broadcast: $(s_j^i, \alpha_0^i, \alpha_1^i, \ldots, \alpha_t^i)$.

DKG.Reconstruct$(j, s_j^{i=\{0,\ldots,n\}})$

Each player j :
- Verify shares s_j^i,

$$g^{c_i} \stackrel{?}{=} \prod_{j=0}^{t} \alpha_j^i (\text{mod } \mathsf{q})$$

- Compute main share s_j,

$$s_j = \sum s_j^i$$

Figure 4.7: Pedersen VSS Distributed Key Generation scheme [164], allowing share verification with the commitments α^i.

4.3 Privacy

Although encryption provides confidentiality of the content shared, it does not directly provide anonymity and thus privacy. Traffic analysis presents a

powerful instrument to identify the user communication without knowledge of the content. However, providing a definition of privacy is a challenging task. In order to standardize privacy as anonymity, Pfitzmann and Köhntopp [166] defined anonymity as follows.

> *"Anonymity is the state of being not identifiable within a set of subject, the* anonymity set *".*

In other words, the subject u_i is anonymous if it is hard to identify u_i in the set $\mathcal{U} = \{u_i, \ldots, u_N\}$, such that, $N = |\mathcal{U}|$. Consequently, in the effort to quantify anonymity in communications Serjantov and Danezis [178], and Diaz *et al.* [76] both propose the use of entropy as a valid information-theoretic anonymity metric. Whereas the first method allows to measure the effective size of the anonymity set, the latter enables to obtain a classification of the anonymity degree within an interval from 0 to 1. Thus, let $H(x)$ be the entropy of a random variable x, such that, $p_i = \mathtt{Pr}[x = i]$ for the anonymity set $\mathcal{U}$, then the effective size of the anonymity set $H(x)$, and the degree d of the anonymity are calculated, respectively, as:

Definition 8 (Anonymity Size [178])**.** *Given an anonymity set* $\mathcal{U} = \{u_i, \ldots, u_N\}$*, s.t.,* $N = |\mathcal{U}|$*,* $u_i \in \mathcal{U}$*, and considering* $p_i = \mathtt{Pr}[x = i]$ *as the probability distribution of* u_i *among all users in* $\mathcal{S}$*. Then, the effective anonymity size is computed as follows:*

$$H(x) = -\sum_{i=0}^{i=N} p_i \cdot \log_2 p_i.$$

Definition 9 (Degree of Anonymity [76])**.** *Given the anonymity set for the set* $\mathcal{S}$*. The degree* d *of anonymity of* $\mathcal{S}$ *is calculated as follows:*

$$d = \frac{H(x)}{H_{max}} \qquad for \qquad H_{max} = log_2 N.$$

Unlinkability. Within a system, a content is unlinkable if an adversary is unable to map two or more pieces of information to a single user. In short, if Alice publishes $\mathtt{m}_1$ and $\mathtt{m}_2$, and later $\mathtt{m}_3$, an adversary with a-priori knowledge of $\mathtt{m}_1$ and $\mathtt{m}_2$ is unable to link $\mathtt{m}_3$ to Alice.

Unobservability. Similarly to indistinguishability in cryptography, indicates the state whereby the action performed by a user is indistinguishable from any other action of the same type from the same or other user. Thus, it is hard to distinguish the action from any other random action. Often unobservability implies anonymity [166].

Pseudonymity. Represents the state where users replace their identities by a false random identity (pseudonym). Although used for protecting real identities, the use of single pseudonyms leads to single unique identifiers. Therefore, anonymity systems generally apply short lived random pseudonyms per action to achieve unlinkability by pseudonymity.

Part II

Private Information Sharing in Online Communities

5

Audience Segregation

"Diviser chacune des difficultés que j'examinerais, en autant de parcelles qu'il se pourrait, et qu'il serait requis pour les mieux résoudre."

– René Descartes, *Le Discours de la Méthode (1637)*

Audience segregation occurs naturally in current society, with people adapting their behavior according to the surroundings, and audiences. The natural offline behavior can be directly correlated to the actions and access control definitions of Online Social Networks (OSNs), so that access control rights of shared content entirely depend on the users, and can be adapted to fit specific contexts. In this chapter, we propose general concepts describing access control for OSNs. After modeling the group and collaborative access control, we devise a collaborative scheme based on secret sharing. Furthermore, we demonstrate its applicability, and discuss the implementation challenges.

Publications.

[21] Beato, F., Kohlweiss, M., and Wouters, K., Enforcing Access Control in Social Networks. In *HotPets 2009* (Jul. 2009).

[24] Beato, F., and Peeters, R., Collaborative Joint Content Sharing for Online Social Networks. In *IEEE SESOC 2014* (Mar. 2014).

Contributions. Principal author together with Roel Peeters.

Chapter Outline. This chapter describes mainly the work published in [21] and [24]. After formalizing audience segregation and collaborative access control on OSNs, we devise a collaborative scheme taking into account the limitations of modern OSNs. Finally, we analyze, and evaluate the scheme along with the implementation challenges.

5.1 Motivation

The concept of audience segregation was introduced in 1959 by Goffman [100] and describes the phenomenon of people performing different roles for different audiences, and within different contexts, mainly as a form of protection of sensitive information, and aiming at providing favorable images. Although initially defined as an off-line world phenomenon, it arguably corresponds to the current digital society where Online Social Networks (OSNs) represent the communication ecosystem [197]. Users map their off-line actions to their OSN behavior, applying different roles within different OSN groups. This makes audience segregation, and data handling important privacy issues [3]. Popular OSNs empower users with some customizable *"privacy settings"* to take on access rights decisions based on access groups, and thus, limiting access rights to a subset of the users, for instance, *Friends*, *Friends-of-Friends*, and everyone. However, those mechanisms are often difficult and coarse, leading to accidental leakages [32, 204] as discussed in Chapter 2. Gürses and Berendt [112] support this fact, by arguing that the current access control design on OSNs represents the principal bottleneck of privacy.

In addition, a vast majority of information shared on OSNs relates to more than one user. For instance, picture tagging on Facebook, where apart from the publisher there are other users in the picture. The most commonly used approach by OSNs is to provide access rights to the union of all related users, lacking collaboration, and thus leading to possible unwanted leakages. In fact, Facebook's Data Use Policy[1] stated the following.

> *"If you tag someone, that person and their friends can see your story no matter what audience you selected. The same is true when you approve a tag someone else adds to your story."*

Then, if Alice shares a picture, tagging Bob and Charlie, the union of the friends of all the users related to the image are given direct access. Therefore, it is hard

[1] https://www.facebook.com/about/privacy/your-info-on-fb (Accessed: 28 March, 2014). Note that, Facebook Policies constantly change.

to control who is able to access the content, leading to unwanted audiences, and social implications. In contrast, Erickson and Kellogg [84] argue that discussion and collaboration on the access control rights of shared content are important to increase social privacy.

Motivated by the lack of control of the information flow on the OSNs (Chapter 3), and the social design from Erickson and Kellogg [84], this chapter presents a general model for audience segregation in OSNs. We first generalize the concept of access control based on groups and collaboration. Then, we devise a collaborative sharing scheme for joint content in OSNs; more specifically, we present a scheme in which content-related users can collaboratively decide to disseminate the related information shared in OSNs. Our collaborative sharing scheme is based on some social assumptions, and makes use of secret sharing [180]. Moreover, we show that our scheme is secure towards unwanted audiences, and discuss the efficiency with concrete cryptographic algorithms. Then, we demonstrate the practical challenges of implementing on top of existing OSNs.

5.2 Model

In this section, we model audience segregation on OSNs. We consider OSNs to be represented by an undirected graph, such that friendship connections are symmetric.[2] So that, each user in the OSN is considered to hold a profile $\mathcal{P}$, manage friendship connections $\mathcal{R}$, and share content $\mathtt{m} \in \mathcal{M}$ with $\mathcal{R}$.

5.2.1 Audience Segregation Model

Aligned with the offline audience division [21], we consider user profiles to be a tree-like structure categorized by two types classes: connection and content classes. Connection classes classify user connections $\mathcal{R}$, such as Friends, Family or Co-Workers, and can be represented by different groups $\mathcal{L}$ with overlapping users. Content classes represent the shared content $\mathtt{m}$, so that connection groups are associated to various $\mathtt{m} \in \mathcal{M}$.

Thereby, the mapping between content and connection classes defines the access control rights, so that connections and content form a partially ordered set (lattice). Figure 5.1 illustrates the model, where the connection classes composed of the set $\mathcal{S} = \mathcal{L}^{\mathsf{Friends}}_{\mathsf{Alice}} \cup Bob$ holds access rights to the content $\mathtt{m}_{\mathsf{Work}}$. This

[2] Access control rules and definitions similarly apply to the direct graph setting, whereby friendship connections may not be mutual.

structure allows easy propagation of rights, without overloading users. When new content is introduced, all members from the associated connection class will have access to the content. Similarly, whenever a new connection is added to a connection class, they will have access to the same content as other members of the same class. In particular, the mapping of content can be done by groups and collaboratively, as follows.

Group Access Control. Access to the content should be allowed to a group of users $\mathcal{L}$, such that, for instance, the content $\mathtt{m}$ is only accessible by $\mathcal{L}^{\mathsf{Friends}}_{\mathsf{Alice}}$. However, composed groups are also possible, for instance, $\mathtt{m}_{\mathsf{Work}}$ is accessible by $\mathcal{L}^{\mathsf{Friends}}_{\mathsf{Alice}} \cup \mathsf{Bob}$. Throughout this thesis we denote $\mathcal{S}$ as the set of authorized recipients/viewers of a specific content $\mathtt{m}$.

Collaborative Access Control. Usually, the content shared on OSNs is related to multiple users, namely content-related users, similar to the tag system of Facebook. We denote such shared content as joint content. Following the definition of Erickson and Kellogg [84], we argue that access rights to a content which is related to multiple users should be decided collaboratively among all, or at least a subset of the content-related users.

Although the collaborative access control involves multiple users deciding access rights, it can coexist with the group based access control. Conversely, the content-related users in the collaborative access control may use groups to define the final set of authorized users. In this way users can enforce a fine-grained access control to the joint information. While technically skilled users may find this interesting, less computer-savvy people need to be provided with some default classes of connections, and preferably also a default mapping specifying the privacy level.

5.2.2 Adversarial Model.

We consider as adversaries any unauthorized entity attempting to attain access to joint content $\mathtt{m}$. In practice, there may be a few different adversarial entities, including the OSN provider as well as a curious connection, i.e., friend. The content publisher is assumed to be honest as he already holds the information. In addition, we do not consider denial of service attacks, since we assume that the main motivation of an adversary is to learn, and, thus, will not gain by removing information. However, once the content is distributed, there is no way to prevent a malicious viewer from storing or re-distributing the content. In this case, such user is said to break the social contract established along with the friendship relation.

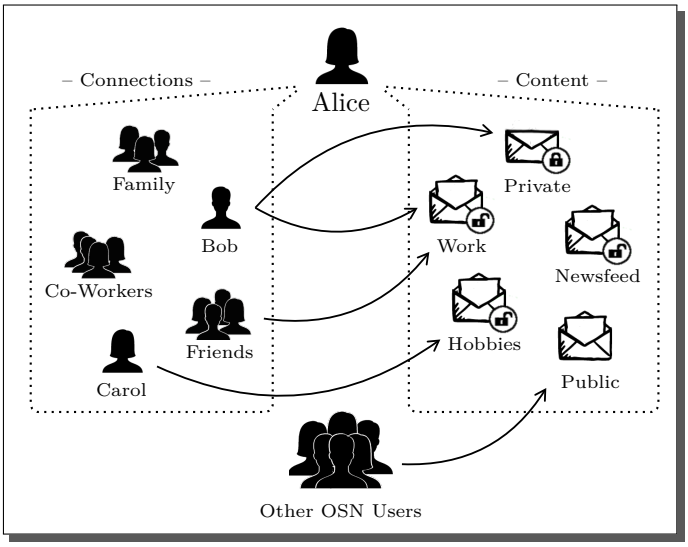


Figure 5.1: Audience Segregation Model.

5.3 Collaborative Access Control

Collaboration and discussion while sharing information represents an important factor and contributor for social privacy, as demonstrated by Erickson and Kellogg [84]. In contrast, the most commonly used approach by OSNs is to give access rights to the union of all content-related users, lacking collaboration, and, thus leading to possible unwanted leakages. For instance, if Alice shares a given content $\mathtt{m}$, and tags Bob and Charlie, then all users in the set $\{\mathcal{R}_{\mathsf{Alice}} \cup \mathcal{R}_{\mathsf{Bob}} \cup \mathcal{R}_{\mathsf{Charlie}}\}$ can view the content $\mathtt{m}$. Thereby, in this section we suggest a practical collaborative access control scheme based on secret sharing. In particular, a collaborative sharing scheme where the content shared $\mathtt{m}$ is private to any unwanted prying eyes, as illustrated in Figure 5.2, attaining the following goals.

– **Collaborative Access Rights.** Access to joint content $\mathtt{m}$ is only granted to viewers who are connected to at least a threshold $t \leq n$ out of n content-related collaborating users (i.e., content-related users choosing to collaborate in the sharing of content $\mathtt{C}$).

– **Content Confidentiality.** Joint content $\mathtt{m}$ should be published in encrypted format $\mathtt{C}$, such that adversaries are not able to infer any information nor access

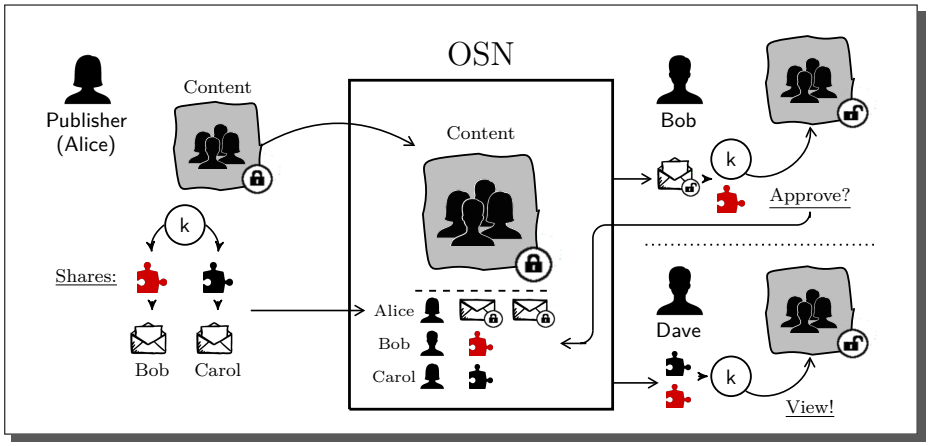


Figure 5.2: Collaborative Scheme. The publisher (Alice) publishes to the OSN the encrypted content related to the tagged users, i.e., Bob and Carol. For each tagged user, the publisher provides the secret, used to encrypt the content; and a share of the secret. The tagged user (Bob) can see the content, and decide weather or not to disseminate their own share among $\mathcal{R}_{\mathsf{Bob}}$. Only viewers that collect enough shares can reconstruct the secret, and hence see the content, e.g., Dave.

the content `m`.

5.3.1 Collaborative Sharing Scheme

We devised a collaborative sharing scheme using secret sharing, where access rights are only passed to other profiles if, and only if, a threshold of $t \leq n$ users out of all n content-related users collaborate, i.e., from the set $\mathcal{S}$.

Definition 10 (Collaborative Sharing Scheme)**.** *Let the universe of users $\mathcal{U}$, a collaborative sharing scheme is composed by a setup algorithm and three protocols, as follows:*

`Setup(`λ`):` A randomized algorithm that generates the user u initial parameters $\mathtt{I} \leftarrow \{(pk, sk), \mathtt{k}\}$ from a security parameter λ.

`Publish(m):` A randomized protocol that generates a fresh secret key `k`, the encryption `C` of `m`, the polynomial $f(\cdot)$ of degree $t-1$, and the shares for the n content-related users, such that $t \leq n$.

`Collaborate(`γ_i`,C):` Used by each of the n content-related users from the set $\mathcal{S}$ to distribute their shares.

`Retrieve(C,`$\delta_{0,\ldots,t}$`):` Extracts the content `m` from the ciphertext `C` using t shares to reconstruct the secret `k`. Otherwise, return $\perp$, if `C` is malformed.

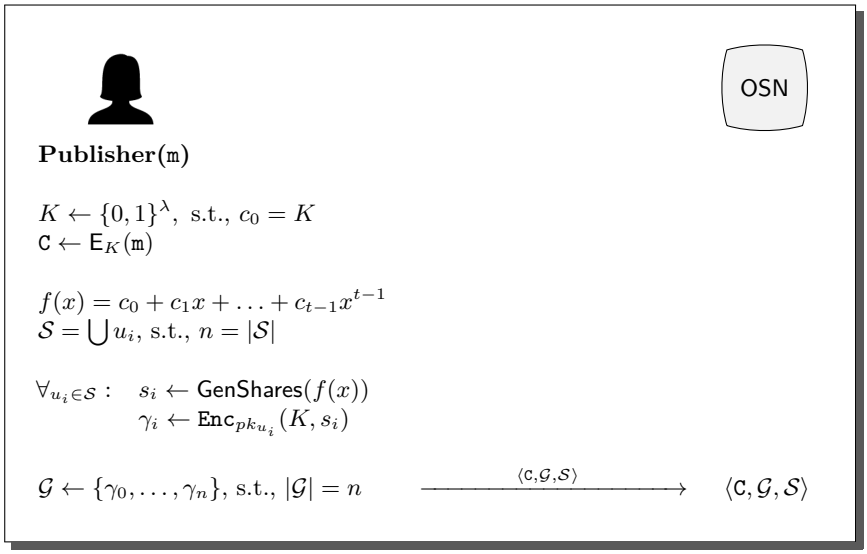


Figure 5.3: Collaborative Scheme: Publish Protocol.

Without loss of generality, we consider Alice to be the publisher, Bob and Carol content-related users, and Dave the viewer.

Setup. Each user runs the setup algorithm once to generate a public-secret key pair (pk, sk) alongside with the random collaborative sharing key k. The public key pk and collaborative sharing secret k are made available, for instance, on the user profile. Whereas pk is publicly accessible, k is restricted to the set $\mathcal{L} \subset \mathcal{R}$. For new users added to $\mathcal{L}$, for instance, after a friendship request acceptance, the procedure of making these two keys available has to be repeated. In addition, for each group $\mathcal{L}$, a different k is generated and made available to the respective set $\mathcal{L}$. However, for the ease of exposition, we consider access rights to be equal among all members in $\mathcal{R}$.

Publishing content. Assuming Alice wants to publish some joint content $\mathtt{m}$ on the OSN, such that the content $\mathtt{m}$ is related to a number of other users. For instance, apart from Alice the set {Bob, Carol} is also associated to $\mathtt{m}$, e.g., "tagged", and thereby must undertake a collaborative sharing approach, so that $\mathcal{S} = \{\mathsf{Alice}, \mathsf{Bob}, \mathsf{Carol}\}$. To publish $\mathtt{m}$, Alice generates a random key K used to produce the authenticated symmetric encryption $\mathtt{C} \leftarrow \mathsf{E}_K(\mathtt{m})$. Thereafter, Alice constructs a polynomial $f(x)$ of degree $t-1$ with $f(0) = K$, with t representing the minimum number of users required to collaborate. After generating n

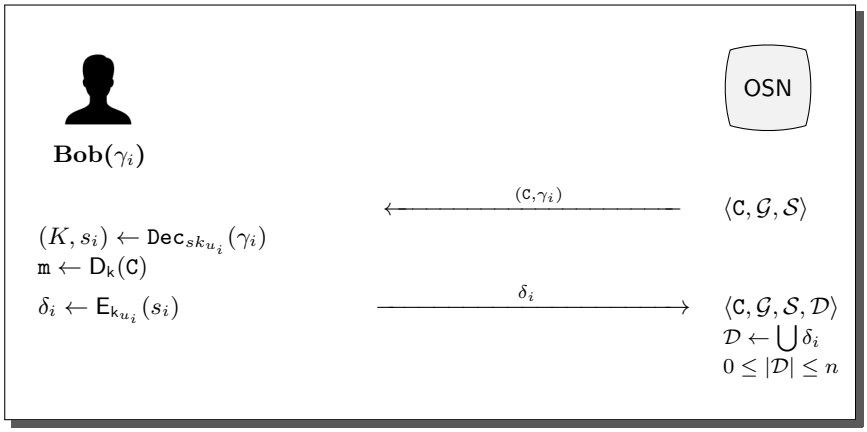


Figure 5.4: Collaborative Scheme: Collaborate Protocol.

different shares, one per content-related user, Alice encrypts the shares along with $\mathtt{k}$ using the content-related user's public keys $\gamma_i \leftarrow pk_{u_i} : u_i \in \mathcal{S}$. The session key K is distributed along with the shares, allowing content-related users the possibility to make informed decisions with respect to the joint content, i.e., by reviewing $\mathtt{m}$. Finally, Alice publishes $\mathtt{C}$ alongside with the set of encrypted shares $\mathcal{G}$ and $\mathcal{S}$. Figure 5.3 illustrates this protocol.

Collaborate with shares. The process of collaboration resumes to the distribution of shares. Each content-related user is empowered to contribute with the distribution of his personal share s_i after reviewing the content $\mathtt{m}$. Subsequently, publishing δ_i, the authenticated encryption of his associated share using the collaborative sharing key, for instance, in the form of a comment. This allows other users in $\mathcal{R}$, to retrieve this encrypted share needed to retrieve the content in a later stage. Figure 5.4 depicts the protocol.

Retrieving content. Any viewer (Dave) requires at least t shares, to be able to reconstruct the secret $\mathtt{k}$, and retrieve the content $\mathtt{m}$. However, to get hold of t shares, the viewer needs to have access to the collaborative sharing key from t collaborating users. This means that, in order to view $\mathtt{m}$, one needs to be a member of at least t sets $\mathcal{R}$ of collaborating users, in order to have access to the corresponding collaborative sharing keys $\mathtt{k}$. Recall that these keys were made available to the users in $\mathcal{R}$ during setup. Therefore, the authenticity of the decrypted shares is guaranteed by the used authenticated encryption scheme. The protocol to retrieve the content is depicted in Figure 5.5.

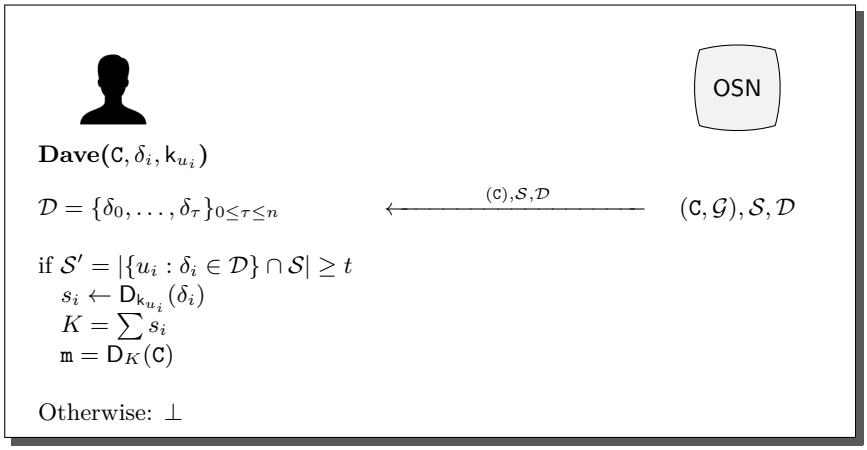


Figure 5.5: Collaborative Scheme: Retrieve Protocol.

5.3.2 Threshold Selection

Since the set of authorized viewers is constructed implicitly, the threshold t represents an important parameter, quantifying the degree of content dissemination. Hence, deciding the threshold is a matter of design that needs to be decided either by the OSN (top down) as part of their service, or based on the trust necessary and the sensitivity of the information (a subjective value to be decided in a given application domain). Higher trust or lower sensitivity is likely to imply a low threshold, whereas lower trust or greater sensitivity would imply a high threshold. Different trust algorithms or interfaces to solve this problem are described in [101, 160], and represent a topic for future research.

However, we stress that a very high threshold could lead to a deviation from the purpose of sharing, as the content will only be accessible by a very limited amount of viewers. For instance, a threshold value of 2 or 3 may probably suffice to stop unwanted coworkers of viewing pictures of a party with old friends, while at the same time not denying access to other old friends that might not be friends with all the tagged users.

5.4 Security Evaluation

To achieve content confidentiality, adversaries with no knowledge of the collaboration keys k should not infer information with respect to the joint content $\mathtt{m}$. Authenticated encryption schemes, used to encrypt the joint content

and shares, provide security against chosen ciphertext attacks (IND-CCA), thus the public key encryption scheme requires to be secure under chosen plaintext attacks (IND-CPA), and not IND-CCA. To collaborate, users verify the joint content authenticated with $\mathtt{k}$, before publishing the share to the subset of connections $\mathcal{R}$. Moreover, motivated adversaries intending to use a content-related user as a decryption oracle to obtain the share s_i, require access to the collaboration key $\mathtt{k}$ of the associated user, in order to derive a valid authenticated encrypted content. This case subsequently implies that the attacker either knows $\mathtt{k}$ and has access to the joint content $\mathtt{m}$, or forwarded the original $\mathtt{C}$, and γ_i to the user. Similarly, substitution attacks on $\mathtt{C}$ or on any of the γ_i represent a very hard task for other users on the OSN and the OSN provider itself. The (t, n)-secret sharing scheme requires at least the threshold number t of shares to reconstruct the shared secret $\mathtt{k}$. As Shamir's secret sharing scheme is information theoretically secure, adversaries cannot infer any information about $\mathtt{k}$ when knowing less than t shares. This means that until t tagged users collaborate, the joint content remains fully confidential. At the moment that t or more tagged users collaborate, the ability to view the joint content $\mathtt{m}$ boils down to being able to collect t shares, which means having access to t collaboration keys $\mathtt{k}$ of tagged users that collaborated in the sharing of the joint content. Hence, collaborative sharing is achieved, so that only users related to enough collaborating tagged users can see the content $\mathtt{m}$. If the OSN is considered adversarial, then it should be kept outside the audience, collaboration keys $\mathtt{k}$ cannot be disseminated using the OSN. Therefore, users should leverage external channels, as well as publishing encrypted under the public keys of different $\mathcal{L}$ or all connections $\mathcal{R}$.

5.5 Implementation Details

The implementation of our collaborative sharing scheme could be integrated with the OSN design. However, it is hard to advocate for changes on current OSN architectures mostly if they produce a problem to the business model. In this section, we sketch the design and architectural possibilities, and later we demonstrate that the cryptographic overhead is limited.

5.5.1 Design Decisions

Our scheme implementation can be fulfilled by a browser extension, e.g., Firefox or Chrome extensions. The development of browser extensions for popular browsers is done in Javascript, which represents a bottleneck for performance of asymmetric cryptographic operations. However, the SJCL [189] library

handles symmetric cryptography efficiently. Therefore, to address this issue, we divide our design into a client side responsible for the user interactions, and a (local) server side responsible for the cryptographic operations. The communication between both is performed locally in the same machine through socket communication. Such design allows the cryptographic library to be implemented efficiently [176]. In addition, it eases the process of possible migrations to different browser platforms.

Public keys are made publicly available by each user on their OSN profile, while the collaborative secret k is shared only with $\mathcal{R}$.[3] In this way, the tuple (pk, k) can be automatically retrieved from the OSN, and stored locally. To engage in the scheme, the user publishes the encryption of the content $\mathtt{C}$ along with the encrypted shares, and the list of tagged users $\mathcal{S}$. Later, content-related users in $\mathcal{S}$ can collaborate by disseminating their shares as comments. In order to be aligned with the general *Terms of Service* of popular OSNs, where encryption is usually not allowed, $\mathtt{C}$ can be published in an external storage, such as Dropbox, and the link to the external server in the OSN. The browser extension automatically translates the encrypted content, retrieves the t-collaborative shares, decrypts the respective shares, and subsequently displays the content.

5.5.2 Performance

For a security parameter $\lambda = 128$ *bit*, the size of the secret key k is 128 *bit*. Alongside, the size of each share, using Shamir's secret sharing scheme, is 128 *bit*. Using the elliptic curve cryptography (ECC) public key encryption scheme with the curve25519 [26], which is among the fastest curves for a security parameter of 128 *bit*, benchmarks show that, elliptic curve point multiplication on this curve takes about 60 *msec* on a single core desktop machine, while on a smartphone processor needs 200 *msec*.[4]

In addition, each 256 *bit* number is a valid point on the curve, allowing the encryption of a 256 *bit* message $M = \mathsf{k} \parallel s_i$ as follows: $R = rP, M \oplus rY$, for the random number r, the curve generator P, and the public key $Y = sP$ of the intended recipient of the message. For the private key s, decryption returns $M = M \oplus s(rP) \oplus rY$, such that $Y = sP \wedge s(rP) \oplus rY = 0$. This represents the summarized version of the ECC-ElGamal encryption scheme, and is semantically secure (IND-CPA) under the computational Diffie-Hellmann (CDH) assumption.

[3] This assumes trust on the OSN. However, publishing k encrypted under the public keys of $\mathcal{R}$ allows to consider the OSN a honest but curious adversary.

[4] Supercop benchmark tool: http://bench.cr.yp.to/supercop.html

Table 5.1: Collaborative Scheme Overhead: Computational effort, and communication overhead per protocol.

Computational effort	**Publish**	**Collaborate**	**Retrieve**
EC multiplication	$n+1$	1	
Authenticated en-/decryption	1	2	$t+1$
Scalar multiplication	$n(t-1)$		t
Scalar addition	$n(t-1)$		t
Bitwise exclusive OR	1	1	
Communication overhead [bytes]	$(n+1)*32$	48	$t*16$

Publish. For the publisher, the computational effort and communication overhead can be reduced by almost 50% by choosing one random number for all public key encryptions at the time of publishing content `m`. As such we only need $n+1$ EC point multiplications (transfer $n+1$ EC points) instead of $2n$ EC point multiplications (transfer $2n$ EC points). For efficiently evaluating the polynomial, one can make use of Horner's method [121], resulting in $t-1$ multiplications and $t-1$ additions for a polynomial of degree $t-1$.

Collaborate. The tagged user willing to collaborate, first decrypts, using his private key, his share together with the symmetric encryption key that was used to encrypt the content. After approval of the authenticated decrypted content, the user then encrypts his share under a collaborative secret.

Retrieve. To view content, one needs to collect t shares (for which one has to do t authenticated decryptions), then combine these shares using Lagrange interpolation and finally use the resulting key to decrypt the content. In case the collaboration keys are not cached by the browser extension, one needs an extra transfer of $t*16$ bytes.

Table 5.1 depicts an overview of the computational effort, ordered according to efficiency, and communication overhead in comparison with posting the content `m` directly on the OSN.

Elliptic curve multiplications require higher computational effort than authenticated encryption schemes. Thus, native support of AES instructions, AES-based authenticated en-/decryption may even be more efficient, for instance, using AES-CCM [206]. The computational cost for scalar arithmetic and bitwise exclusive ORs is negligible. In summary, the computational effort for publishers

rises with the number of tagged users n, whereas the computational effort for viewers is minimal. In fact, viewers require no costly public key operations which are solely dependent on the selected threshold t, and independent of n.

5.6 Summary

Motivated by the audience segregation phenomenon and the scarcity of collaborative access control solutions on Online Social Networks (OSNs), this chapter elaborates on access control on OSNs. After generalizing the notion of access control, we devised a solution for a collaborative sharing scheme for OSNs based on secret sharing, while leaving the group access control enforcement as topic for Chapter 6. Each content related user can collaborate by disseminating his share, and only users with access to at least t shares can access the content. Therefore, the set of authorized viewers is constructed implicitly. At the same time, protection is guaranteed from any curious viewers without knowledge of at least t shares. Furthermore, collaboration becomes mandatory, and in the process provides related users with information about co-related users privacy preferences. Finally, we have evaluated the cost of the scheme: we have estimated the computational overhead and demonstrated that a practical implementation is feasible.

6

Information Sharing

"Cryptography is the essential building block of independence for organisations on the Internet, just like armies are the essential building blocks of states . . . "

– Julian Assange, *Cypherpunks: Freedom and the Future of the Internet (2012)*

The content shared on OSNs represent a threat to the privacy of users, vulnerable for data leakages problems and unauthorized access, as defined by surveillance categories, introduced in Chapter 2. Hence, aligned with the audience segregation definitions from Chapter 5, we present different private sharing (PS) schemes for protecting information and delivering *end-to-end encryption* on OSNs. After formalizing the PS scheme functionality and modeling the end-to-end encryption paradigm for OSNs, we present three efficient instantiations obtained from Symmetric Encryption, Broadcast Encryption, and Identity-Based Encryption techniques, respectively. For each, we elaborate on the technical details, key management, and our experiences with implementing the different schemes. The schemes are designed to be applied to OSNs using Scramble (Appendix A) with a minimum overhead on viewers.

Publications.

[20] Beato, F., Ion, I., Čapkun, S., Preneel, B., and Langheinrich, M., For some eyes only: protecting online information sharing. In *ACM CODASPY 2013* (Feb. 2013).

[22] Beato, F., Kohlweiss, M., and Wouters, K., Scramble! your social network data. In *PETS 2011* (Jul. 2011).

[23] Beato, F., Meul, S., and Preneel, B., Practical Identity Based Broadcast Encryption for Online Social Networks. In *COSIC internal report* (2014).

Contributions. Main author together with Iulia Ion and Stijn Meul.

Chapter Outline. This chapter assembles the following contributions from different articles published and presented at international peer-reviewed conferences [20], [22], and a internal report [23]. Aligned with the audience segregation definitions from Chapter 5, we develop cryptographic schemes from different building blocks that allow selective access rights enforcement per content on OSNs. Further, we evaluate the security of the schemes, and compare the key management overhead, according to the OSN limitations.

6.1 Motivation

Online Social Networks (OSNs), such as Facebook, Google+, and Twitter have become prominent communication channels for millions of users, providing efficient, and reliable sharing and dissemination channels. Given their prominent role and design, OSNs end up centralizing and storing large amounts of information, exposing users to several privacy threats. Although most OSNs grant users with customizable "privacy settings", as aforementioned in previous chapters, these settings offer a limited level of control, relying on providers to be trustworthy during management and enforcement. These preferences usually do not exclude OSNs from the recipients, since they need to access the shared information, to share with third parties as a consequence of their economical business model [161, 171], consequently exposing users to several privacy threats. Aside from the business model dependence issues, the reports of mass breaches of large datasets of personal information are becoming increasingly common [38], as evident in recent accounts of surveillance programs like Prism [204], and the recent iCloud mega leak [139].

All these worrisome issues motivate the need to (at least) implement more reliable privacy protection mechanisms, such as providing users with *end-to-end encryption* properties in OSNs. Similarly to client-server applications, with the

use of TLS [78], *end-to-end encryption* in OSNs must provide content protection from an initiator to the specific final destination. However, in OSNs this notion requires extensions, as the OSN is mediating the channel and storage. Thus the content must be protected from the publisher to the intended viewers, while communicating through OSNs. This allows users to selectively enforce a more granular control over their data shared in OSNs. Even though privacy is an important requirement, arguably supported by OSN providers, it has been neglected due to complexity and added overhead claims [150].

Therefore, this chapter formalizes the concept of *end-to-end encryption* aligned with the notion of private sharing in OSNs, in order to keep the user's content private when published and shared in OSNs. In addition, it presents three instantiations of private sharing (PS) schemes for information sharing in OSNs, delivering *end-to-end encryption*. For each of the PS-schemes we discuss the implementation details and challenges entangled with the development of Scramble, described in Appendix A. Consequently, we demonstrate the minimum complexity and overhead added. Scramble is an open source tool, developed as a browser extension, that empowers users with definition of access control rules and enforcement of such rules on top of popular OSNs by means of the PS-schemes.

6.2 Model

We consider a user u to be a member of an OSN, and to be connected by a friendship relationship $\mathcal{R}_u$ with other users in the same OSN [40]. For simplicity's sake, we assume relationships among users to be symmetric. Inherently, we infer that users interact using OSNs with the intent of sharing content information $\mathtt{m}$ with other users in the same OSN, while keeping $\mathtt{m}$ oblivious to outsiders, as depicted in Figure 6.1.

6.2.1 Private Sharing

Usually, OSN users disseminate information to multiple recipients. In particular, targeting as recipients subsets of connections $\mathcal{S} \subset \mathcal{R}$, following the definitions presented in Chapter 5, such that, $\mathcal{S}$ can be composed of a single or a combination of audience segregation groups $\mathcal{L}$. Hence, for the security parameter λ of l bits, and the set of desired recipients $\mathcal{S}$, such that $\mathcal{S} = \{u_1, .., u_\eta\}$, where $\eta = |\mathcal{S}|$. We model a general PS-scheme for OSNs as follows.

Definition 11 (OSN PS). *For the universe of users* $\mathcal{U} = \{u_0, \ldots, u_N\}$ *represented in an OSN, an OSN private sharing (PS) scheme* Π *is composed of four randomized algorithms:*

- Π.`Setup`(λ): *On the input of a security parameter* λ*, generates the public parameters* params.
- Π.`KeyGen`($params, u_i$) *Returns the public-private key pair (*pk_i,sk_i*) for the user* u_i *according to the* params.
- Π.`Publish`($params, \mathtt{m}, \mathcal{S}$): *Takes a subset* $\mathcal{S}$*, the respective public keys, s.t.,* $\mathcal{S} \subset \mathcal{U}$*, along with the secret message* `m`*, and generates* `C`.
- Π.`Retrieve`(sk_i, `C`): *Reconstructs* `m` *from* `C` *using the private key* sk_i *if, and only if,* $u_i \in \mathcal{S}$*. Otherwise, return* $\perp$.

6.2.2 Adversarial Model

In this chapter we consider an adversary to be any entity attempting to passively access the shared information `m` by monitoring the communication channel, with no incentive to tamper with the content. This can be any curious user in the OSN, the OSN provider, or even a government agency [204]. Such adversaries should not learn the content of `m`, and the identity of members in the recipient set $\mathcal{S}$. Otherwise the adversary is considered to break both confidentiality and the recipient set anonymity [16].

We assume, however, that an adversary cannot control the user computing environments, such as the user's browser, computer, and any extra device used in the protocol. In addition, we consider that once the content is distributed, there is no way to prevent a *malicious* authorized recipient from storing or re-distributing `m`. In this case, such a recipient is said to break the social contract associated with the establishment of the friendship relation.

6.2.3 End-to-End Encryption for OSNs

End-to-end encryption is a (tele)communication paradigm that provides an encrypted communication channel between two parties, such that the data sent by an originator is only accessible by the intended recipient(s). In common client-server applications this is achieved with the use of TLS [78]. However, on the OSN use case, TLS solely protects the communication channel between users and the OSN against eavesdroppers, and man-in-the-middle attacks. Naturally, any adversary with access to the OSN can access the exchanged content. Hence, we argue that *end-to-end encryption* on OSNs should protect the content from the publisher to the viewers, towards any unwanted recipients even when storing

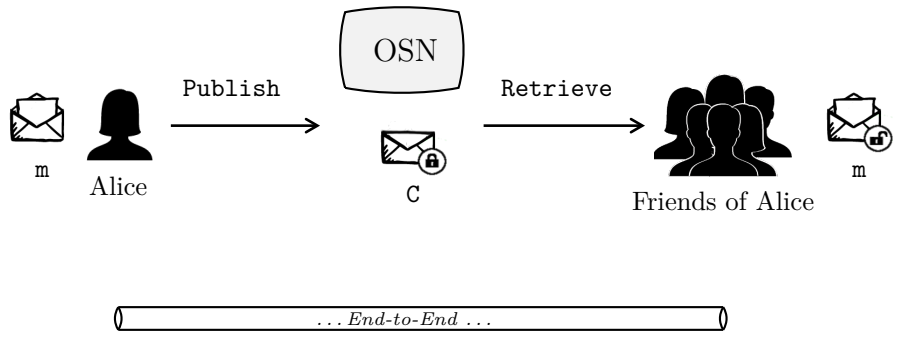


Figure 6.1: End-to-End Encryption Model for OSNs.

m in the OSN, as illustrated in Figure 6.1. In particular, we aim to protect shared content on OSNs by ensuring confidentiality, data integrity, and recipient anonymity. This allows users to selectively enforce access control, as defined in Chapter 5, without having to rely on the privacy preferences offered by OSNs. At the same time, we aim at limited modifications to the OSN environment, and require as little effort as possible in order to achieve a user-friendly cryptographic scheme as defined by Balsa *et al.* [12]. Protection against traffic analysis or timing attacks is, however, beyond the scope of the protocols described in the chapter.

Therefore, we consider an OSN PS-scheme to deliver *end-to-end encryption* on OSNs, if it fulfills the following security, and privacy requirements.

– **Correctness.** The OSN PS-scheme is correct if for every member u_i present in the recipient set $\mathcal{S}$, such that, $sk_i \leftarrow \texttt{KeyGen}(\texttt{params}, \texttt{u}_\texttt{i})$, it outputs the message:

$$\texttt{m} = \texttt{Retrieve}(params, sk_i, \texttt{Publish}(params, \mathcal{S}, \texttt{m})).$$

– **Confidentiality.** The confidentiality property holds if the OSN PS-scheme is achieves ciphertext indistinguishability. In particular, if the adversary $\mathcal{A}$ does not win the following game between the Challenger Ch with high probability.

Game 1 (OSN PS Security)**.** *Let* $\Pi \leftarrow \{$Setup, KeyGen, Publish, Retrieve$\}$ *be a OSN PS-scheme,* $\mathcal{A}$ *a probabilistic polynomial time (PPT) adversary, and* Ch *the challenger. We say that* Π *is (IND-CCA) secure if* $\mathcal{A}$ *wins the below game with* Ch *with negligible probability.*

Init: $\texttt{Ch}$ *runs* $Setup(\lambda)$*, and gives* $\mathcal{A}$ *the resulting params.*
Setup: $\texttt{Ch}$ *generates keys for each potential recipient* $i \in \mathcal{S}$*, running* $sk_i \leftarrow \texttt{KeyGen}(\texttt{params}, \texttt{u}_\texttt{i})$*, and sends each* pk_i *for* $i \in \mathcal{S}$ *to the* $\mathcal{A}$*.*
Phase 1: *The* $\mathcal{A}$ *adaptively performs queries to the* $Retrieve(\texttt{C}, sk)$ *oracle.*
Challenge: $\mathcal{A}$ *sends to the* $\texttt{Ch}$ *two different messages* $(\texttt{m}_0, \texttt{m}_1)$*, s.t.,* $|\texttt{m}_0| = |\texttt{m}_1|$*.* $\texttt{Ch}$ *picks a random bit* $b \in \{0,1\}$*, runs* $\texttt{C}' \leftarrow \texttt{Publish}(params, \mathcal{S}_1, \texttt{m}_b)$*, and sends* $\texttt{C}'$ *to* $\mathcal{A}$*.*
Phase 2: $\mathcal{A}$ *adaptively issues additional decryption queries* $Retrieve(\texttt{C}', sk)$*, such that,* $\texttt{C} \neq \texttt{C}'$*.*
Guess: $\mathcal{A}$ *outputs a guess* $b' \in \{0,1\}$ *and wins if* $b = b'$*.*

The $\mathcal{A}$ *advantage to win the above game is defined as:*

$$\mathsf{Adv}^{\mathrm{Ind}}_{\mathcal{A},\Pi} = |Pr[b = b'] - \frac{1}{2}|.$$

– **Recipient Set Privacy.** The high-level idea behind recipient set privacy is as follows. For any two recipient sets $\mathcal{S}_0$ and $\mathcal{S}_1$ an adversary $\mathcal{A}$ cannot distinguish between a ciphertext intended for the recipient set $\mathcal{S}_0$, and a ciphertext intended for the recipient set $\mathcal{S}_1$, given that $\mathcal{A}$ does not possess the secret key of any user in $\mathcal{S}_0 \cup \mathcal{S}_1$.

Game 2 (OSN PS Recipient Anonymity)**.** *A OSN PS-scheme* $\Pi \leftarrow \{\texttt{Setup}, \texttt{KeyGen}, \texttt{Publish}, \texttt{Retrieve}\}$ *is recipient anonymous (ANO-PS) if a PPT adversary* $\mathcal{A}$ *wins the following game with the challenger* $\texttt{Ch}$*, with negligible probability:*

Init: $\texttt{Ch}$ *runs* $Setup(\lambda)$*, and gives* $\mathcal{A}$ *the resulting params.* $\mathcal{A}$ *outputs* $\mathcal{S}_0, \mathcal{S}_1 \in \mathcal{U}$*, such that,* $|\mathcal{S}_0| = |\mathcal{S}_1|$*, and* $(\mathcal{S}_0 \triangle \mathcal{S}_1) = \emptyset$*.*
Setup: $\texttt{Ch}$ *generates keys for each potential recipient* i*, running* $sk_i \leftarrow \texttt{KeyGen}(\texttt{params}, \texttt{u}_\texttt{i})$*, and sends each* pk_i *for* $i \in \mathcal{S}_0 \cap \mathcal{S}_1$ *and* $sk_i \in \mathcal{S}_0 \cup \mathcal{S}_1$ *to the* $\mathcal{A}$*.*
Phase 1: $\mathcal{A}$ *adaptively issues decryption queries* $q_1 = (i, \texttt{C})$*, and* $\texttt{Ch}$ *returns* $\texttt{Retrieve}(params, sk_i, \texttt{C})$*.*
Challenge: $\mathcal{A}$ *gives the* $\texttt{Ch}$ *a message* $\texttt{m}$*. The* $\texttt{Ch}$ *picks a random bit* $b \in \{0,1\}$ *and runs* $\texttt{C}' \leftarrow \texttt{Publish}(params, \{u_i | u_i \in \mathcal{S}_b\}, \texttt{m})$*, and sends* $\texttt{C}'$ *to* $\mathcal{A}$*.*
Phase 2: $\mathcal{A}$ *adaptively issues additional decryption queries* $q_2 = (i, \texttt{C})$*, such that* $\texttt{C} \neq \texttt{C}'$*.*
Guess: $\mathcal{A}$ *outputs a guess* $b' \in \{0,1\}$ *and wins if* $b = b'$*.*

The advantage of $\mathcal{A}$ *of winning the above game is defined as:*

$$\mathsf{Adv}^{\mathrm{RecPriv}}_{\mathcal{A}} = |Pr[b = b'] - \frac{1}{2}|.$$

Remark 1. *The definition of recipient anonymity for an OSN ANO-PS-scheme can be relaxed to outsider recipient anonymity oANO-PS-scheme by forcing the restriction* $(\mathcal{S}_0 \cap \mathcal{S}_1) = \emptyset$ *instead of* $(\mathcal{S}_0 \triangle \mathcal{S}_1) = \emptyset$ *in the* Init *phase.*

We note that due to the OSNs design, recipient privacy definition holds up to the first comment, or reply by a user in $\mathcal{S}$. However, we assume that replies to encrypted content are made within an encrypted domain.

6.3 Symmetric-Key based PS scheme

The most generic construction of a OSN PS-scheme, denoted OSN PS-SK, uses shared keys to encrypt shared content. So that, for the multiple recipient setting, the shared keys k_i are chosen per content $\mathtt{m}$, and re-distributed among a single or a group of recipients $\mathcal{S}$. The obvious solution is to share different keys among $\mathcal{R}$, however, this increases the key storage and distribution overhead of the keys, becoming bounded to size of $\mathcal{R}$, i.e., $\mathcal{O}(\mathcal{R})$. Although sharing a single secret among all $\mathcal{R}$ solves the storage issue, it does not offer fine grained access control. This further means that sharing different keys among different groups still delivers some storage overhead for publishers.

PS-SK Scheme. For the universe of users $\mathcal{U}$, represented in an OSN, an OSN private sharing scheme using symmetric keys (PS-SK) is composed of four randomized algorithms, described as follows.

Π-SK.Setup(λ): Outputs the public *params* of the system with respect to the security parameter λ.

1. Choose a one-way hash function $\mathtt{H} : \{0,1\}^* \rightarrow \{0,1\}^\lambda$.
2. Semantic secure authenticated $\langle \mathtt{C} \parallel \mathtt{T} \rangle \leftarrow \mathsf{E}(\cdot), \mathsf{D}(\cdot)$.

Π-SK.KeyGen($params, \mathcal{S}$): On the input of a user defined set, output a random secret $K \leftarrow \{0,1\}^\lambda$, and a binary tree T with root K, such that, the left and right leaves are computed as $\langle L|R \rangle = \mathsf{k}_i \leftarrow \mathtt{H}(K \parallel depth_i)$.

Π-SK.Publish($params, \mathcal{S}, \mathsf{k}$): Takes the message $\mathtt{m}$ along with the leaf key k_i, for the subset $\mathcal{S}$, and output a broadcast message $\mathtt{C}$ along with the $depth_i$ of the tree T.

1. (Optional) $\mathsf{k}_j \leftarrow \mathtt{H}(\mathsf{k}_i \parallel depth_i + 1)$.
2. $\langle \mathtt{C}, \mathtt{T} \rangle \leftarrow \mathsf{E}_{\mathsf{k}_j}(\mathtt{m})$.

Π-SK.Retrieve($params, \mathsf{k}, \mathtt{C}$): On the input of the encrypted message $\mathtt{C}$, a valid secret key k_i, and the associated $depth_i$, reconstruct the plaintext message $\mathtt{m}$.

1. Compute $\langle \mathtt{m}, \mathtt{T}' \rangle \leftarrow \mathsf{E}_{\mathsf{k}_i}(\mathtt{C})$.
2. Verify if the tag $\mathtt{T}' \stackrel{?}{=} \mathtt{T} \in \mathtt{C}$, and return $\mathtt{m}$. Otherwise return $\perp$.

Correctness. Correctness is valid if the authenticated symmetric encryption is also correct, such that $\mathtt{m} = \mathsf{D}_{\mathsf{k}_i}(\mathsf{E}_{\mathsf{k}_i}(\mathtt{m}))$, for the valid shared key k_i.

Complexity. The decryption process is very efficient with the ciphertext size bounded to λ, and the size of the shared content $\mathtt{m}$. Each user requires to store the private seed K, and at most $|\mathcal{R}|$ different keys, i.e., one per connection. However, this may grow for viewers present in different groups of other users.

Security Analysis. We now show that the OSN PS-scheme provides *end-to-end encryption* on OSNs, by achieving the requirements from Section 6.2.3.

Theorem 1. *Let atk* $\in \{CPA, CCA\}$*. If the OSN PS-SK scheme is correct, and the* $\mathsf{E}(\cdot)$ *is an atk secure authenticated encryption scheme, then a PS-SK scheme is also atk recipient private.*

Proof Sketch: For the PS-SK scheme it suffices to show that $\mathsf{E}(\cdot)$ scheme is secure, as the key-privacy follows. Hence, for any secure authenticated encryption scheme, it is hard for an adversary $\mathcal{A}$ to win the indistinguishability game with a non negligible advantage, or to forge the authentication tag $\mathtt{T}$ after tampering with the content. The recipient set privacy follows, as the shared keys k_i are independent from the identity of the users in $\mathcal{S}$. Therefore, $\mathcal{A}$ cannot distinguish whether $\mathtt{C}_1$ and $\mathtt{C}_2$ are from set $\mathcal{S}_1$ and $\mathcal{S}_2$, respectively, better than the random guess $\frac{1}{2}$. Therefore, the PS-SK scheme is a secure *end-to-end encryption* scheme for OSNs.

Key Management. The OSN PS-scheme assigns shared secrets k_i per group using a tree-based hierarchical structure. Publishers are required to store a single key K as the root of the binary tree T, along with the *depth* of each group branch. The viewers' key storage is bounded to binary-tree level or the number of groups they belong to, thus, on the worst case $\mathcal{O}(\log n)$. However, key distribution requires the use of out-of-band secure channels. The process of adding new members does not directly give access to previous shared content, as sharing new branch keys k_j does not reveal any information about the previous branch, due to the one-wayness property of the hash function. Yet group key revocation forces re-randomization and re-sharing of a new branch key among members of the group $\mathcal{L}$. For example, for the new key $\mathsf{k}_i \leftarrow \mathtt{H}(\mathsf{k}_j \parallel depth_j \parallel r)$, s.t., $r \stackrel{r}{\leftarrow} \{0,1\}^\lambda$ requires re-sharing r over a secure channel. Although lowering the key storage, the tree structure does not offer a fine-grained access control per

content, but a hierarchical one. Even though other symmetric-key based group-key generation approaches exist, all require a secure channel for key distribution, or an authenticated public key infrastructure. For instance, to achieve a more flexible access control using dynamic group key agreement Kim *et al.* [129] suggest to collaboratively blend different users trees T using Diffie-Hellman key exchanges.

Implementation. The implementation of these scheme, and integration into Scramble on a user's machine relies on the implementation of the underlying symmetric key encryption. In fact, symmetric-key operations can be efficiently executed in Javascript using the Stanford Javascript Crypto Library (SJCL) [189], such as AES-CCM, and HMAC-SHA-256, taking about 2 *msec* for each AES 128 *bit* operations.

6.4 Public-Key based PS scheme

For the PS-scheme construction using public keys, we describe the construction used in the initial version of Scramble, and detail the extended version [20, 22]. We start to demonstrate how to construct an outsider anonymous recipient private scheme based on a general hybrid cryptographic scheme such as the one presented in the OpenPGP standard [44], and the one from Barth *et al.* [16]. Using hybrid or broadcast encryption (BE) techniques simplifies the shared secret exchange problem by using public keys. Therefore, to publish it suffices to generate a fresh random secret, encrypt that fresh secret using the public keys of all the intended recipients, and subsequently use the fresh secret to symmetric encrypt the message.

PS-BE Scheme. For the universe of users $\mathcal{U}$ represented in a OSN represented by their public keys pk_i, a OSN private sharing (PS) scheme Π is composed of four randomized algorithms, as follows.

$\texttt{Setup}(\lambda)$: Outputs the public *params* of the system with respect to the security parameter λ.

1. Select a strongly correct, key private $\texttt{PKE}=\{\texttt{KeyGen}(\lambda), \texttt{Enc}_{pk}(\cdot), \texttt{Dec}_{sk}(\cdot)\}$ scheme.
2. Choose a one-way hash function $\texttt{H} : \{0,1\}^* \rightarrow \{0,1\}^{\lambda}$.
3. Select a strongly unforgeable $\texttt{DSig} = \{\texttt{KeyGen}(\lambda), \texttt{Sign}_{sgk}(\cdot), \texttt{Ver}_{vk}(\cdot)\}$ scheme.
4. Choose a semantically secure $\mathsf{E}(\cdot), \mathsf{D}(\cdot)$.

$\texttt{KeyGen}(params, u_i)$: On input of the public $params$ and the user u_i identity, generates a valid public-private encryption key pair $(pk_i, sk_i) \leftarrow \texttt{PKE.KeyGen}(\lambda)$.

$\texttt{Publish}(params, \mathcal{S}, \texttt{m})$: Takes the message $\texttt{m}$, the subset $\mathcal{S}$ of size η containing pk_j, and the public parameters $params$, output a broadcast message $\texttt{C}$.

1. Generate a random symmetric session key $\texttt{k} \leftarrow \{0,1\}^l$, and a fresh public-private signing key pair $(vk_i, sgk_i) \leftarrow \texttt{DSig.KeyGen}(\lambda)$.
2. For each recipient $pk_i \in \mathcal{S}$, compute the ciphertext c_j, running the `PKE.Encrypt` algorithm, as follows.

$$c_j \leftarrow \texttt{Enc}_{pk_j}(\texttt{k}, vk_i)$$

3. $h_{\texttt{m}} = \texttt{H}(\texttt{m})$, and $\sigma \leftarrow \texttt{Sign}_{sgk_i}(h_{\texttt{m}})$.
4. Let c_1 be the (random) concatenation of $c_j \in \mathcal{S}$.
5. Let $c_2 \leftarrow \texttt{E}_{\texttt{k}}(\texttt{m} \parallel \sigma)$, then the final output is:

$$\texttt{C} \leftarrow \langle c_1 \parallel c_2 \rangle$$

$\texttt{Retrieve}(params, sk_{\texttt{id}_i}, \texttt{C})$: on input of the broadcast message $\texttt{C}$, and the private key sk_i of user u_i, reconstruct the plaintext message $\texttt{m}$, as follows.

1. Parse $\texttt{C} = c_1 \parallel c_2$.
2. Compute $\texttt{k} \leftarrow \texttt{Dec}_{sk_i}(c_{1,i})$.
3. Extract $\langle \texttt{m}, \texttt{T} \parallel \sigma \rangle \leftarrow \texttt{D}_{\texttt{k}}(c_2)$.
4. Verify if: $\texttt{true} \leftarrow \texttt{Ver}_{vk_i}(\sigma, \texttt{H}(\texttt{m}))$. Otherwise, return $\perp$.

Correctness. The OSN PS-BE scheme is correct if for every member $pk_i \in \mathcal{S}$, s.t., $sk_i \leftarrow \texttt{KeyGen}(params, u_i)$, and $\texttt{m} = \texttt{Retrieve}(\texttt{params}, \texttt{sk}_\texttt{i}, \texttt{Publish}(params, \mathcal{S}, \texttt{m}))$ holds if $\texttt{m} \leftarrow \texttt{D}_{\texttt{Dec}_{sk_i}(c_i)}(c_2)$ also hold. This is true for every correct `PKE`, and correct authenticated symmetric encryption scheme.

Complexity. Decryption requires on average $|\mathcal{S}|/2$ tries to obtain the valid c_i, and respective key $\texttt{k}$. To improve the decryption efficiency while increasing encryption, and get $\mathcal{O}(1)$, it suffices to add additional placeholders to each c_i $(\texttt{H}(g^{a \cdot r}))$ with r being a public random value, similarly to Barth *et al.* [16], as illustrated in Figure 4.4 on Chapter 4. Further, an additional g^a is added for each pk, and $a \xleftarrow{r} \{0,1\}^\lambda$ to each sk. The ciphertext size is linear in the size of the recipient set $\mathcal{S}$, i.e., $\mathcal{O}(\mathcal{S})$. Besides that there is no storage overhead for the session key $\texttt{k}$, as it is encrypted in the content, the storage of public keys may be high, and dependent on the size of the connections $\mathcal{R}$.

Security Analysis. We now show that the OSN PS-BE scheme fulfills the requirements from Section 6.2.3, and thus is a secure *end-to-end encryption*

scheme on OSNs. We stress that, for social reasons, we aim at outsider anonymous recipient property rather than full recipient anonymity.

Theorem 2. *Let atk* $\in \{CPA, CCA\}$*. If the OSN PS-BE scheme is correct, the* `PKE` *scheme is atk-secure and atk-key private, the digital signature scheme* $\mathtt{DSig}(\cdot)$ *is unforgeable, and the* $\mathsf{E}(\cdot)$ *is a semantically secure encryption scheme. Then a PS-BE scheme is atk* outsider *recipient private.*

Remark 2. *For a more relaxed notion of recipient privacy, i.e.,* outsider *recipient private, it suffices for atk* $\in \{CPA, CCA\}$*, that the OSN PS-BE scheme is correct, the* `PKE` *scheme is atk-secure and atk-key private, and the* $\mathsf{E}(\cdot)$ *is a semantically secure authenticated encryption scheme.*

Proof Sketch: The confidentiality, integrity, and outsider recipient anonymity hold as a consequence of the security of the underlying encryption blocks. In particular, the symmetric key encryption scheme is semantically secure, and the session key can only be obtained if the recipient holds the corresponding secret key $sk_{\mathtt{id}}$, assuming the public key scheme is IND-CPA or IND-CCA secure. Regarding recipient privacy, according to Theorem 2 a OSN oANO-PS-scheme is recipient privacy if the underlying constructions fulfill certain requirements. Specifically, the public key scheme used is required to be CPA or CCA key private, which on the case of the OpenPGP versions of Elgamal [83] and RSA-OAEP [94] are both CPA key private. For obtain CCA, a more CCA-secure scheme, such as Cramer and Shoup [62]. However, if a authenticated symmetric encryption scheme is used the PS-BE scheme provides the more relax notion of outsider key privacy (Remark 2), whereas the construction presented before provides full recipient privacy relying on the fresh digital signature keys from a unforgeable signature scheme [36].

Key Management. Although public-keys can be exchanged over a private channel, it could also be publicly or semi-publicly stored on the OSN, for example, with QR codes. This allows all members in $\mathcal{R}$ to automatically retrieve the keys. However, the process of distributing public keys requires verification. This can be done by out-of-band verification of the public key fingerprint, or via the Web-of-Trust approach extension to the OSN scenario, as proposed by Bischel *et al.* [29]. In addition, similar key storage services allow public key storage, while verification is enhanced with the extra usage of social media, such as Keybase.[1] With recent adoption of the Google end-to-end encryption project, the verification process could be executed using certificate transparency [137], or using a decentralized approach like DNS block chains [183]. As access control is selectively defined per content, the revocation

[1] Keybase.io: `https://keybase.io/`

process of groups members requires the removal of the revoked user public key, and is only valid for future content. Nevertheless, we assume that it is hard to protect content from malicious authorized recipients, who save, store, and broadcast the content. Otherwise, re-encryption of the full content is required.

Implementation. The PS-BE scheme is implemented using the Java Bouncycastle library,[2] and is the default PS-scheme implemented in Scramble (see. Appendix A). The overhead time for publishing, and retrieving a simple content message is illustrated in Table 6.1. The results demonstrate the very limited overhead on the viewer side. This is the consequence of re-using randomness, and placing anonymous placeholders similarly to Barth *et al.* [16].

Table 6.1: Overhead time (*msec*) of the oANO-PS-IBE scheme for varying sizes of the recipient set $\mathcal{S}$.

$\lvert\mathcal{S}\rvert$	PS-BE.Publish	PS-BE.Retrieve
1	65 *msec*	11 *msec*
10	73 *msec*	20 *msec*
15	91 *msec*	25 *msec*
50	125 *msec*	28 *msec*
100	203 *msec*	33 *msec*

6.5 Identity-Based PS scheme

In this section, we show how to obtain a PS-scheme from any Identity-Based Encryption (IBE) scheme, which can be seen as an extension of the OSN PS-BE scheme. As mentioned in Chapter 4, the IBE scheme generally requires a trusted PKG, which can be mitigated if the master secret is divided among multiple PKGs, following a Distributed Key Generation (DKG) [163] protocol based on Verifiable Secret Sharing (VSS) [55]. The multiple PKG setting could be supported, and maintained by several OSNs, considering that collaboration between competing OSN providers is a difficult task, and opposite to their business model. As many OSN users are represented on different OSNs, they can potentially abuse this fact for verification and authentication to a PKG, and to handle multiple identities. Figure 6.2 depicts an overview of the proposed model, in which users authenticate to t-PKGs of their choice to retrieve private keys. Due to the use of multiple PKGs this scheme requires extensions to the

[2]Bouncycastle: `http://bouncycastle.org/`

initial model. In particular, no more than t PKGs can collaborate, and the authentication to the PKGs requires to be performed under an authenticated channel, such as TLS, e.g., using a token similar to Facebook OAuth. For a stronger adversarial model these providers should operate under different jurisdictions to avoid coercion from the government to reveal their shares. For instance, Twitter (US), Spotify (Sweden/UK), Shazam (UK), SoundCloud (Germany), or Privalia (Spain). Thereby, overcoming more powerful adversaries with the power to affect at least t-PKGs by means of legal measures [147]. An analysis of the security provided by a trans-jurisdictional distribution is beyond the scope of this thesis.

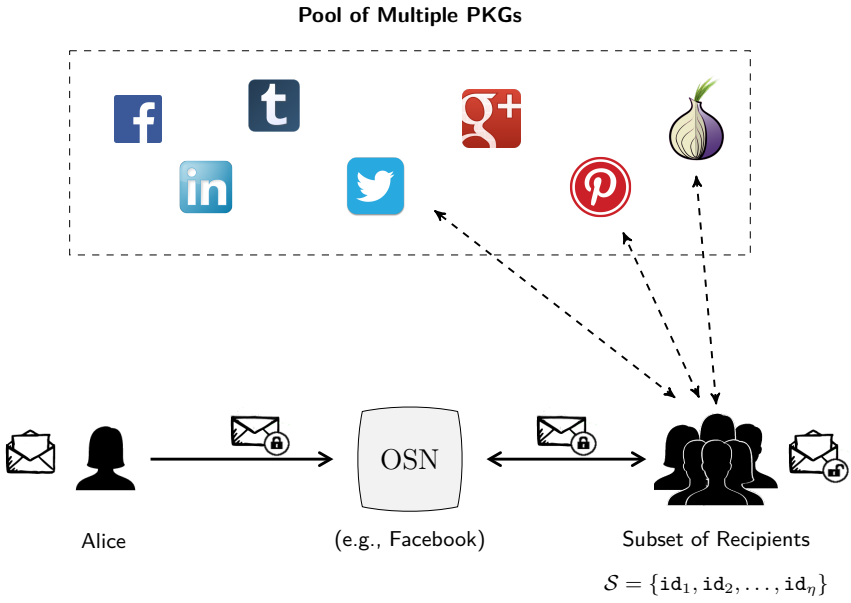


Figure 6.2: Multiple (n, t)-PKG IBE for OSNs overview, for a message $\mathtt{m}$ published for the set $\mathcal{S}$ for $t = 3$.

PS-IBE Scheme. This scheme proposes a solution based on the IBE scheme from Boneh *et al.* [31], and a relaxed version of the broadcast scheme from Libert *et al.* [140]. Moreover, it relies on the DKG protocol described by Pedersen [163] to bootstrap multiple PKGs. Thus, our oANO-IBE-PS scheme Π for OSNs is composed of four randomized algorithms, as follows.

$\texttt{Setup}(\lambda, t, n)$: Outputs the public *params* of the system with respect to the security parameter λ, a list of available PKGs $\Gamma = \{\texttt{PKG}_0, \ldots, \texttt{PKG}_n\}$, such that $|\Gamma| = n$, for the threshold t.

1. On input of security parameter λ generate a prime q, two groups $\mathbb{G}_1, \mathbb{G}_2$ of order q satisfying the BDH assumption, and an admissible bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. Choose random generators $P \in \mathbb{G}_1$, and $Q \in \mathbb{G}_2$.
2. Choose the hash functions $\texttt{H}_1 : \{0,1\}^* \to \mathbb{G}_1$, $\texttt{H}_2 : \mathbb{G}_T \to \{0,1\}^l$, $\texttt{H}_3 : \{0,1\}^l \times \{0,1\}^l \to \mathbb{Z}_q^*$, and $\texttt{H}_4 : \{0,1\}^l \to \{0,1\}^l$, modeled as random oracles.
3. Each $\texttt{PKG}_j \in \Gamma$ generates $n-1$ shares σ_{jv} of a Pedersen VSS scheme by executing `DKG.Setup`, and redistributing the $n-1$ shares σ_{jv} with the other v PKGs.
4. $\texttt{PKG}_j$ publishes $P_{pub}^{(j)} = s_j P$, s.t., $s_j = \sum_{v=1}^{n} \sigma_{jv}$.
5. Select a semantically secure authenticated $\langle \texttt{C} \parallel \texttt{T} \rangle \leftarrow \mathsf{E}(\cdot), \mathsf{D}(\cdot)$.

The master secret key $msk = \sum_{j \in \Psi} b_j s_j$ for $b_j = \prod_{z \in \Psi} \frac{z}{z-j}$ cannot be retrieved unless a subset $\Psi \subseteq \Gamma$ is of size at least t, s.t., $|\Psi| \geq t$. The following parameters are published publicly:

$$params = \{p, q, \mathbb{G}_1, \mathbb{G}_2, e, P, Q, \texttt{H}_1, \texttt{H}_2, \texttt{H}_3, \texttt{H}_4, t, n, P_{pub}^{(0)}, \ldots, P_{pub}^{(n)}\}$$

$\texttt{KeyGen}(\Psi = \{\texttt{PKG}_0, \ldots, \texttt{PKG}_t\}, \texttt{id}_i)$: On input of a user $\texttt{id}_i$ the subset Ψ of size t of PKG servers, generates a valid private key for $\texttt{id}_i$.

1. User with identifier $\texttt{id}_i$, authenticates to a subset Ψ, s.t., $|\Psi| \geq t$, or all PKGs and sends $\texttt{id}_i$.
2. Each $\texttt{PKG}_j \in \Psi$ determines the respective secret share s_j by computing $Q_{\texttt{id}_i} = \texttt{H}_1(\texttt{id}_i)$, and $Q_{priv,\texttt{id}_i}^{(j)} = s_j Q_{\texttt{id}_i}$.
3. The user $\texttt{id}_i$ computes the shared public parameter P_{pub} using the Lagrange coefficients b_j as follows:

$$P_{pub} = \sum_{j \in \Psi} b_j P_{pub}^{(j)} \quad \text{for} \quad b_j = \prod_{z \in \Psi} \frac{z}{z-j}$$

4. All PKGs in Ψ return $Q_{priv,\texttt{id}_i}^{(j)}$ to the corresponding user $\texttt{id}_i$ over a secure channel.
5. Each user verifies for each $Q_{priv,\texttt{id}_i}^{(j)}$ value whether,

$$e\left(Q_{priv,\texttt{id}_i}^{(j)}, P\right) \stackrel{?}{=} e\left(Q_{\texttt{id}_i}, P_{pub}^{(j)}\right)$$

Finally, the user with $\texttt{id}_i$ calculates the associated private key $sk_{\texttt{id}_i}$ using the Lagrange coefficients b_j as follows:

$$sk_{\texttt{id}_i} = \sum_{j \in \Psi} b_j Q_{priv,\texttt{id}_i}^{(j)}$$

In this way, no user nor PKG learns the master key msk of the system. In fact, an adversary is required to corrupt at least t or more parties to reconstruct msk. This algorithm combines DKG.Reconstruct, IBE.Extract and BE.KeyGen algorithms.

Publish($params, \mathcal{S}, \mathtt{m}$): Takes the message m, the subset $\mathcal{S}$ of size η and the public parameters $params$, output a broadcast message C.

1. Generate a random symmetric session key $\mathsf{k} \leftarrow \{0,1\}^l$.
2. Choose a random value $\rho \in \{0,1\}^l$ and compute r as a hash of concatenated values $r = \mathtt{H}_3(\rho, \mathsf{k})$
3. For each recipient $\mathtt{id}_i \in \mathcal{S}$, compute the ciphertext, running the IBE.Encrypt algorithm, as follows.

$$w_i = \rho \oplus \mathtt{H}_2(g_{\mathtt{id}_i}^r) \qquad \text{where} \qquad g_{\mathtt{id}_i} = e\left(Q_{\mathtt{id}_i}, P_{pub}\right) \in \mathbb{G}_T$$

4. Let W be a random permutation of w_i, $v \leftarrow k \oplus \mathtt{H}_4(\rho)$, and $U \leftarrow rP$, then the authenticated data c_1 is computed as,

$$c_1 \quad = \quad \{U \parallel v \parallel W\} \text{ s.t. } W = \{w_1 \parallel w_2 \parallel \ldots \parallel w_{|\mathcal{S}|}\}$$

5. Apply authenticated symmetric encryption on M, the concatenation of the intended recipient set $\mathcal{S}$ and the plaintext message m, such that $\mathtt{M} = (\mathtt{m} \parallel \mathcal{S})$. (BE.Encrypt)

$$\langle c_2, \mathtt{T} \rangle \leftarrow \mathsf{E}_{\mathsf{k}}(\mathtt{M})$$

6. Publish $\mathtt{C} = \{c_1 \parallel c_2 \parallel \mathtt{T}\}$ on the OSN.

Retrieve($params, sk_{\mathtt{id}}, \mathtt{C}$): on input of the broadcast message C and the private key $sk_{\mathtt{id}}$ of user $\mathtt{id}_i$, reconstruct the plaintext message m. This algorithm comprises the {IBE,BE}.Decrypt algorithms. For each $w_i \in W$:

1. Compute $w_i \oplus \mathtt{H}_2(e(sk_{\mathtt{id}}, U)) = \rho$ for $sk_{\mathtt{id}}$, and $v \oplus \mathtt{H}_4(\rho) = \mathsf{k}$
2. Set $r = \mathtt{H}_3(\rho, \mathsf{k})$. Verify $U \stackrel{?}{=} rP$. If the check fails, try next w_i, and return to 1.
3. Retrieve $\langle \mathtt{M}, \mathtt{T}' \rangle \leftarrow \mathsf{D}_{\mathsf{k}}(c_2)$
4. Verify whether $\mathtt{T}' \stackrel{?}{=} \mathtt{T} \in \mathtt{C}$, and return m. Otherwise return $\perp$.

Correctness. The OSN oANO-PS scheme is correct if for every member $\mathtt{id}_i \in \mathcal{S}$, s.t., $sk_{\mathtt{id}} \leftarrow \mathtt{KeyGen}(\{\mathtt{PKG}_0, \ldots, \mathtt{PKG}_t\}, \mathtt{id}_i)$, then $\mathtt{m} = \mathtt{Retrieve}(\mathtt{params}, \mathtt{sk}_{\mathtt{id}}, \mathtt{Publish}(params, \mathcal{S}, \mathtt{m}))$.

1. Let $w_i = \rho \oplus \mathtt{H}_2(g_i^r)$, where $g_i^r = e(Q_{\mathtt{id}}, P_{pub})^r \in \mathbb{G}_T$, $P_{pub} = \sum_{j \in \Psi} b_j P_{pub}^{(j)}$, $Q_{priv,\mathtt{id}_i}^{(j)} = s_i Q_{\mathtt{id}_i}$, and $sk_{\mathtt{id}} = \sum_{j \in \Psi}(b_j s_i Q_{\mathtt{id}_i})$. Then:

$$\begin{aligned} w_i \oplus \mathtt{H}_2(e(sk_{\mathtt{id}}, U)) &= \rho \oplus \mathtt{H}_2(g_i^r) \oplus \mathtt{H}_2(e(sk_{\mathtt{id}}, rP)) \\ &= \rho \oplus \mathtt{H}_2(e(Q_{\mathtt{id}_i}, P_{pub})^r) \oplus \mathtt{H}_2(e(sk_{\mathtt{id}}, rP)) \\ &= \rho \end{aligned}$$

2. Let $v \oplus \mathtt{H}_4(\rho) = \mathsf{k} \oplus \mathtt{H}_4(\rho) \oplus \mathtt{H}_4(\rho) = \mathsf{k}$.
3. Retrieve $\mathtt{M}/\perp, \mathtt{T}' \leftarrow \mathsf{D}_{\mathsf{k}}(c_1)$.

Complexity. In terms of efficiency, users are required to decrypt w_i on average $|\mathcal{S}|/2$ times before obtaining the symmetric key k. The size complexity is linearly bounded to the size of the recipient set $\mathcal{S}$, i.e., $\mathcal{O}(\mathcal{S})$. In contrast, the complexity of key storage is minimal, requiring only the need to store the private keys, as the public keys of the users are represented by their public $\mathtt{id}$s, and the session key is encrypted with the content.

Security Analysis. As the OSN ANO-PS scheme consists of secure underlying key privacy IBE, and authentication encryption schemes, the semantic security follows directly.

Theorem 3. *Let $atk \in \{CPA, CCA\}$. If the OSN oANO-PS-IBE scheme is correct, the DKG protocol is secure such that no more than t-PKGs gets compromised, the IBE scheme is atk-secure and atk-key private, and the $\mathsf{E}(\cdot)$ is a secure authenticated encryption scheme. Then a PS-IBE scheme is atk* outsider *recipient private.*

Proof Sketch: The confidentiality, integrity, and outsider recipient anonymity hold as a consequence of the security of the underlying authenticated encryption scheme. In particular, the session key can only be obtained if the recipient holds the corresponding secret key $sk_{\mathtt{id}}$, assuming the IBE-scheme is also semantically secure, i.e., IND-CCA. Regarding recipient privacy, according to Theorem 3 a OSN oANO-PS-scheme is recipient privacy if the underlying constructions fulfill certain requirements. As shown by Boneh and Waters [31], the underlying IBE is semantically secure under an adaptive adversary. As demonstrated by Paterson and Srinivasan [162] an IBE scheme is CCA-key private, and PKG anonymous if its also IND-CCA secure. Hence, if the chosen authentication encryption scheme is semantically secure, e.g., AES-GCM, then we show that our scheme is recipient private. As the OSN oANO-PS scheme also shares $\mathcal{S}$ along with the message we conclude that the scheme is outsider-anonymous. However, as the ciphertext size increases linearly with the size of $\mathcal{S}$, a powerful

adversary may infer the cardinality of the set. The use of dummies (i.e., extra random w_i values) for padding will increase recipient privacy at the cost of ciphertext size for smaller $\mathcal{S}$. A user is able to detect malicious behavior of any PKG from the public commitments of the Pedersen VSS [163]. It is also required that at least t from n PKGs do not get compromised. In case the OSN providers would maintain the PKG infrastructure, one could rely on the assumption that direct business competitors do not collude nor get legally coerced. Furthermore, the authentication and identity verification to the different servers can be done via, for instance, an open id token. This token could be generated as a proof of identity by any of the OSN providers.

Key Management. In contrast to the other versions of PS-schemes, the IBE version requires very little to any effort for key distribution, while the public key (`id`) verification is bound to the OSN identity, along with authentication to the different PKGs. The DKG approach solves the key escrow issues that come with generic Identity-Based solutions. In contrast to classic public key infrastructure, if a public key is revoked, the user would no longer be able to use that identifier for encryption, e.g., Facebook ID. Therefore, to support revocation an expiration date is concatenated to the identifier [31], requiring an extra periodic key update process. Similarly to the PS-BE scheme, the access control rights are selected per content, thereby allowing group revocation to be represented by removal of the revoked user id. Similarly to PS-BE version, revocation is just applied to future content, providing no forward security.

Implementation. The PS-IBE scheme, and the PKG-servers are implemented using the multi-precision MIRACL library [175]. Table 6.2 illustrates the execution time for $\lambda = 256$ *bit*, among different set $\mathcal{S}$ sizes. In turn, we use BLS (Barreto-Lynn-Scott) curve [15] along with the ATE pairing [117] for the bilinear pairings, suitable for $\lambda = 256$ *bit*. However, BN (Barreto-Naehrig) curves could also be used. For the symmetric authenticated encryption we used AES-GCM [174], and SHA-256 for the hash functions. According to Ugander *et al.* [195] the average group size for users in Facebook, with friendship connections size $|\mathcal{R}| = 100$ is 15. Thus, the PS-IBE scheme presents a tolerable overhead on the viewers side for retrieving content.

6.6 Replying and Placing Comments

It is common on OSNs for users to post replies and comments to the previously shared content `m`. Since users in the recipient set $\mathcal{S}$ are able to reconstruct the symmetric session key `k`, it is possible to encrypt the new comment with

Table 6.2: Overhead time (*msec*) of the oANO-PS-IBE scheme for varying sizes of the recipient set $\mathcal{S}$.

$\lvert\mathcal{S}\rvert$	PS-IBE.Publish	PS-IBE.Retrieve
1	284.5 *msec*	275.4 *msec*
10	2564.5 *msec*	460.9 *msec*
15	3799.6 *msec*	560.6 *msec*
50	12300.5 *msec*	1237.8 *msec*
100	25867.7 *msec*	2260.2 *msec*

k. However, it is not advisable using the same key, hence a hash chain can be used, so that the first reply would be H(k), then H(H(k)). In this way, a conversation among users can be built, and new users can be added at the middle of the conversation just by receiving the respective hash value of the joint point without learning previously shared information. This is possible due to the one-way secure hash functions property, as it is infeasible for any adversary to reverse the hash and obtain a previous node of the chain. At the same time, for the recipient privacy property to hold, the comments requirements is twofold: using the same encrypted blob, or a new and different message. In fact, the recipient set anonymity property holds only for the first message, as the first reply (by OSN design) will give one of the recipient's identity. In turn, one could argue that other users not in $\mathcal{S}$ could comment, thus not revealing any identity from $\mathcal{S}$. However, this is considered to represent a very unlikely event.

6.7 Summary and Discussion

We now discuss the different PS-schemes and demonstrate the different scenarios where each PS-scheme would be more suitable, with full or no cooperation from the OSN provider. All PS-schemes implement *end-to-end encryption* on OSNs by virtue of delivering some trust in OSN providers. In particular, relying on the OSN solely for storage, high-availability of data, as well as some fractions of key management with the exception of key generation. In fact, providers hold full control of the communication channel, making impersonation and removal of delicate or controversial information a simple task. Hence, we examined the complexity differences among the PS-schemes along with the details, and challenges required for being applied onto centralized OSNs. We consider the OSN to operate under the honest-but-curious model, placing as limited trust on the OSN as possible while taking into account the current limitations of modern

Table 6.3: Overhead length, distribution cost, and storage for all PS-Schemes.

	pk			sk		$\mathtt{C}$
	Length	Distribution	Storage	Length	Storage	Length
PS-SK[1]	–	$\mathcal{O}(\mathcal{L})$	–	$\mathcal{O}(\lambda)$	$\mathcal{O}(\mathcal{R})$	$\mathcal{O}(\mathtt{m})$
PS-BE	$\mathcal{O}(\lambda)$	$\mathcal{O}(1)$	$\mathcal{O}(\mathcal{R})$	$\mathcal{O}(\lambda)$	$\mathcal{O}(1)$	$\mathcal{O}(\mathtt{m}+\mathcal{S})$
PS-IBE	$\mathcal{O}(\mathtt{id})$	$\mathcal{O}(1)$	$\mathcal{O}(1)$	$\mathcal{O}(\lambda)$	$\mathcal{O}(1)$	$\mathcal{O}(\mathtt{m}+\mathcal{S})$

[1] As a special case, sk represents $\mathtt{k}$, and pk distribution indicates $\mathtt{k}$ sharing.

OSNs. In addition, each scheme aims at delivering high efficiency during the decryption process on viewers the side. As encryption is a more cumbersome process, we consider that publishers are strongly motivated to protect their content, thus extra overhead is acceptable. Table 6.3 illustrates the complexity differences of key storage and ciphertext complexity.

Although the PS-SK scheme presents the best efficiency with respect to the ciphertext size as well as relatively low publish and retrieve complexity, it requires high key management storage complexity and requires a secret key distribution process. PS-BE and PS-IBE deliver extra access control properties, by allowing a fine grained access control rights enforcement per content item according to group definitions. While key distribution is eased by the use of public keys, key verification for PS-BE involves an extra step when compared to PS-IBE which is identity bound.

In this chapter, we presented three different privacy sharing schemes for sharing content while protecting privacy as confidentiality in today's OSNs. Aside from describing the technical aspects, challenges, and key management, we discussed the different security properties. Then, we discussed the differences among the schemes, as well as the introduced overhead, demonstrating the impact towards viewers is minimal for all cases.

7

Undetectable Communication

"Even if you're not doing anything wrong, you are being watched and recorded."
– Edward Snowden *(2012)*

In this chapter we formalize the concept of undetectable communication in Online Social Networks (OSNs), whereby unauthorized entities are unable to detect the existence of secret messages posted and exchanged by OSN users. To this end, we present a secure covert information sharing scheme that achieves undetectable communication in OSNs. In particular, we extend the schemes presented in Chapter 6 in order to achieve undetectability in the OSN setting. Finally, to support the applicability of our solutions we discuss the implementation challenges, and show that our solutions introduce a low overhead.

Publications.

[19] Beato, F., Cristofaro, E. D., and Rasmussen, K. B., Undetectable communication: The online social networks case. In *PST 2014* (Jul. 2014).

[20] Beato, F., Ion, I., Čapkun, S., Preneel, B., and Langheinrich, M., For some eyes only: protecting online information sharing. In *ACM CODASPY 2013* (Feb. 2013).

Contributions. Principal author together with Emiliano De Cristofaro, Kasper B. Rasmussen, and Iulia Ion.

Chapter Outline. This chapter makes several contributions, describing the work published in [19] and [20]. First, it formalizes the notion of undetectable communication in OSNs, taking into account the limitations of modern OSNs. We then propose a protocol that provably achieves undetectability in OSNs. Finally, we build and evaluate an open-source prototype.

7.1 Motivation

Aligned with their vast popularity Online Social Networks (OSNs) have become primary targets of tracking, profiling, as well as censorship and surveillance. These facts have been supported by the leaked documents by Snowden [204], and demonstrated by Chaabane *et al.* [1] for the Syrian case.

These worrisome issues motivate the need for effective techniques to protect user privacy in OSNs when transferring sensitive messages, as motivated in Chapter 2. While decentralized architectures have often been advocated as a privacy-respecting alternative, they often hinder reliability and real-time availability or require users to buy cloud storage for their data, e.g., Cachet [159], Vis-à-Vis [179], or Safebook [63] (see. Chapter 3). On the contrary, centralized OSNs support high-availability content dissemination to a large number of non-tech-savvy users. Arguably, centralized OSNs are here to stay and actively being used by hundred of millions of people around the world, thus we focus on technologies that can be deployed atop existing OSNs.

Internet users can protect themselves from surveillance using anonymous communications (e.g., through Tor [79]), so that actions performed online cannot be mapped to offline identities. However, modern OSNs require users to create and maintain a profile, thus, only pseudonymity—rather than anonymity—is actually feasible with respect to the OSN provider. Solutions as the ones presented in Chapter 6 hide sensitive social content from the potentially prying eyes of the OSN provider and/or surveying entities. Despite the fact that encryption often violates the terms of service, posting encrypted data actually draws even more attention on a user targeted by censorship and surveillance.

Traditionally, the process of transferring hidden messages without suspicion is achieved by applying steganography to images. Although the security of steganography has been intensely studied in the last several years [43, 120], to the best of our knowledge, there is very little work analyzing the notion of steganography in the specific context of OSNs.

To this end, this chapter formalizes the concept of undetectable communication in OSNs, so that unauthorized entities are kept oblivious of the existence of secret

messages posted and exchanged by users through OSNs. In addition, it presents a scheme for secure covert information sharing that achieves undetectable communication in OSNs. Several approaches aim at undetectability by assuming a setting where a cover object, e.g., an image, has enough entropy to embed a secret. However, not all OSNs fit into this setting, and many providers process published images by applying compression, resizing, or removing metadata, thus, image-based steganographic techniques are moot in the OSN setting. After defining two different system models, based on the amount of entropy available in the cover object (high versus low), we introduce concrete attacker models and present an information sharing scheme in OSNs with provable undetectability.

7.2 Model

This section introduces the system and adversarial models used throughout the rest of the chapter.

7.2.1 Undetectability in OSN

Let Alice be an OSN user willing to send a secret message `m` to another OSN user, Bob. We assume that Alice uses the OSN infrastructure and, optionally, some auxiliary out-of-band channel. Alice and Bob wish to protect the confidentiality of `m` and also hide its existence from the adversaries defined below. We also assume that Alice and Bob share a symmetric key `k`.

7.2.2 Adversarial Model

We consider as adversaries any entity attempting to break the undetectability and/or the confidentiality of the secret message `m` sent by Alice to Bob. In practice, there may be a few different adversarial entities, including the social network provider as well as a passive adversary monitoring Alice's and Bob's connection to the Internet.

As many social network providers rely on user data for targeted advertisement, data mining, marketing and sentimental analysis, financial and commercial interests often lead them to restrict the use of encryption mechanisms. This restriction motivates the need for undetectability in case users wish to share encrypted content. Restrictions on the use of encryption are also crucial in the presence of a surveying government attempting to systematically monitor its citizens; besides partially (or totally) monitoring users' traffic, governments

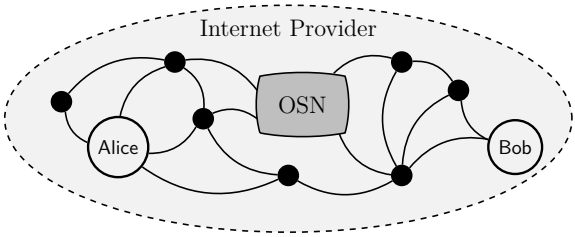


Figure 7.1: System Model. Alice posts content to the Online Social Network (OSN), so that Bob can retrieve it.

authorities can often obtain social networking data from OSN providers, e.g., through a subpoena or even warrant-less wiretapping, and, in extreme cases, coerce citizens to surrender encryption keys.

We consider an adversary such as the Online Social Network. This adversary may eavesdrop on all communication and access data routed to, or stored at, the OSN provider. This includes all data that has been posted to the OSN in the past, along with relationships, explicit or inferred, with other users of the same OSN.

7.3 Steganographic Models in OSNs

This section defines the concept of steganography in Online Social Networks (OSNs). We consider two possible models for undetectable communication. The first is a *high-entropy model* that captures the traditional notion of steganography, where a message is embedded inside a "normal looking" object (cover object), e.g., an image or a music file. The second is a *low-entropy model* which models the case where the cover object does not have enough entropy to contain the message.

7.3.1 High-Entropy Model

Our first model, which we denote as the *high-entropy* model, mirrors the traditional steganography setting where a message is embedded into a cover object using an embedding function Encode specific to the cover object. This process may, or may not require a key. The resulting stego-object is self-contained, i.e., nothing besides the stego-object (and possibly a key) is required

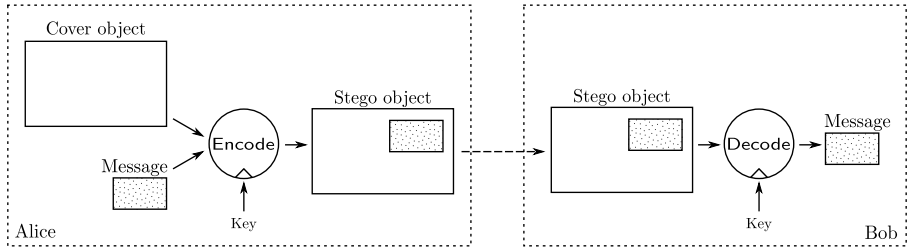


Figure 7.2: The high-entropy model. It involves three objects: a *cover object*, a *message*, and a *stego-object*. While the message is embedded in the stego-object, the adversary should not be able to determine this without access to the key.

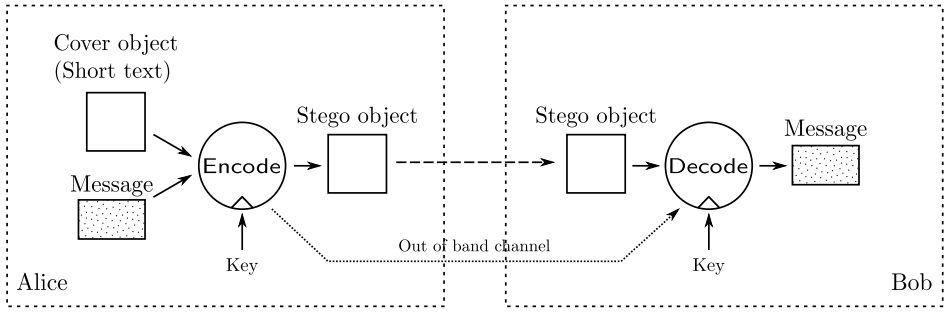


Figure 7.3: The low-entropy model. The stego-object, e.g., in the form of a *short text*, is public and is used as input to Decode (possibly along with a key) in order to recover the secret message.

to extract the message. Definition 12 formalizes the notion of high-entropy stego-systems, and Figure 7.2 illustrates the model.

Definition 12 (High-entropy stego-system)**.** *A high-entropy stego-system* $\mathbb{S}_h$ *consists of the following efficient algorithms.*

- $\mathsf{Setup}(\lambda)$: *is a probabilistic algorithm that takes as input, a security parameter* λ*, and returns a key* $\mathsf{k} \in \mathcal{K}$*.*
- $\mathsf{Encode}(\mathsf{k}, c, \mathtt{m})$: *is a probabilistic algorithm that takes as input a key* k*, a cover object* $c \in \mathcal{C}$*, and a (secret) message* $\mathtt{m} \in \{0,1\}^l$*, and returns a stego-object* $o \in \mathcal{O}$*.*
- $\mathsf{Decode}(\mathsf{k}, o)$: *is a deterministic algorithm that takes as input a key* k *and a stego-object* o*, and returns the embedded message* $\mathtt{m}$*.*

There must exist a polynomial $p(|c|)$*, such that:*

$$\forall \mathtt{m}, |\mathtt{m}| < p(|c|): \quad \mathsf{Decode}(\mathsf{k}, \mathsf{Encode}(\mathsf{k}, c, \mathtt{m})) = \mathtt{m}.$$

Security in the high-entropy model pertains to the unfeasibility of an attacker to distinguish between a cover object and a stego-object. Security definitions are presented in Section 7.3.3.

Naturally, the cover object must have enough entropy to contain the message. For example, if the cover object is a 2MB image and the message is a short 100-byte text, the image could be modified in such a way that the 100 bytes of text could be embedded, without noticeably altering the image [48, 51]. On the other hand, if the cover object does not have enough entropy to hide the message (e.g., a large image cannot be embedded in a short text), then another approach has to be used. We present such an approach below as the low-entropy model.

7.3.2 Low-Entropy Model

The *low-entropy* model is used if the cover object does not have enough entropy to contain the message. Without loss of generality, we consider the cover object as a short text, e.g., some text that could seamlessly be published on a social network such as, Twitter or Facebook. The low-entropy model is illustrated in Figure 7.3.

The cover object (e.g., some short text) is chosen to *represent* the secret message, rather than have the message encoded in it. The process of linking the stego-object to the message may, or may not, require a key. Therefore, the secret message itself must be sent to the recipient(s) using an out-of-band channel, since, by definition, the stego-object cannot contain it. Definition 13 formalizes the notion of low-entropy stego-system.

Definition 13 (Low-entropy stego-system)**.** *A low-entropy stego-system $\mathbb{S}_l$ consists of the following efficient algorithms:*

- $\mathsf{Setup}(\lambda)$: *is a probabilistic algorithm that, takes as input a security parameter λ, and returns a key $\mathsf{k} \in \mathcal{K}$.*
- $\mathsf{Encode}(\mathsf{k}, c, \mathtt{m})$: *is a probabilistic algorithm that takes as input a key k, a cover object $c \in \mathcal{C}^*$, and a secret message $\mathtt{m} \in \{0,1\}^l$, and returns a stego-object $o \in \mathcal{O}^*$ and a secret message $\mathtt{m}'$ (which may be identical to $\mathtt{m}$).*
- $\mathsf{Decode}(\mathsf{k}, o, \mathtt{m}')$ *is a deterministic algorithm that takes as input a key k, a stego-object o, and a secret $\mathtt{m}'$, and returns the message $\mathtt{m}$.*

So that, the following holds:

$$\forall \mathtt{m}: \quad (o, \mathtt{m}') = \mathsf{Encode}(\mathsf{k}, c, \mathtt{m}); \mathsf{Decode}(\mathsf{k}, o, \mathtt{m}') = \mathtt{m}.$$

Then, Bob uses the stego-object to determine the nature of the out-of-band channel and eventually get the secret message. For instance, consider the following example scenario:

> "Alice and Bob have agreed on two different physical sites for a dead-drop (an envelope with confidential information). They have also agreed on three keywords that Alice will use when the information is ready to be picked up, "Hello" for the first location, "Good morning" for the second location and "Good day" for abort pickup."

Prearranged keywords and locations represent the *key* in the low-entropy model, while the dead-drop represents the out-of-band channel. We foresee that our novel covert information sharing, aiming to achieve steganography in OSNs and presented in Section 7.4, follows the low-entropy model.

7.3.3 Security Definition

We now formalize the notion of steganographic security in OSNs. We aim to provide a generic definition that applies to both high- and low-entropy stego-systems. Thus, we start by defining the general notation for a stego-system.

Definition 14 (Stego-system)**.** *A stego-system* $\mathbb{S}$ *is either a high-entropy stego-system* $\mathbb{S}_h$ *or a low-entropy stego-system* $\mathbb{S}_l$*.*

To simplify the notation used in the rest of this section, we let $o(\mathtt{m})$ denote a cover object that simply encodes the message $\mathtt{m}$. Henceforth, for a high entropy stego-system $o(\mathtt{m}) = \mathsf{Encode}(\mathsf{k}, c, \mathtt{m})$, whereas, for a low-entropy stego-system $(o(\mathtt{m}), \mathtt{m}') = \mathsf{Encode}(\mathsf{k}, c, \mathtt{m})$.

To provide a notion of security of a stego-system $\mathbb{S}$, we first introduce the Game 3, as follows.

Game 3 ($\mathsf{IND\text{-}STEGO}_{\mathcal{A},\mathtt{Ch},\mathbb{S}}(\mathsf{k})$)**.** *The game between an adversary* $\mathcal{A}$ *and a challenger* $\mathtt{Ch}$ *proceeds as follows:*

1. *$\mathcal{A}$ is given access to an oracle that returns cover objects $\{c_1, \ldots, c_q\}$, which are taken from the set appropriate for the type of stego-system, i.e., $\mathcal{C}$ for a high-entropy stego-system and $\mathcal{C}^*$ for a low-entropy stego-system.*
2. *$\mathcal{A}$ outputs a message $\mathtt{m}$ and $\mathtt{Ch}$ returns either $c' = o(\mathtt{m})$ or a random stego-object c' with probability $\frac{1}{2}$.*
3. *Eventually, $\mathcal{A}$ outputs a bit b, where $b = 1$ if $\mathcal{A}$ believes that $c' = o(\mathtt{m})$ and $b = 0$ otherwise. The game outputs 1 iff $(c' = o(\mathtt{m}) \wedge b = 1) \vee (c' \neq o(\mathtt{m}) \wedge b = 0)$, i.e., if $\mathcal{A}$ could successfully guess the type of object returned by $\mathtt{Ch}$.*

We now define IND-STEGO security using Game 3 above:

Definition 15 (IND-STEGO security)**.** *A stego-system $\mathbb{S}$ is IND-STEGO-secure if there exists a negligible function ϵ, such that, for any probabilistic polynomial time adversary $\mathcal{A}$, it holds that:*

$$\mathtt{Pr}[\mathit{IND\text{-}STEGO}_{\mathcal{A},\mathtt{Ch},\mathbb{S}}(\lambda) = 1] \leq \frac{1}{2} + \epsilon.$$

7.4 Covert Information Sharing Scheme

In this section, we describe a low-entropy stenography scheme, that conforms to the low-entropy model defined in Section 8.4. The scheme represents the undetectability extension to the OSN-private sharing schemes (OSN-PS) proposed in Chapter 6, allowing Alice to communicate some secret information to Bob, via a OSN. For simplicity's sake, we first describe a general protocol with a single receiver (Bob), then we generalize to multiple receivers and groups. A generalized version of the scheme is illustrated in Figure 7.4. In addition to the OSN platform, our covert scheme utilizes two extra entities: a storage server and a mapping server, described as follows.

Storage Server (*srv*). represents any service that allows users to store and access data in the cloud, e.g., Dropbox, SugarSync. We assume that *srv* requires user registration prior to storage, and that each data item stored is accessible through a unique URL *url*. The *url* allows anyone to access and retrieve the associated data without authentication. However, only the account owner can modify and delete stored data. The communication with the *srv* is required to be over a secure connection, e.g., TLS.

Mapping service (*MS*). is a web-based service that stores short strings mapping $(index, value)$ pairs, such as URL shortener services, e.g., TinyURL, Bit.ly, or a specific Tor Hidden Service. Given an *index*, it allows anybody to

retrieve the value. The service does not accept duplicate indexes and places a restriction on the length of both the *index* and *value* strings, e.g., 30–140 characters long. We consider that stored entries do not expire and cannot be deleted. We also assume that MS accepts any anonymous requests to store and retrieve entries, and does not limit the number of entries a user can make.

7.4.1 Low-Entropy Information Sharing Scheme

Alice and Bob share a key k, used to derive the encryption key k_{ENC} and the $\mathtt{MAC}$-key k_{MAC}. Given some secret information $\mathtt{m}$ that Alice would like to communicate, Alice will first pick a short text *st*, independent of the secret, that will be published on the OSN, e.g., Facebook. The short text can be any arbitrary string that does not invoke suspicion. Alice creates the encryption key $\mathsf{k}_{ENC} = \mathtt{H}(\mathsf{k} \parallel 0)$ and the $\mathtt{MAC}$-key $\mathsf{k}_{MAC} = \mathtt{H}(\mathsf{k} \parallel 1)$, using a collision resistant hash function $\mathtt{H}(\cdot)$. She then uploads the secret $\mathtt{m}$, optionally encrypted $\mathsf{E}_{\mathsf{k}_{ENC}}(\mathtt{m})$, to the *srv*. At the same time, Alice uploads to the *MS* the *url* to the *srv* along with a mapping index, $index = \mathtt{MAC}_{\mathsf{k}_{MAC}}(st)$. This corresponds to Encode in our model. Note that the result *index* must be an uniformly distributed string, thus the $\mathtt{MAC}$ being a $\mathtt{PRF}$ is the appropriate tool to achieve this in the standard model. Then, the *MS* links the *index* to the *url*, and allows flexibility for the choice of the *srv*. In addition, if the *srv* provider supports the option to setup accounts, Alice can also create a temporary username and password and set an account as $(usr, pwd) = \mathtt{MAC}_{\mathsf{k}_{MAC}}(st \parallel index)$. These steps can all be done days before Alice actually intends to transmit the secret to Bob, if needed.

When Alice wishes to send the secret information to Bob, she publishes her chosen message *st* on Facebook. This will look to Facebook (and anyone else) as an innocent message that does not carry any additional information. Bob will $\mathtt{MAC}$ the text using the key shared with Alice to derive the $\mathtt{MAC}$-key and obtain the *index* that points to the *url* of the storage service by using the mapping service, such that $index = \mathtt{MAC}_{\mathsf{k}}(st)$. This corresponds to Decode in our model. Bob can then connect to *url* and retrieve the (possible encrypted) secret $\mathtt{m}$. Again the communication with the storage server and the mapping service must be over a secure connection. The covert information sharing scheme is summarized in Definition 16.

Definition 16 (Low-entropy covert sharing scheme for OSN)**.** *A low-entropy covert information sharing scheme for OSNs is a low-entropy stego-system* $\mathbb{S}_l^{OSN}$ *that consists of the following efficient algorithms:*

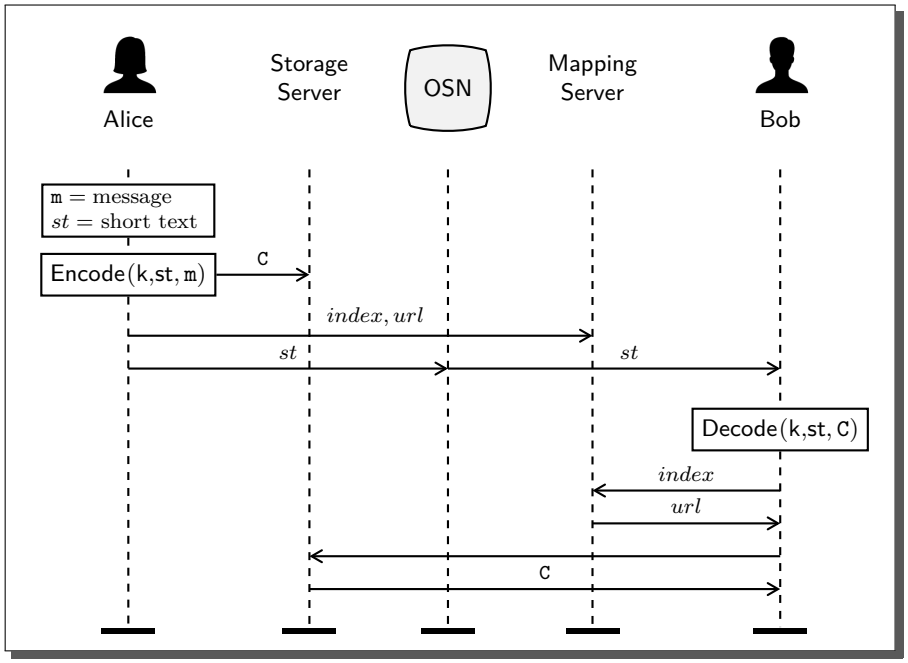


Figure 7.4: Low-entropy information sharing scheme. Alice derives an *index* from the MAC of a small cover text. She then uses a mapping service, e.g., *tinyurl* or a Tor hidden service, to create a mapping from that *index* to the *url* that contains the secret data. Alice posts the cover text on OSN (e.g., Facebook) where Bob reads it. Later, Bob can retrieve the secret by first deriving the index from the cover text and then obtaining the *url* for the secret message from the mapping server. All connections to the storage- and mapping server are assumed to be encrypted, e.g., using TLS.

Setup(λ): *According to the security parameter* λ *output* $\mathsf{k} \xleftarrow{r} \{0,1\}^{\lambda}$. *Using a collision resistant hash function* $\mathtt{H}(\cdot)$*, derive the keys for encryption* k_{ENC} *and the* MAC k_{MAC}*, s.t.,* $\mathsf{k}_{ENC} = \mathtt{H}(\mathsf{k} \parallel 0)$ *and* $\mathsf{k}_{MAC} = \mathtt{H}(\mathsf{k} \parallel 1)$.

Encode($\mathsf{k}, st, \mathtt{m}$): *Given the secret information* m*, pick a short text* st*, independent of the secret, and publish it to the OSN. Upload* m*, optionally encrypted* $\mathsf{E}_{\mathsf{k}_{ENC}}(\mathtt{m})$ *(or using* `Publish()` *algorithm from any OSN-PS schemes described in Chapter 6), to the storage service* srv*. Upload url to the mapping service (*MS*) with a mapping index,* $index = \mathtt{MAC}_{\mathsf{k}_{MAC}}(st)$.

Decode(k, st): *Retrieve* $index = \mathtt{MAC}_{\mathsf{k}_{MAC}}(st)$*, and subsequently url from* MS*. Connect to url, retrieve* C *from the storage service secret, and* m*for the valid* k*, i.e.,* $\mathsf{D}_{\mathsf{k}_{ENC}}(\mathtt{m})$*. Otherwise, return* $\perp$.

Security extension. Currently the *srv* location is part of the secret and thereby not secret for adversaries monitoring the *MS* service. This, consequently affects the protection of the information shared, and makes the system vulnerable to a possible collusion between the mapping service and the OSN. Hence, *srv* is required to be secret and flexible for changes. To this end, the Encode algorithm can be extended, so that Alice can encrypt *url* using a symmetric algorithm to obtain $elc = \mathsf{E}_{\mathsf{k}}(url)$. By using symmetric encryption to compute *elc* the computed ciphertext is smaller than public key encryption considering the length limitation enforced by *MS*. Therefore, Alice can use different *srv* per secret while keeping the *MS* oblivious of the *srv* location.

7.4.2 Group Communications

The multi-recipient information sharing problem has been discussed in Chapter 6, with different OSN-PS schemes approaches. For a simplified approach, Alice can simply create multiple accounts, one for each receiver, on the storage server. Each receiver can then independently retrieve a copy of the secret. For revocation, Alice simply deletes the account corresponding to the key she wishes to revoke, and users of that key can no longer access the secret. Users in possession of the revoked key cannot even tell that the message *st* posted to Facebook corresponds to a secret, since the storage server will not recognize the temporary (usr, pwd) generated using the revoked `MAC`-key.

7.4.3 Use of the OSN Infrastructure

Our covert information sharing scheme assumes users share a key, hence, at some point they must have established a secure, and possibly authenticated, channel. Therefore, one could question why these users would later communicate using the social network infrastructure rather than this secure channel. In some cases, direct communication might be the best option but there are several reasons, beyond convenience, why one might want to use a low-entropy steganographic approach instead.

First, and foremost, the direct secure channel might not be available all the time. Alice and Bob could have exchanged USB sticks with each others' cryptographic keys at some point in the past, but the information they want to communicate is only available now. Another reason could be that the secure channel is very low bandwidth and cannot be used to transfer the entire secret message.

Given that Alice and Bob share a key, they could also choose to communicate directly, e.g., via an encrypted email attachment, rather than relying on the

OSN. Again there are plenty of scenarios where this would be the preferred way but if Alice and Bob are trying to conceal the fact that they are transferring a potentially large and secret message, our protocol is suitable.

In addition, by using a mapping service our scheme allows flexibility with respect to the choice of the storage server per shared secret. Thereby, if a motivated adversary blocks this service then the user can always switch to a more privacy-friendly server.

7.4.4 Security Analysis

As mentioned earlier, our proposed scheme is an example of a low-entropy stego-system. We now prove that it is IND-STEGO-secure, by measuring the advantage an adversary $\mathcal{A}$ has in winning the IND-STEGO-game. We start to analyze the Online social network adversary, and then the case where on of the entities is under observation.

Without loss of generality we will use Facebook as an example of a social network adversary. Facebook has the ability to read any message posted by any user as well as monitor user behavior.

Even though Facebook has full access to the short text st, there is no way to check if st corresponds to any secret information, since st is specifically chosen independently of the secret. The short text need not have any specific structure or be about any specific topic. In fact, st could be a text that Alice would have posted anyway and therefore it is indistinguishable from any other message in $\mathtt{C}^*$, in fact Facebook does not even know which server a potential secret is stored on, since this information is part of the key. This means that Facebook can only win the IND-STEGO-game with probability 1/2, and the scheme is thus IND-STEGO-secure under a social network adversary.

In addition, if *at least one other party* is under observation, we need a more careful analysis. Without loss of generality we assume that Alice is the one under observation. The adversary ($\mathcal{A}$) will see that Alice connected to the storage server, i.e, $\mathcal{A}$ will know srv, which is part of the key. $\mathcal{A}$ can try to use this knowledge to get an advantage in the IND-STEGO-game.

The IND-STEGO-game proceeds according to Game 3 as follows.

1. $\mathcal{A}$ has access to all the users previous messages (as well as any arbitrary message).
2. $\mathcal{A}$ submits a secret message $\mathtt{m}_{\mathcal{A}}$ to $\mathtt{Ch}$, and $\mathtt{Ch}$ must now return a stego-object. This involves computing the $\mathtt{MAC}$ of an independently chosen short text $st_{\mathtt{Ch}}$, to obtain a username and password, then upload a secret (either

$\mathtt{m}_{\mathcal{A}}$ or random data, chosen with probability 1/2) to the storage server srv, and make the secret accessible using the newly created username and password. After that the stego-object $st_{\mathtt{Ch}}$ is returned to $\mathcal{A}$.
3. $\mathcal{A}$ must now guess if the secret on the storage server is $m_{\mathcal{A}}$ or not.

Having Alice under observation, $\mathcal{A}$ knows the location of the storage server srv, but assuming the connection between Alice and srv is secure, $\mathcal{A}$ learns nothing about the data exchanged between Alice and srv. $\mathcal{A}$ learns nothing by supplying an incorrect username and password to the storage server, and $\mathcal{A}$ cannot create the username and password without knowledge of the MAC-key. Guessing the username and password corresponds to guessing the output of the MAC, which can only be done with probability $1/2^n$, where n is the number of bits of the MAC output. Since $1/2^n$ is negligible the protocol is IND-STEGO-secure.

For the extended version of the scheme, where $elc = \mathsf{E}_{\mathsf{k}}(url)$ is posted instead of url on the mapping service. Then, an adversary monitoring the mapping service no longer learns the location of the server. In particular, assuming that $\mathsf{E}(\cdot)$ is a semantically secure symmetric encryption scheme, it is infeasible for $\mathcal{A}$ controlling MS and monitoring Alice to identify the location of the srv.

7.4.5 Social Indistinguishability

The nature of *undetectable* communication requires more than just confidentiality. It requires that no one is able to identify the cover messages as suspicious. We call this notion *Social Indistinguishability*. It is very difficult to quantify exactly what social indistinguishability means. For example, even if the cover message is completely unrelated to the secret topic, it can still be suspicious if it is unusual for an individual to express themselves in a certain way. Consider the following example: if Alice is usually interested in football but has never expressed any interest in politics, an adversary might have a good reason to suspect that a message containing political comments is a cover object for a secret message. However, close to elections, it might be perfectly normal for Alice to comment on political figures, even though she is not normally very politically active. Similar behavior may exist for any major event, such as, TV show, news story, and Internet mime.

We choose to model the notion of social indistinguishability in terms of the constraints a communication system places on the cover message. For a naive stego-system where the secret message is derived from the first letter of every word, the cover message must use words that start with that specific letter in order to convey the secret message. This will make it hard to choose a cover message that appears innocent, i.e., socially indistinguishable.

Our low entropy stenography scheme can use any cover text without any constraints. This means that the user is free to express himself in the exact way that he chooses, and that is appropriate to the context of the message. This freedom comes from the fact that the cover message itself does not actually contain the secret message, rather, it acts as an index to where the secret can be found. Since the cover message is known to Alice when she stores the secret message, she can change the storage location (path in a URL) to fit the cover message, rather than the other way around.

7.4.6 Traffic Analysis

Our security analysis has, thus far, set aside the issue of traffic analysis [67], although it can ostensibly help the adversary in the scenario where the storage server is under observation. Consider the following example: as the storage server is under observation, the adversary notices that Alice uploads 1,564 bytes of data. Later, Bob connects to the same server and downloads exactly 1,564 bytes of data. Even without considering their interaction on Facebook, it seems likely that the adversary can guess that there was a transfer of information between Alice and Bob.

Traffic analysis constitutes a traditional obstacle to privacy, e.g., for confidentiality [192] and anonymity [153], as well as censorship resistance [122]. To cope with it, a few solutions have been proposed both in the general Internet setting [208] and in OSNs [77]. We readily acknowledge that the security of our proposed scheme holds assuming traffic analysis resistance and leave, as part of future work, a thorough study of traffic analysis issues and countermeasures in the context of our covert information sharing scheme.

7.5 Implementation

To demonstrate the viability of our proposal, we implemented a proof-of-concept prototype of the covert information sharing scheme proposed in Section 7.4.[1]

In the description of the implementation, we distinguish between server- and client-side components, as depicted in Figure 7.5. The former is used to realize the out-of-band storage service, whereas, the latter runs as a browser extension on the user environment.

[1]Source of our implementations is freely available upon request.

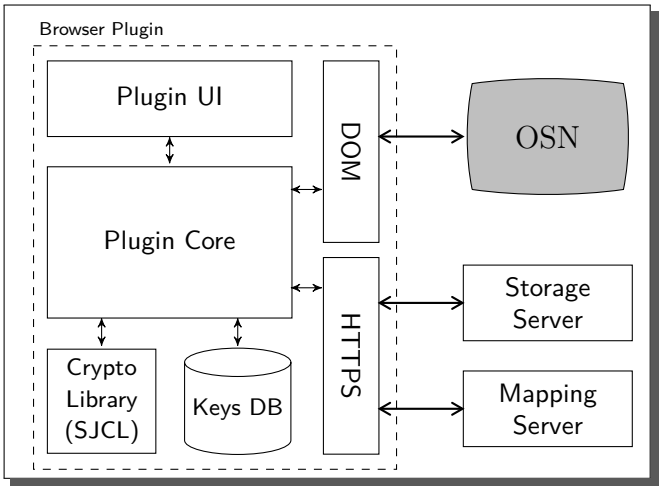


Figure 7.5: Covert scheme plugin block diagram. The user interacts with the plugin in three ways. The key `k`, the secret message `m` and the cover text st is communicated to the user via a dedicated *Plugin UI*. The plugin communicates with the OSN, e.g., Facebook, using the Document Object Model (DOM) in the browser, and communication with the storage- and mapping server is done using HTTPS.

Server-side. In our prototype, the server-side corresponds to a simple PHP back-end server and a MySQL database. It supports post and get actions:

Post: The storage server returns an url location when receives, from the user, the tuple (s, usr, pwd), i.e., the secret s (optionally encrypted), along with the username and password generated according to our scheme in Section 7.4, and stores such tuple in its database.

Get: On input (usr, pwd), the storage server returns s to the user.

Client-side. Our covert information sharing scheme is designed to work with existing OSNs, such as, Facebook, Twitter, Google+. Therefore, users will interface with the system via the regular OSN web site. Each operation in our scheme, such as Encode and Decode, is implemented as a web transaction, and users perform them from their web browser. There is no need to perform any operation outside the browser: our covert information sharing scheme only involves simple symmetric-key operations that can be executed, e.g., in Javascript using the Stanford Javascript Crypto Library (SJCL) [189]. We use AES-CMAC [186] for the `MAC` implementation and AES-CCM [206] for the authenticated symmetric encryption, as both are already available in SJCL.

However, we need a mechanism to seamlessly implement the interaction between

the user and the storage server, i.e., without requiring the user to run other software other than their browser or to leave the OSN website. To this end, we built a Firefox Extension (FE) that, when installed on the user's device, is used to post and read secret messages, alongside with the TinyURL website for the mapping service. Specifically, the Encode and Decode operations are as follows:

- Encode *(Post):* The user, Alice, selects a text area on the OSN website. The FE launches a dialog where the user inserts the secret message $\mathtt{m}$ and the short text st. In addition, the FE publishes st in the selected text area, and produces $(usr, pwd) = \mathtt{MAC}_{\mathtt{k}_{MAC}}(st)$. Subsequently, FE uploads the tuple $(\mathtt{C}, usr, pwd)$ automatically into the server that returns a url, where $\mathtt{C} = \mathsf{E}_{\mathtt{k}_{ENC}}(\mathtt{m})$. At the same time, the FE uploads the tuple $(index, url)$ to the TinyURL server, such that, $index = \mathtt{MAC}_{\mathtt{k}_{MAC}}(st)$ for $\mathtt{k}_{MAC} = \mathtt{H}(\mathtt{k} \parallel 1)$.
- Decode *(Get):* FE parses the messages on the OSN, and, for each message from Alice, produces $index = \mathtt{MAC}_{\mathtt{k}_{MAC}}(st)$. The $index$ is used to query the TinyURL service for the valid url. Then, the FE submits the tuple (usr, pwd) to url, that outputs c_A if there is a match, and $\perp$ otherwise. If $\mathtt{C}$ exists and the decryption result is $\mathtt{m}$, then the FE replaces, transparently, st with the secret message $\mathtt{m}$. Thus, this operation will have a $O(l)$ overhead.

The current prototype is compatible with Firefox 14+, but it could be easily ported to other browsers extensions, e.g., to Chrome, as it is written in simple Javascript. In terms of performance, the cryptographic operations, i.e., the MAC and AES implementations, take about 2 *msec*, while the communication latency existent between the client- and server-side presents limited complexity. Thus, while it only supports desktop browsers at the moment, it is perfectly suitable for resource-constrained devices, such as smartphones. This is crucial considering that a significant portion of users access OSNs via their mobile devices, for instance, almost 60% of Facebook users in October 2012 [131, 168].

7.6 Summary

Motivated by the limited effectiveness of privacy-enhancing technologies aiming at confidentiality and anonymity in Online Social Networks (OSNs) to provide undetectability, this chapter demonstrated a study of undetectability in OSNs. After formalizing the system and adversarial models, we presented a novel scheme for secure covert information sharing in OSNs as a valid extension to the

OSN-PS schemes in Chapter 6. Via an open-source prototype, we demonstrated that incurred additional computational costs are sufficiently low. Although inherently limited by the centralized nature of modern OSN architectures, as well as by the power of global government level adversaries, the attained degree of privacy constitutes an important step forward toward secure OSN communications.

8

Hiding Interactions

"I am not apt to follow blindly the lead of other men."
– Charles Darwin, *The Life and Letters (1887)*

Encrypting data does not directly protect the identity of users while browsing OSNs. Thereby, this chapter extends the problem of information sharing privacy from confidentiality and integrity to the behavior domain. In particular, for cases where adversaries can infer information from the user browsing behavior when monitoring the communication channel, i.e., the OSN. In order to address this issue, we devise a system denoted VirtualFriendship that allows users to browse OSNs while keeping their track anonymous. After formalizing the system, we discuss the challenges of the implementation, and evaluate it with a set of thorough experiments.

Publications.

[17] Beato, F., Conti, M., and Preneel, B., Friend in the Middle (FiM): Tackling De-Anonymization in Social Networks. In *IEEE SESOC 2013* (Mar. 2013).

[18] Beato, F., Conti, M., Preneel, B., and Vettore, D., Virtualfriendship: Hiding interactions on online social networks. In *IEEE CNS 2014* (Oct. 2014).

Contributions. Main author.

Chapter Outline. This chapter makes several contributions, describing the work published in [17] and [18]. After formalizing the system and adversarial model we devise a hybrid system, denoted VirtualFriendship, to protect users browsing activities in the OSN. We suggest cryptographic protocols that allow users to communicate and browse the OSN with minimal overhead. Finally, we build and evaluate an open-source prototype.

8.1 Motivation

With the large popularity and large number of users, Online Social Networks (OSNs) turned into the main communication channel, becoming, at the same time, the main source of information for advertisements, trackers, and profilers, either from government or third party business. Besides the importance of the content information, user browsing behavior contains valuable data to later infer sensitive information, containing, for instance, the main interests, hobbies, and the strength of the relationships of each user [13]. In fact, all user information can be used as auxiliary information to help track, and de-anonymize users within different systems [77]. As aforementioned in previous chapters, customizable privacy settings insufficiently protect the information shared in OSNs [89, 130], whereas cryptographic techniques aim at secrecy and authenticity of the content shared. Therefore, these techniques are a void protection for users browsing behavior on OSNs. Usually, users resort to anonymous networks, such as Tor [79], to protect browsing identity. While these solutions hinder the process of linking online behavior to an offline identity, and tracking users among different systems. Major OSNs require users to create and maintain a profile, requiring users to log on to profit from their advantages, and making anonymity by using anonymous networks infeasible. Still when using pseudonymity it suffices to a powerful adversary to hold access the user profile, shared information, and social interactions, in order to assemble and create a valid identity [9, 156]. Moreover, anonymity networks, like Tor, are often judged and subsequently blocked by global attackers [151, 207].

Motivated by the scarcity of privacy-preserving solutions to address anonymity in centralized OSNs, as discussed in Chapter 3, along with the impact browsing behavior holds towards the privacy of OSN users. In this chapter, we describe VirtualFriendship a solution aiming to guarantee anonymity while browsing centralized OSNs. Hence, we build the solution based on the social trust delivered by the user connections similarly to Danezis [66] and Drac [68], and introduce the concept of *routing friends*. In particular, we use a decentralized network composed of a subset of trust friendship connections, denoted *routing friends*, that route, deliver and forward low-latency content from inside the network. In

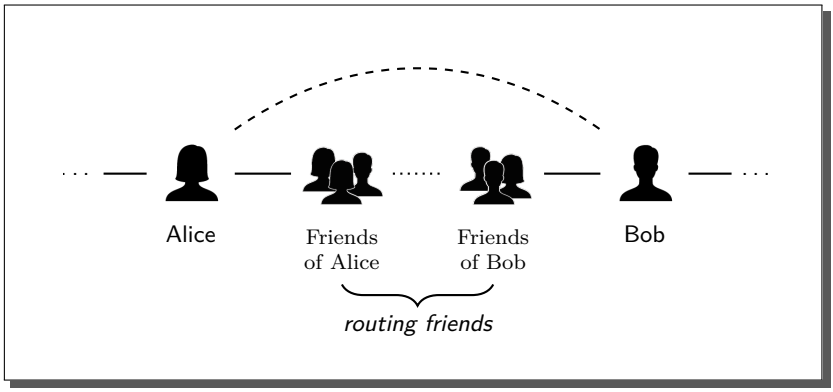


Figure 8.1: VirtualFriendship Routing Friends: Alice retrieves content related to Bob by relaying the traffic through a decentralized network composed of routing friends.

this way users are acquainted to browse and communicate privately towards the *routing friends* network, while enjoying all the benefits from centralized OSNs, essentially, storage, and connections. Figure 8.1 illustrates a high-level overview view of the system, where Alice use her friends along with Bob's friends to communicate with Bob. In contrast with other solutions, Virtualfriendship does not require re-design of current OSNs nor advocate the move to new privacy-friendly OSNs. Instead, it requires a fraction of users to use the VirtualFriendship. Despite the anonymity of each user is bounded to the number of friends, these sets are generally large enough on average.

8.2 Model

This section introduces the system and adversarial model along with the security and privacy requirements. Without loss of generality, we consider two users of any centralized[1] OSN – Alice and Bob; that browse information in the OSN, and subsequently share information with other users using the OSN infrastructure, while, optionally, leveraging external channels. Each user is represented in the OSN by a profile $\mathcal{P}$, manages a list of symmetric connections $\mathcal{R}$ and holds an asymmetric key pair (pk,sk). We assume that members in $\mathcal{R}$ are able to relay regular OSN-type traffic through a decentralized network composed of other members in the OSN. Such network is composed of social trusted connections $\Gamma \subset \mathcal{R}$. The communication between users and the routing friends network

[1] Although the changes for applying to a decentralized setting represent a straightforward exercise.

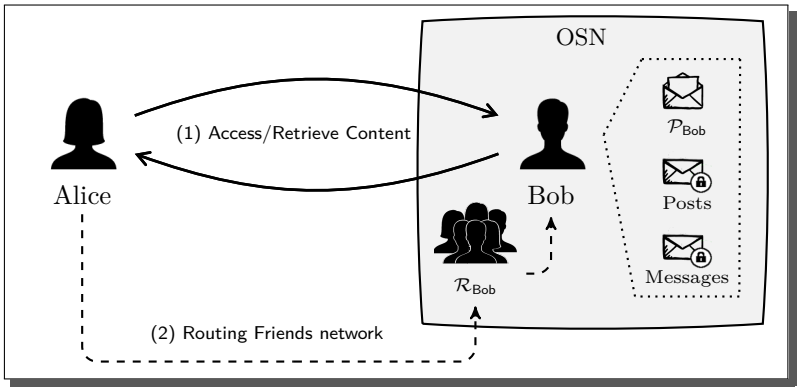


Figure 8.2: VirtualFriendship model with the routing Friends, whereby Alice instead of extracting $\mathcal{P}_{\text{Bob}}$ through the usual flow (1), uses the routing friend network (2), and thus, not revealing her identity.

is synchronous and assumed to be unobservable towards the OSN. Figure 8.2 illustrates a general overview of the model, whereby Alice makes use of the routing friend network to extract content from Bob.

8.2.1 Adversarial Model

Unlike previous chapters, we consider a passive adversary aiming at deriving information and breaking anonymity of users browsing OSN content, such as user visited pages, profiles, and comments. In particular, an adversary able to monitor, track and recover such browsing information is able to create a generic user profile, and, subsequently obtain sensitive information, for instance, main interests and the weight of their friendship connections [13]. Even though all content is shared encrypted using techniques described in previous chapters, we infer that such adversary adheres to the honest-but-curious model by following the protocol specification and not tampering with nor deleting content. In addition, we assume that the OSN does create fake nodes, i.e., using fake profiles.

8.2.2 Security and Privacy Requirements

In order to fulfill the security and privacy goals, we require that our system fulfills the following requirements:

– **Requester Anonymity.** A passive adversary monitoring the OSN traffic cannot tell the user identity based on his browsing actions, e.g., when requesting another user profile information. The requester anonymity is calculated based on the metrics proposed by Serjantov and Danezis [178] and Diaz *et al.* [76], as described by Definition 8 in Section 4.3.

– **Access Control Rights.** Only authorized users should be able to access content, providing a valid authorization token. In addition to the privacy settings provided by the OSN, each user should apply segregation rules as defined in Chapter 5 and 6, respectively.

– **Token Unforgeability.** Any unauthorized user to a specific content cannot forge the authentication token, and thus access the content. In particular, we endeavor that no adversary $\mathcal{A}$ can produce a valid authentication token with no-negligible probability.

– **Token Privacy.** The authorization token should not leak any information with respect to the user accessing a specific content, e.g., the user identity. In fact, the user verifying the token should just learn if the user requesting information is authorized.

– **Content Secrecy.** The secrecy and confidentiality of the exchanged content should be protected towards unauthorized users and the OSN provider. The authenticity of the content shared should also be protected to avoid, for instance, impersonation attacks.

– **Communication Unobservability.** The communication among two users should be unobservable. In particular, it is required to be hard for an adversary to detect that two users are exchanging or retrieving information, e.g., messages, profile information. We say that Alice and Bob are undetectable while communicating, if a bounded adversary $\mathcal{A}$ cannot distinguish if the two users communicating are in fact Alice and Bob with non negligible probability.

8.3 VirtualFriendship

VirtualFriendship is a hybrid architecture aiming to provide users with unobservable browsing and communicate experience on centralized OSNs. This is achieved by leveraging the communication through a privacy-friendly decentralized channel composed of social trusted connections. Users are, however, required to utilize a local server Λ_u to route traffic through a different channel,

optionally using an anonymous network (AN), e.g., Tor. In this way, users browse privately the OSN while profiting from the advantages of content availability and storage from the OSN.

8.3.1 Entities

The VirtualFriendship system is composed of three main entities: the users, the OSN, and, optionally an Anonymous Network (AN). For simplicity's sake, we consider users to be registered and represented in the OSN by a profile – $\mathcal{P}_{\mathsf{Alice}}$ and $\mathcal{P}_{\mathsf{Bob}}$. Further, each user is required to control a local server Λ distinguished by an unique identifier, for instance, the Facebook username. The operations on the system do not change for the case that Alice is an external independent user that does not hold an account in the OSN, but is a private connection in one or several groups $\mathcal{L}$ of Bob. Cryptographic keys are generated upon deployment, and the public parameters made available, for instance, on users profiles using QR code images. For the core of the system, we introduce the concept of *routing friends*, $\mathcal{F}$, represented by a direct trusted connection in $\mathcal{R}$ acting as intermediaries for actions performed by users in the OSNs. *Routing friends* receive and redirect requests through a decentralized $\mathcal{F}$-network formed by others $\mathcal{F}$, and optionally using a AN. Hence, the system considers two types of users:

– **Communication users.** Represent the users exchanging information, so that, for instance, Alice requests $\mathcal{P}_{\mathsf{Bob}}$; each user controls a single or multiple private list Γ of routing friends, associated to different segregation groups $\mathcal{L}$ for access control rights definition. In addition, $\mathcal{L}$ can hold users that are not present in $\mathcal{R}$. Therefore, $\Gamma_{\mathsf{Alice,Friends}} = \{\mathsf{fr} : \mathsf{fr} \in \mathcal{F}_{\mathsf{Alice},} \wedge \mathsf{fr} \subset \mathcal{L}^{\mathsf{Friends}}_{\mathsf{Alice}}\}$ for a access group with label "Friends".

– **Routing users/friends $\mathcal{F}$.** Denote regular OSN users that act as *entry* and *exit* points to the requested information on the OSN, e.g., $\mathcal{P}_{\mathsf{Bob}}$. In particular, routing friends are trusted connections of the user, such that $\mathcal{F} \subset \mathcal{R}$, acting as intermediaries forwarding requests between communication users. Is is assumed that such users have incentives to stay online and relay their friends traffic.

Entry-point. authenticate and retrieve the requested content from the OSN, e.g., $\mathcal{P}_{\mathsf{Bob}}$, on behalf of the requesting node, e.g., Alice. Entry points are required to have equal or higher access rights than requesting nodes, such that $(\mathcal{F}_{\mathsf{Bob},} \wedge \text{Alice}) \in \mathcal{L}_{\mathsf{Bob}}$.

Exit-point. performs similarly to Tor Bridges, and are optionally used for privacy enhancement. Such nodes operate the request from the requesting

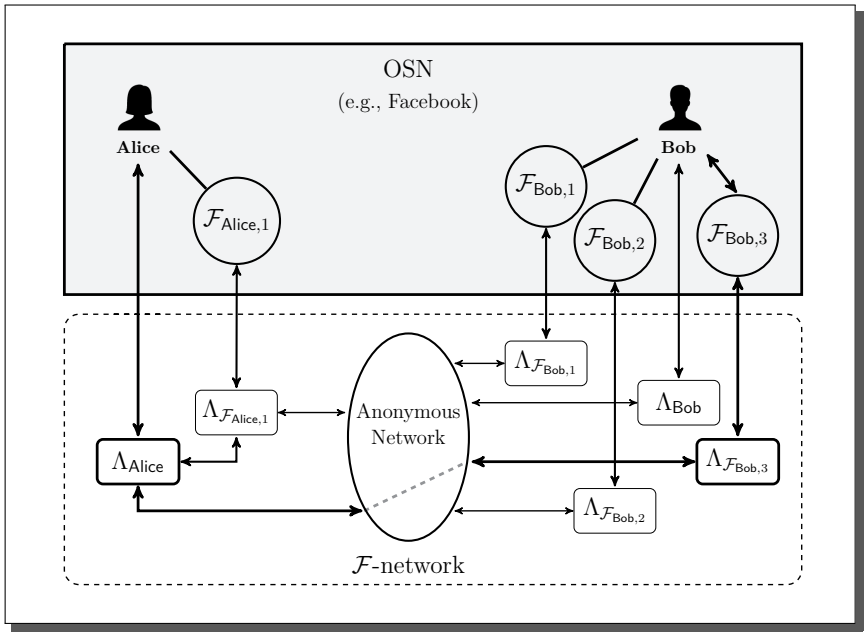


Figure 8.3: VirtualFriend System Overview: The connection between Alice and $\mathcal{F}_{\mathsf{Bob},3}$ is performed using Λ_{Alice} and $\Lambda_{\mathcal{F}_{\mathsf{Bob},3}}$ that are connected through a AN, e.g., Tor.

user (Alice) to another routing friend at the destination or the final node directly, $\mathcal{F}_{\mathsf{Bob}}$, or Bob, respectively.

The link between routing friends can be optionally through an anonymous network AN. Tunneling through an anonymous network AN provides additional attractive security and privacy features. Whereas the most marked is enhanced anonymity, it also offers encryption. For our system, the AN is used to provide anonymity to the content requester, for example, when Alice requests $\mathcal{P}_{\mathsf{Bob}}$, with respect to a compromised $\mathcal{F}$ and the OSN. Although, we assume the use of Tor [79] throughout the chapter, AN can be represented by any other centralized or decentralized network, such as Tarzan [90]. Also, the definition of a AN goes beyond this chapter and thesis. Figure 8.3 depicts an overview of our system, where Alice retrieves $\mathcal{P}_{\mathsf{Bob}}$ using $\mathcal{F}_{\mathsf{Bob},2}$ as the *entry* point, such that $\mathcal{F}_{\mathsf{Bob},2} \in \mathcal{R}_{\mathsf{Bob}}$ and $(\mathsf{Alice} \wedge \mathcal{F}_{\mathsf{Bob},2}) \in \mathcal{L}_{\mathsf{Bob}}$.

8.3.2 Protocols

This section describes the protocols of the system. For ease of exposition, we consider the simple scenario with only two users, such that Alice requests data related to Bob. Let $\tau \leftarrow \{0,1\}^{\lambda}$ be a random token used for authentication[2] for the security parameter λ, and $\mathcal{L}$ the access groups pre-defined by the user. $\texttt{Enc}_{pk}(\cdot)$ and $\texttt{Sign}_{sgk}(\cdot)$ represent a non-deterministic secure asymmetric encryption and an unforgeable digital signature [80] algorithms, respectively. $\mathsf{E_k}(\cdot)$ a semantically secure symmetric authenticated encryption, such as AES in CCM mode [206], or a dedicated scheme, such as AEGIS [209], and $\texttt{H}(\cdot)$ a collision resistant hash function. As aforementioned, Γ represents a subset of $\mathcal{L} \cap \mathcal{R}$ since not all users in $\mathcal{L}$ act as a routing friend, nor all in $\mathcal{R}$ have the same access rights. Inherently, users can specify a token τ per group $\mathcal{L}$, assuming the anonymity set is large enough, so that, the amount of users in the group is such that is hard to identify which one is accessing the content. In this case, the τ may be generated from a lower branch of a hash tree [149].

Initialization. To bootstrap the system, Alice and Bob need to become connected, in such a way that, Bob $\in \mathcal{R}_{\mathsf{Alice}}$ and Alice $\in \mathcal{R}_{\mathsf{Bob}}$. For cases where Alice is not represented in the OSN and subsequently not in $\mathcal{R}_{\mathsf{Bob}}$, then Alice is solely in $\mathcal{L}$. In the course of the connection establishment, Alice and Bob exchange an initial set of values $\texttt{I}$ composed of a list of routing friends Γ along with the authorization token τ associated to the $\mathcal{L}$, such that if $\mathcal{F} \in \Gamma$ then $(\mathsf{Alice} \wedge \mathcal{F}) \in \mathcal{L}$. Γ is, in fact, composed of the list of tuples: $(\mathcal{F}, \Lambda)$.

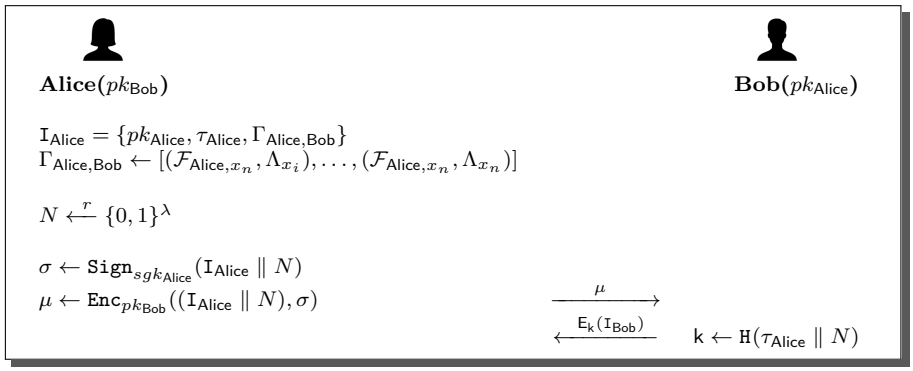


Figure 8.4: Initialization protocol between two users – Alice and Bob.

[2] Although the token τ is currently long lived token, it can become short lived by associating a timestamp.

The initialization protocol, as depicted in Figure 8.4, is a two-step protocol initiated by Alice. First, Alice encrypts (using pk_{Bob}) $\mathtt{I}_{\mathsf{Alice}}$ along with a random nonce N, and adds a signature for $\mathtt{I}_{\mathsf{Alice}}$. Upon receiving, Bob decrypts and verifies the authenticity of the content, and replies with the encryption (symmetric, for efficiency reasons) of $\mathtt{I}_{\mathsf{Bob}}$. Note that, if Bob specifies groups, such that, for example, Alice $\in \mathcal{L}^{\mathsf{Work}}_{\mathsf{Bob}}$, then all members in $\Gamma_{\mathsf{Alice,Bob}}$ are also in $\mathcal{L}^{\mathsf{Work}}_{\mathsf{Bob}}$. The overhead storage for Alice with respect to τ_{Alice} is linear with the number of groups, whereas the τ_{u_i}, such that $u_i \in \mathcal{R}_{\mathsf{Alice}}$ is linear with the size of $\mathcal{R}_{\mathsf{Alice}}$. For revocation, Alice is required to re-generate and distribute a new τ and shares with the connections with the affected connections, creating a linear communication overhead.

In practice this process is operated between Λ_{Alice} and Λ_{Bob}, using the unique identifiers as the local server addresses throughout the $\mathcal{F}$-network. However, as it is performed in encrypted format, it can be executed directly inside the OSN or using an out-of-band communication channel, e.g., email.

Accessing Content. In order to access Bob's content Alice follows the three-step protocol as illustrated in Figure 8.5. The three steps of the protocol are described as follows:

1. `Produce Request.` Alice performs the request using a trusted entry point, i.e., *routing friend* from $\mathcal{R}_{\mathsf{Alice}}$, or by reaching an exit point, such as $\mathcal{F}_{\mathsf{Bob}}$, from $\Gamma_{\mathsf{Bob,Alice}}$ or Bob himself. In both cases, $\mathcal{F}_{\mathsf{Alice},i}$ and $\mathcal{F}_{\mathsf{Bob},j}$ are chosen at random from $\Gamma_{\mathsf{Bob,Alice}}$ and $\mathcal{R}_{\mathsf{Alice}}$ respectively. Alice uses his Λ_{Alice} component to send the request of the form $(\mathcal{P}_{\mathsf{Bob}}, r, \psi_{Bob})$, where r is a random value generated per session, and ψ_{Bob} the `MAC` of r using the token τ_{Bob}.

2. `Authentication Request.` To access the content, Alice provides a proof of knowledge of τ_{Bob}. As $\mathcal{F}_{\mathsf{Bob},j}$ holds τ_{Bob} is able to produce the same `MAC` output as the one sent by Alice. If the authentication fails, $\Lambda_{\mathcal{F}_{\mathsf{Bob},j}}$ replies $\perp$ to Alice indicating a reject on accessing the content. Otherwise, $\Lambda_{\mathcal{F}_{\mathsf{Bob},j}}$ retrieves $\mathcal{P}_{\mathsf{Bob}}$, and encrypts it using a hash of the tuple: τ_{Alice}, r as the key. For requests where $\mathcal{L}$ is only composed of Alice, these can only be processed by Bob directly. Although the choice of using a `MAC` for authentication is mainly motivated by its efficiency, more secure zero-knowledge methods could also be applied at a cost of efficiency, comparatively to a Sigma protocol [133] or Anonymous Credentials [45, 47].

3. `Process Request.` Upon receiving the content, Λ_{Alice} decrypts and verifies μ_{Bob} using $\mathtt{H}(\tau_{\mathsf{Bob}} \parallel r)$ as the key. Hence, retrieving $\mathcal{P}_{\mathsf{Bob}}$ anonymously from the OSN. In fact, for the prying eyes of an adversary it is $\mathcal{F}_{\mathsf{Bob},j}$ that accesses and retrieves $\mathcal{P}_{\mathsf{Bob}}$ from the OSN.

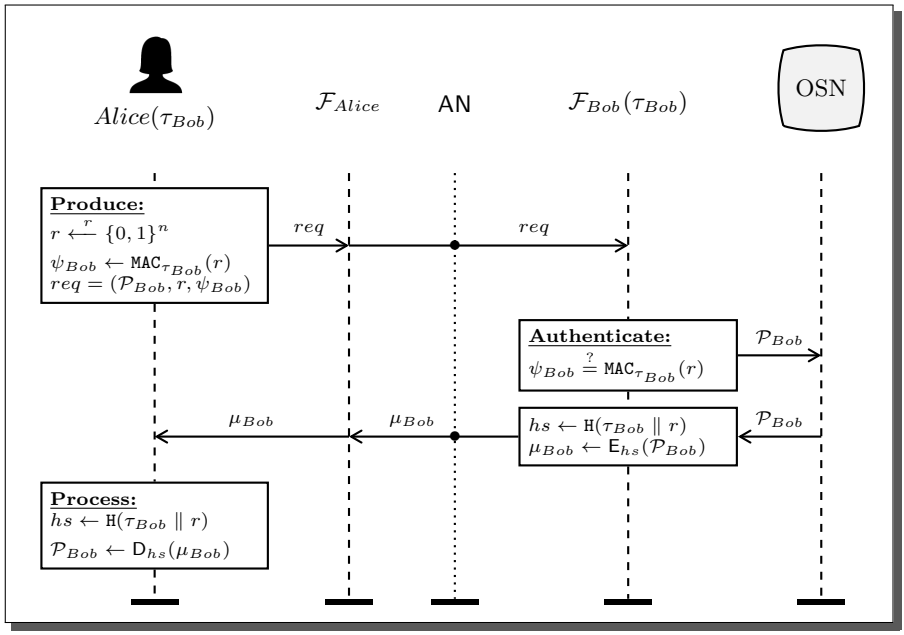


Figure 8.5: VirtualFriendship Protocol Overview. Alice request Bob's profile $\mathcal{P}_{\mathsf{Bob}}$ through the routing friends $\mathcal{F}_{\mathsf{Alice},i}$ and $\mathcal{F}_{\mathsf{Bob},j}$. Such that, $i \in \mathcal{R}_{\mathsf{Alice}}$ and $j \in \mathcal{R}_{\mathsf{Bob}}$.

Exchanging Private Messages. We now discuss how to privately send/read a message. For ease of exposition, we assume that there are only two participants – Alice and Bob, and later we discuss the scenario of multiple recipients. As the goal is to achieve unobservability of the communication towards the OSN, Alice and Bob establish a direct point to point secure communication channel, such as TLS [78]. In preference to use one of the routing friends $\mathcal{F}_{\mathsf{Bob},j}$ to forward the message. In practice, users are required to be online, and thus, Λ_{Alice} performs an initial check to verify if the peers are online before engaging the chat. A simplified version of the protocol is depicted in Figure 8.6, whereby Alice connects directly to Bob. For cases where Alice communicates with Bob through $\mathcal{F}_{\mathsf{Bob},j}$, an extra encapsulation is used using τ_{Bob}, i.e., $\mathsf{E}_{\tau_{\mathsf{Bob}}}(\mathtt{Enc}_{pk_{\mathsf{Bob}}}(\mathtt{m}_{(\mathsf{init},\sigma)}))$, similarly to Drac [68]. Subsequently, $\mathcal{F}_{\mathsf{Bob},j}$ authenticates Alice as aforementioned, decrypts the content using τ_{Bob}, and forwards the result to Bob. For each reply, Bob can use $\mathcal{F}_{\mathsf{Bob},j}$ or a different route using any $\mathcal{F}_{\mathsf{Alice},i}$, applying similar encapsulation. Due to the lack of knowledge of N and, possibly, τ_{Alice} or τ_{Bob}, the routing friends are not able to access the exchanged content. Although using different routing friends per message enhances security and privacy, it also introduces a high communication overhead. Since the cryptographic operations are handled

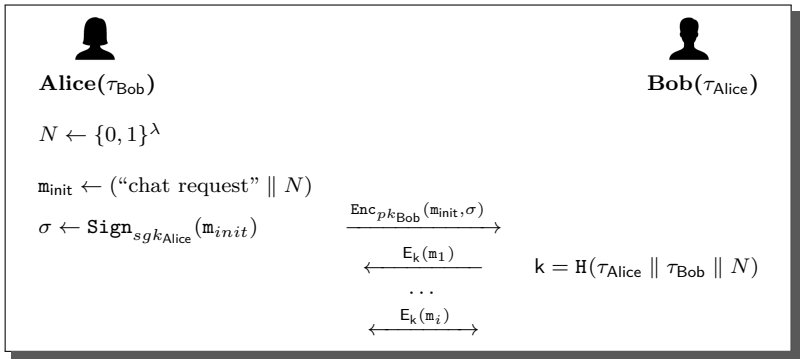


Figure 8.6: Exchange messages, where k represent the short lived session key and N a fresh random nonce.

by the components Λ_{Alice} and Λ_{Bob}, the authorization access procedure does not affect the usual usability flow. In order to obtain an extra property like forward security, users can resort to use the Off-the-record (OTR) protocol proposed by Borisov *et al.* [35], and security improved by Di Raimondo *et al.* [169]. The most notable difference is, however, the authenticated Diffie-Hellman key agreement at the place of $\mathtt{m}_{\mathsf{init}}$, allowing the generation of short lived key that, once discarded it is hard to recover the key and the messages encrypted with it.

In the case of multiple recipients, for instance, when Alice exchanges messages with $\mathcal{S} = \{\text{Bob, Charlie, Dave}\}$, then multiple secure channels should be used and a fresh secret generated using a group key agreement protocol [163, 145]. As an alternative the initiator could generate a short lived secret and use a OSN-PS scheme to exchange messages, as depicted in Chapter 6. This would largely increase the overhead of the communication. Otherwise, using the OTR group setting proposed by Liu *et al.* [141]. However, privacy often comes at a cost, and we argue that this overhead is an unavoidable privacy tradeoff.

Posting Comments. Posting comments usually takes multiple recipients. As mentioned earlier we aim to keep all user's identity anonymous towards outsiders, such that OSN is kept oblivious on who is involved in the interactions. Even though comments should be posted encrypted, Alice's identity would still be compromised. Also, we stress that using a routing friend $\mathcal{F}$ is also problematic as it may lead to impersonation and, subsequently, to social issues. Therefore, Alice utilizes Bob to place comments on her behalf, so that Alice sends Bob an encrypted message containing the message $\mathtt{m}$ and the intended recipient set $\mathcal{S}$. Then, Bob authenticates Alice and publishes the comment in his wall to the intended recipient set. In this way, Bob can verify the message before

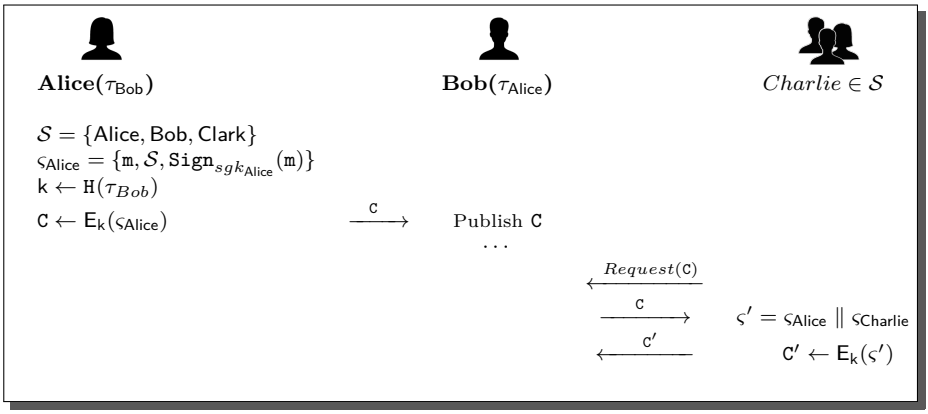


Figure 8.7: VirtualFriendship posting comments process overview.

being published, and subsequently perform extra edits and publish it. The simplified process is exemplified in Figure 8.7. Note that, to provide comments to the $\mathtt{m}$ published by Alice, recipients in $\mathcal{S}$ require to prove knowledge of τ_{Bob}. Consequently, Bob acts as a moderator and enforcer of comments for each message, allowing just members in $\mathcal{S}$ to publish replies to the message.

The protocol illustrated in Figure 8.7 does not, at present enforce any kind of access control and its secret k does not provide forward secrecy. However, this can be achieved by using one of the OSN-PS protocols depicted in Chapter 6.

8.3.3 Access Management

Different levels of access control or segregation of information presents an important property for OSN users privacy, as discussed in Chapter 5. Thereby, we now discuss the access rights and revocation to obtain forward security. Note that it is hard to control users with previous access to copy, and redistribute the shared content or the authorization token while holding access. In fact, we assume this event to unlikely occur, and in case of occurring such user is considered to break the social contract.

Currently, our system provides a single token for all connections or per group, and thus, requires entry points to have the same access rights as the requester. For instance, Bob defines the following lists $\mathcal{L}^{\mathsf{Work}}_{\mathsf{Bob}}$ and $\mathcal{L}^{\mathsf{Family}}_{\mathsf{Bob}}$, so that $\mathcal{L}^{\mathsf{Work}}_{\mathsf{Bob}} = \{\text{Alice, Clark, Dave}\}$, and $\mathcal{L}^{\mathsf{Family}}_{\mathsf{Bob}} = \{\text{Mom, Dad, Sister}\}$. This, however, increases overhead storage and complicates the revocation procedure. Also, users cannot enforce a more flexible access control per content, e.g., for cases where Alice

and Clark are in different groups and the content published should be accessed by just Alice and Clark. In addition, it does not provide transparency, as Alice is not aware of who else is in $\mathcal{L}_{\mathsf{Bob}}^{\mathsf{Work}}$ besides $\mathsf{Alice} \in \mathcal{S}_{Bob}$. The OSN-PS schemes from Chapter 6 present valid solutions for content privacy and for employing access control per content.

8.4 Security and Privacy Evaluation

We now turn to analyze the security and privacy resilience of our system under a passive adversary like the OSN. We demonstrate that such an adversary does not learn the interactions occurring, whether working independently or in collaboration with one of the routing friends. Furthermore, we show that a routing friend with no access to the content cannot authenticate himself to access unauthorized content by impersonating other.

Token Unforgeability. We consider that a cheating user cannot produce a valid request along with the proof of the authentication token τ, i.e., $\psi \leftarrow \mathtt{MAC}_\tau(r)$ for a given value r. This roughly means that the adversary $\mathcal{A}$ that can produce ψ can, with the same offer forge the valid output of a secure MAC. We consider that the MAC used is represented by a pseudo random function (PRF) [104], such that $\mathtt{PRF} : \mathcal{K} \times \{0,1\}^n \rightarrow \{0,1\}^t$, for $\tau \in \mathcal{K}$. In particular, we claim that an adversary $\mathcal{A}$, can win the following game with negligible probability:

1. Allow $\mathcal{A}$ to have access to a function f, such that $f : \{0,1\}^n \rightarrow \{0,1\}^t$.
2. $\mathcal{A}$ can repeatedly perform the following actions:
   a. Retrieve $\psi' \leftarrow f(r)$, for some $r \leftarrow \{0,1\}^n$.
   b. Request the content $\mathcal{P}$ to $\mathcal{F}$, with access rights to $\mathcal{P}$, by sending the tuple (ψ', r).
3. $\mathcal{A}$ wins the game iff outputs $\psi' = \psi \leftarrow \mathtt{MAC}_\tau(r)$.

Sketch: The probability that ψ constitute the output of $f(\cdot)$ for r as input is 2^{-t}, i.e., it is bounded to the size t of the output of the PRF. In our system, the process of proving access to ψ, should not reveal any information about the user requesting it. Cheating users can, however, abuse $\mathcal{F}$ with access to the content and produce q requests until receiving access. Thereby, the advantage of the adversary to forge the output ψ of the PRF for a given r, and, therefore win the game is as follows.

$$\mathtt{Pr}[(r,\psi) \leftarrow \mathcal{A}^{f(\cdot)}] \leq \frac{q}{2^t}.$$

Currently we do not protect against replay attacks, thus, the same tuple (r, ψ) can be used by a cheating user to authenticate to a specific content. Nevertheless, we consider exit nodes to be trusted, and entry nodes to have no motivational reason as they possess equal or higher access rights to the requested information.

Content Privacy. As content privacy ensures secrecy and authenticity of the content towards any unauthorized recipient. We assume that the encryption schemes used are semantically secure, thus, it is hard for an adversary to distinguish the encryption from random noise. Therefore, as all communications are encrypted, only authorized recipients are able to retrieve the content. The authenticity of the message is protected by strongly unforgeable signature scheme [80].

Communication Unobservability. It is hard for the OSN to detect that Alice and Bob are communicating, i.e., exchanging messages or placing comments. In fact, the communication is executed by Λ_{Alice} and Λ_{Bob} leveraged through different channels outside the prying eyes of the OSN, thus unobservable. Although stronger global adversaries, such as governments colluding with the Internet Service Providers and monitoring all the communication can infer that Alice and Bob are communicating, it is hard to decrypt the content of the communication.

Anonymity. Traffic analysis tools present a powerful tool to identify users communicating without knowledge of the content, even if information is exchanged encrypted anonymity assured. Therefore, for our system to achieve user anonymity, Alice should not be identified by the OSN when accessing, for instance, Bob's profile. To quantify the anonymity of our system, as mentioned before, we utilize the entropy metric described in Definition 9. Towards this means we classify two possible passive adversaries, with different capabilities: the OSN provider, and the exit point $\mathcal{F}$. Whilst we consider users OSN connections $\mathcal{R}$ to be publicly known, the trusted connections Γ are partially known by $\mathcal{F}$. In fact, it is up to the users to disclose the same set Γ among the $\mathcal{F}$. For simplicity of description we evaluate the anonymity under the scenario where Alice initiates a request to retrieve $\mathcal{P}_{\mathsf{Bob}}$, while using $\mathcal{F}_{\mathsf{Bob},}$ as the entry point.

Considering the OSN to be adversarial and with no prior knowledge that users are using our system, then the request from $\mathcal{F}_{\mathsf{Bob},}$ on behalf of Alice is indistinguishable from any other request from $\mathcal{F}_{\mathsf{Bob},}$ as $\mathcal{F}_{\mathsf{Bob},} \in \mathcal{R}_{\mathsf{Bob}}$. In contrast, for untrusted OSNs, the anonymity is dependent on the probability distribution p_i of each member in $\mathcal{R}_{\mathsf{Bob}}$ of being the requester. Assuming no previous knowledge, then $p_i = |\mathcal{R}_{\mathsf{Bob}}|^{-1}$, i.e., each connection has equal probability of requesting information. For the case where the adversary controls

$\mathcal{F}_{\mathsf{Bob},j}$, then, it would have more knowledge, as Γ represents a smaller anonymity set and may leak more information than $\mathcal{R}_{\mathsf{Bob}}$. If Bob shares the same Γ with all the $\mathcal{F}$, then $p_i = |\Gamma_{\mathsf{Bob}}|^{-1}$.

Ugander *et al.* [195] demonstrated that Facebook users have a median number of connections of 100. Hence, assuming Bob is a Facebook user with $|\mathcal{R}_{\mathsf{Bob}}| = 100$, then the maximum entropy value is $H = 6.6$ *bits*. Furthermore, users with 100 friends have an average degeneracy of 15, i.e., connections clusters, whereas for users with 500 connections it is about 53 [195]. Thereby, assuming that users with 100 connections have groups of size 15, i.e., for cases where $\mathcal{R}_{\mathsf{Bob}} = \mathcal{L}_{\mathsf{Bob}}$, then the maximum anonymity achieved towards a cheating $\mathcal{F}_{\mathsf{Bob},i}$s in average $H = 15 \times ((1/15) \cdot log_2(15))$, i.e., $H = 3.90$ *bits*.

Using the routing friend along with Tor provides a good level of anonymity, as the entry point does not learn the identity of the exit point nor the requester. Besides the extra security and privacy features offered by Tor, it also contains other issues. For instance, Johnson *et al.* [125] showed that Tor users are susceptible to realistic adversaries. On the case that the entry point of Tor is compromised, then the adversary can only infer that $\mathcal{F}_{\mathsf{Alice},}$ is making a request. However, even if such a powerful adversary deduces the system is in use it is hard to identify the link between $\mathcal{F}_{\mathsf{Alice},}$ and Alice. In fact, using $\mathcal{F}_{\mathsf{Alice},}$ the anonymity is bounded to the connections of $\mathcal{R}_{\mathsf{Alice}}$. The problem with respect to the Tor entry nodes has been explored, with several other solutions being presented [4]. We acknowledge that Tor can be blocked by the network or a more powerful adversary. Nevertheless, the communication between the requester and the exit point is assumed to be trusted and done by plain HTML, acting as a bridge to Tor.

In addition, alongside with the scalability of our system and the direct increase of routing friends, our system benefits with respect to privacy as the anonymity sets also get larger. Whereas we tackle several privacy and security issues on the social network, we stress the fact that like Tor we do offer a global end to end protection, such as timing and correlation attacks.

8.5 Discussion and Extensions

Although VirtualFriendship constitutes a hybrid privacy-enhanced extension for centralized OSNs, it is far from solving all existent privacy issues. Therefore, in this section, we discuss possible extensions with respect to authentication, extra actions and mobile environments.

Anonymous Credentials. Authentication represents an important functionality of our system. As described in the security analysis, our protocol does not provide forward secrecy. In fact, an attacker can replay the request although he cannot decrypt it. The authentication tokens provided (e.g., Alice provides τ_{Alice} to Bob) could be generated in collaboration, using, for example, authenticated Diffie-Hellman key agreement [118] or Sigma [133] protocols. However, such solution would not provide anonymity as each user would have an unique token and thus an unique identifier.

A different approach, yet less efficient and more complex is to use anonymous credentials [45, 47]. Such solution allows users to prove knowledge of attributes without revealing any other information by employing zero knowledge proofs [105]. For instance, Alice can prove to $\mathcal{F}_{\mathsf{Bob},j}$ or Bob himself that she is eligible to access the requested content without disclosing her identity.

To setup anonymous credentials into VirtualFriendship system mainly affects the initialization protocol, by increasing the roles of each user. In particular, Bob is required to act as an issuer, and issue a credential to each friend according to the respective access rights, e.g., $\mathsf{Alice} \in \mathcal{L}^{\mathsf{Friends}}_{\mathsf{Bob}}$. Consequently, Alice produces one-time tokens τ represented by the proof of knowledge of the credential issued by Bob per content requested, which is then verified by Bob or any $\mathcal{F}_{\mathsf{Bob},}$, acting as verifiers. In this way, Alice benefits from the nice privacy and security properties, such as anonymity and forward secrecy, during the system operation with an efficiency trade-off. In addition, by using accumulators revocation becomes more efficient.

Page and content "likes". Voting actions, just as the "like" action in Facebook, are simple to perform, and extensively used in OSNs. However, such actions are an important source of sensitive information for adversaries. In particular, such adversary can compute the weights and directly determine the strength of friendships and associate common interests [13]. Thus, it is hard to protect user's identity when such voting actions are performed. Whereas addressing this question is beyond the scope of this chapter, an interesting solution could be the use of e-voting techniques with double spending protection, e.g., [46]. Such solution requires an extra, somehow trusted, entity to act as a bank and issue and manage coins.

Mobile Extension. Aligned with the enormous growth on usage of mobile devices, like tablets, and smartphones, mobile users represent the majority of traffic in OSNs [131, 168]. Currently, our implementation is not compatible with mobile devices. However, the low overhead of VirtualFriendship makes it appropriate to such constrained devices, e.g., AES-CCM encryption takes about 50 *msec* on a smartphone. In fact, the mobile setting can represent an asset to

the online availability of routing friends. Nowadays such devices allow users to be constantly online, thereby, allowing users local server to be also continuously online.

Other Mix Networks. During this chapter we imply the, optional, use of an onion routing based approach for the anonymity network with the use of Tor, for delivering low-latency web-browsing efficiently. Yet at the cost of security against a global adversary model. In contrast, the use of high-latency approaches following Chaum solution [52], such as Mixmaster [152] or Mixminion [69], introduces delays on the communication while not requiring cover traffic and forces a particular sequence of nodes. Instead, using freedom networks [7] restricts the paths used creating thus a trade-off between anonymity and efficiency.

8.6 Implementation

To demonstrate the viability of our proposal, we implemented a proof-of-concept prototype as a Firefox plugin, denoted VF-App,[3] and tested it on Facebook. In this section, we describe the architecture, the implemented processes and the performance analysis of our implementation.

8.6.1 Architecture

The architecture of VF-App is illustrated in Figure 8.8, and is composed of two main components: a requester component (VF-Requester) and a routing component (VF-Router). Both components are embedded and run as an unique browser extension, and operate as follows:

VF-Requester. Manages the user interface, and interacts with the VF-Router to perform requests.

VF-Router. Runs as a local server relaying traffic in twofold: (1) as a client to forward VF-Requester requests through Tor using Vidalia;[4] and (2) as a server to relay, authenticate, and realize other users requests. The Facebook information is requested by means of a Facebook specific query (FQL)[5] along with a Facebook authentication token. Each local server currently communicates over port 8765, and is identifiable by an associated

[3]Source available at: https://sites.google.com/site/facebookvirtualfriendship/.
[4]https://www.torproject.org/projects/vidalia.html.en
[5]https://developers.facebook.com/docs/reference/fql/

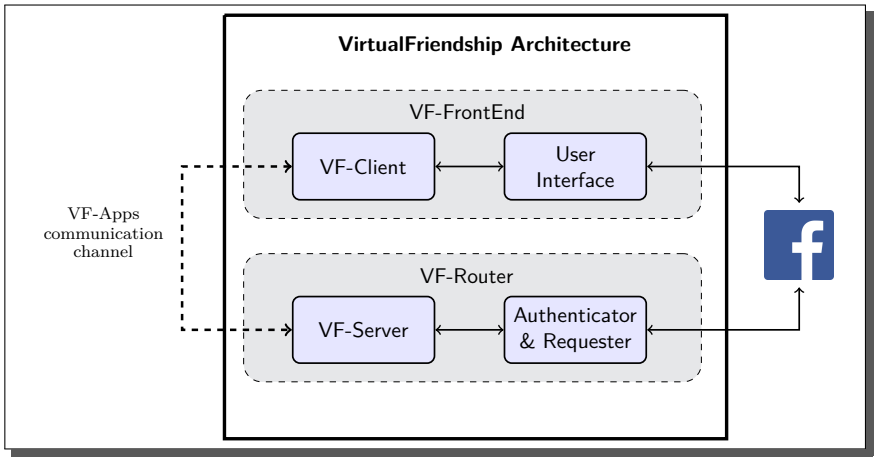


Figure 8.8: VirtualFriendship Firefox Extension Architecture Overview

web address, such as IP or domain name base addresses. This allows, for instance, Λ_{Bob} to be reached by Λ_{Alice}.

The current prototype is compatible with Firefox 14+, and since it is written in plain Javascript could be easily ported to other browsers, e.g., Chrome. Besides the easy installation process, VF-app requires Vidalia for tunneling through Tor. As Tor operates under SOCKS [138], we use Polipo[6] to convert plain HTTP requests into SOCKS. The cryptographic operations are executed using the Stanford Javascript Crypto Library (SJCL) [189].

8.6.2 Processes

Now we overview the VF-App implemented processes, mainly focusing on accessing content related to Bob in Facebook anonymously, e.g., $\mathcal{P}_{\mathsf{Bob}}$.

Bootstrap. To bootstrap the system users establish connections with their friends. Each user exchanges with other users the JSON file containing an initial set of information $\mathtt{I}$ composed by the token τ, the list of possible routing friends Γ, and a symmetric key. Currently, this process is done automatically via mail, however, it could be implemented via other offline channels, like USB flash drives. The list of friends is stored locally as a JSON object.

[6] http://www.pps.univ-paris-diderot.fr/~jch/software/polipo

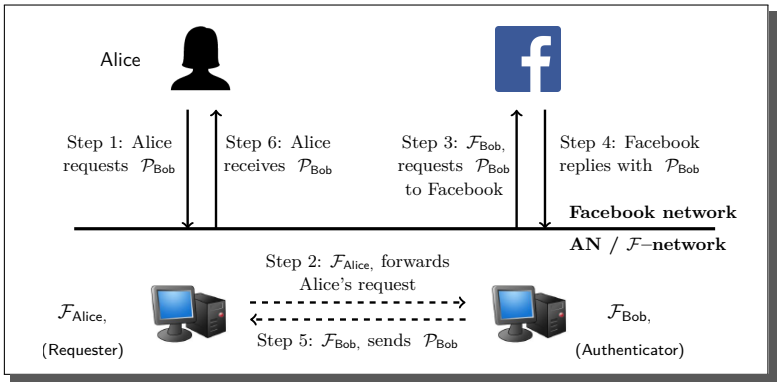


Figure 8.9: VirtualFriendship Application Information Request Process: Alice uses $\mathcal{F}_{\mathsf{Alice},}$ to request $\mathcal{P}_{\mathsf{Bob}}$, using τ_{Bob} to authenticate with $\mathcal{F}_{\mathsf{Bob},}$ and subsequently retrieve $\mathcal{P}_{\mathsf{Bob}}$.

Accessing Content. The process used by Alice to retrieve anonymously the profile of Bob follows the general protocol from Figure 8.5. The practical steps are depicted in Figure 8.9, and summarized as follows:

1. Chose at random a exit point $\mathcal{F}_{\mathsf{Bob},}$ from the list $\Gamma_{\mathsf{Alice,Bob}}$, and entry point $\mathcal{F}_{\mathsf{Alice},}$ from $\mathcal{R}_{\mathsf{Alice}}$. Produce a authentication proof $\varsigma \leftarrow \mathtt{MAC}_{\tau_{Bob}}(random)$ and attach to the request sent to $\mathcal{F}_{\mathsf{Alice},}$.
2. $\mathcal{F}_{\mathsf{Alice},}$ forwards Alice's request to $\mathcal{F}_{\mathsf{Bob},}$ using Tor.
3. $\mathcal{F}_{\mathsf{Bob},}$ receives the request, and verifies the authenticity of the request using τ_{Bob}. Then, $\mathcal{F}_{\mathsf{Bob},}$ as it signed in to Facebook, collects the Facebook token for authentication, and makes a FQL request.
4. $\mathcal{F}_{\mathsf{Bob},}$ processes the Facebook reply with the requested information, e.g., $\mathcal{P}_{\mathsf{Bob}}$.
5. $\mathcal{F}_{\mathsf{Bob},}$ encrypts $\mathcal{P}_{\mathsf{Bob}}$ using Bob's shared key, and forwards the encrypted result to $\mathcal{F}_{\mathsf{Alice},}$.
6. Finally, $\mathcal{F}_{\mathsf{Alice},}$ redirects the response to Alice, which is able to decrypt and access the requested information.

We underline that the first and last steps of the protocol are performed by the VF-Requester, while the remainder are executed by the VF-Router component, outside the OSN network. All actions are automated and transparent to the user, whereas the OSN provider is kept oblivious of the action request.

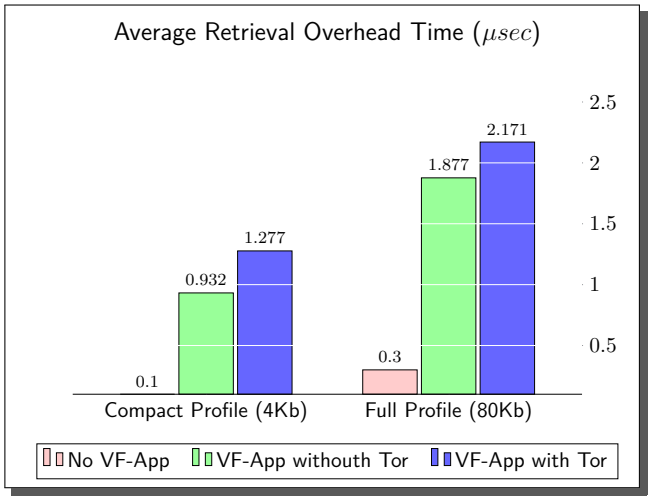


Figure 8.10: VirtualFriendship time overhead.

8.6.3 Performance

In order to analyze the practical usability and performance of our system, we have measured the two most costly factors: the cryptographic overhead added for token protection and authentication of the requested content; and the average communication overhead required for a profile request. We have used AES-CMAC [186] for the MAC implementation and AES-CCM [206] for authenticated symmetric encryption from the SJCL [189] library. The cryptographic overhead of the authentication process is represented by the `MAC` execution which takes about 2 *msec*, while the symmetric encryption of the full profile (approximately 80kB), takes about 10 *msec*. We then compared the overhead of extracting just the personal information with the process of extracting the full profile including recent timeline events, the communication overhead differences are illustrated in Figure 8.10. Our results show that there is a significant difference on performance, however, we stress that it represents a tolerable cost to the user.

8.7 Summary

This chapter presents a solution to mitigate the problem of privacy of browsing information on centralized OSNs, for adversaries able to monitor

users interactions and further derive sensitive information. After formalizing the privacy risks we proposed a system that allows end-users to interact anonymously within the OSNs. We present the concept of *routing friends*, abusing the definition of trust in social interactions from [68].

9

Conclusions

"It always seems impossible until it's done."
– Nelson Mandela

In today's digital era, large amounts of data are shared and disseminated on a daily basis using Online Communities, primarily through Online Social Networks (OSNs). The extreme popularity of OSNs, aligned with the easy and quick dissemination channels provided, and the large data storage, contributed to several privacy and security problems directly impacting users. The role of privacy and security research is to develop solutions to deal with the various challenges, as well as to publish the experiences by demonstrating the capabilities, limitations, and tradeoffs of the solutions in privacy related problems. In this chapter, we conclude this thesis by summarizing the proposed privacy and security solutions, and sketching out open problems alongside new research questions.

9.1 Conclusions

In this thesis we devised solutions to address different privacy and security problems in the domain of Online Communities, with a focus on Online Social Networks. In order to address and suggest solutions for the current privacy problems, we first reviewed in Chapter 2 the prevalent privacy definitions, problems, and solutions that have been described in the literature. After

reviewing the current privacy research paradigms alongside the categorization of the existing privacy problems in OSNs, we emphasized mainly on surveillance problems. In particular, our research focused on privacy-enhancing technologies following notions surveyed in Chapter 4, so that content is only available to a specific target audience while placing minimum trust in providers. Hence, in Chapter 5 we devised a collaborative access control scheme that based on secret sharing, that allows OSN users to define access control rights in a collaborative fashion. Next, in Chapter 6 we modeled *end-to-end encryption* in the context of OSNs, and suggest three different constructions based on different cryptographic primitives. Each construction attains the *end-to-end encryption* property by providing confidentiality and a level of recipient anonymity, while achieving low overhead for recipients, i.e., viewers. In this way, users can enforce fine grained access control on their content, in a similar fashion as their offline social practices. In Chapter 7 we studied the notion of undetectable communications on OSNs, and propose a general cover information scheme achieving undetectability, such that the schemes proposed in Chapter 6 can be extended to achieve the undetectability property when combined with the scheme from Chapter 7. This provides users with the ability to transfer secrets through public channels, such as OSNs, without being detectable. Later, we suggested a system in Chapter 8 that provides anonymity for users browsing OSNs. Our solution leverages the traffic through their socially trusted friendship connections.

Finally, we have demonstrated different solutions addressing specific privacy problems in the category of surveillance can provide more extensive privacy protection. The interconnectivity between the different privacy problems, as shown in Chapter 2, allows the solutions proposed in this thesis to also enhance privacy with respect to the social and institutional privacy problems.

9.2 Open Research and Future Directions

Although inherently limited by the centralized nature of modern OSN architectures, as well as by the power of global government-level adversaries, the attained degree of privacy constitutes an important step forward towards secure OSN communications. Nevertheless, we foresee several open research problems and directions that call for further research, which we enumerate bellow. Some of these problems are fundamental and it is not clear ti all that they can be solved.

End-to-end encryption with support for private Targeted Advertisement. The importance of providing confidentiality of information shared through OSNs is generally blocked by the OSN business model. Henceforth, it is indeed

important to protect users' shared content, as well as to keep the functional business model of OSNs intact. In particular, there is a need for tools that deliver privacy to users while allowing targeting advertisement. This may require cryptographic techniques that reveal an abstraction of possible interests of users, without revealing the full profile of interests. Techniques such as private set intersection [74], attribute-based encryption [119], and anonymous credentials [47], provide interesting properties to reveal minimum attributes while protecting users' privacy.

Social Indistinguishability. As mentioned in Chapter 7 the undetectability of messages may be compromised by the unnatural behavior of users. In fact, the automatically generated messages must be consistent with past user behavior, whereas empowering users to compose dummy messages themselves may require external suggestions mechanisms as users are not good at coming up with diverse socially indistinguishable dummy messages. In particular, these messages should keep user behavior and follow current trends as demonstrated by Constantinides *et al.* [59]. It is an important and open challenge to obtain socially indistinguishable messages automatically from the users' profile, behavior, and current trends. Besides representing a rather complicated task, we foresee that a combination of spam detection techniques along with the current efficient data mining algorithms, e.g., process mining [200], may be able to create socially indistinguishable messages.

Usability. Usability evaluations of large-scale deployments of covert information sharing schemes represent an important challenge. As a consequence of the recent privacy breaches events, such as the Prism Project [204], Facebook and iCloud data leaks [38, 139], and the Twitter peak [157], we envision an expected boost on demand for and adoption of privacy-enhancing technologies by users and OSN providers. The usability evaluation by Balsa *et al.* [12] delivered important feedback with respect to existing problems with the implementation of cryptographic protocols and in particular for Scramble. An extension of this study, as well as a usability framework for better design and implement privacy-enhancing technologies is foreseen as an important step for adoption.

Public information reveals Private Information. The information exposed on OSNs faces privacy dangers, specially when aligned with the persistent web property, and the diligence of some of the social connections. In fact, with the OSNs social dynamics [211] social connections directly impact each user privacy. Burattin *et al.* [42] demonstrated the possibility to partly reconstruct the contact friends list from public available data, such as comments and likes. Although their result only addresses a single victim, it is possible to extend and boost this result to multiple hops to interconnect the graph. In particular, by using correlation among

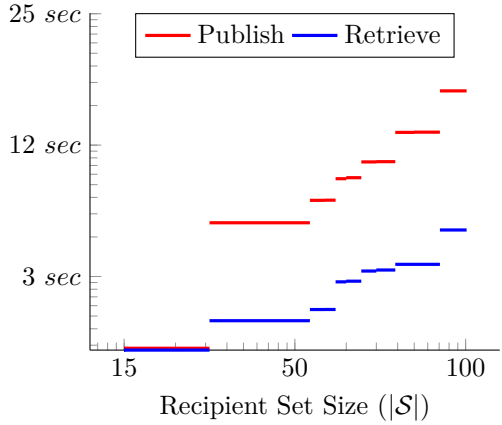


Figure 9.1: The average execution time (in log scale) of the OSN ANOPS scheme for varying sizes of the recipient set.

users that have been identified and inference methods, the result can be extended to extrapolate friends and possibly reconstruct the Facebook graph. With the aid of heuristic methods, this approach could be further refined.

Scramble! Implementation

SCRAMBLE is an open-source tool developed to protect users' privacy on Online Social Networks (OSNs). In particular, allowing the definition and enforcement of access control rules by means of encryption, independently from OSNs. The concepts implemented by Scramble are described in Chapters 5, 6, and 7. This chapter describes the architecture, challenges, and implementation details of Scramble. Finally, via a performance analysis we demonstrate the minimum overhead imposed on end-users.

A.1 Architecture Design

Scramble[1] is an open-source[2] application implemented as a Firefox Extension that allows users to enforce privacy through confidentiality on OSNs. Although it is implemented as a Firefox Extension compatible with Firefox 14+, it is written in simple Javascript, and thereby could easily be ported to other browsers, e.g., Chrome. For the description of the implementation, we distinguish between the cryptographic module and the user interface component, as depicted in Figure A.2. The former is used to realize all cryptographic operations, whereas, the latter runs the main core of the browser extension on the user environment. To overcome the fact that most OSNs *Terms of service* do not allow encryption, or may block encryption, Scramble makes use of two extra services to separate information: the storage and mapping a server, such as Dropbox and TinyURL, respectively. Figure A.1 exemplifies the Scramble process flow.

[1] Scramble!: https://www.cosic.esat.kuleuven.be/scramble/
[2] Sourceforge: http://sourceforge.net/projects/scramble-it/

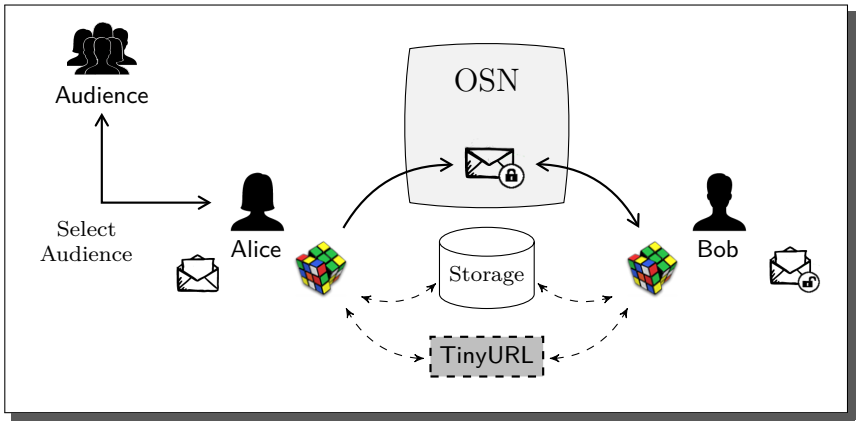


Figure A.1: Scramble! publish, and retrieve processes flow overview.

Cryptographic Module. Module responsible to implement the cryptographic mechanisms described in Chapter 6. Although OpenPGP is the default option for the multiple-encryption mechanism mainly for compatibility with email encryption, other mechanisms are available and implemented in the cryptographic module.

OpenPGP. Implements the RFC4880 [44] using the Java BouncyCastle (BC) library.

Anonymous BE. Implements the broadcast encryption protocol from Barth *et al.* [16] using the BC library as an extension of the OpenPGP, re-using OpenPGP keys, and also implemented using MIRACL [175].

oAnonymous IBE. Uses MIRACL library, and implements the outsider-Anonymous Identity-Based Broadcast Encryption protocol described in Chapter 6. For the public keys, uses Facebook IDs, such as, `http://www.facebook.com/<user_id>`.

For the symmetric encryption, Scramble uses AES-CCM [206] for symmetric (authenticated) encryption, and HMAC-SHA-256 [173] as the pseudorandom function. Simple symmetric-key operations can also be efficiently executed in Javascript using the Stanford Javascript Crypto Library (SJCL) [189], such as AES-CMAC [186] and AES-CCM.

Previous versions of Firefox allowed LiveConnect to perform the interactions between Java and Javascript. Since Firefox version 16, LiveConnect has been discontinued, making the interoperability of some components in Scramble void. Therefore, the cryptographic module interacts with Scramble Core through a local socket connection, allowing easy portability of the cryptographic module

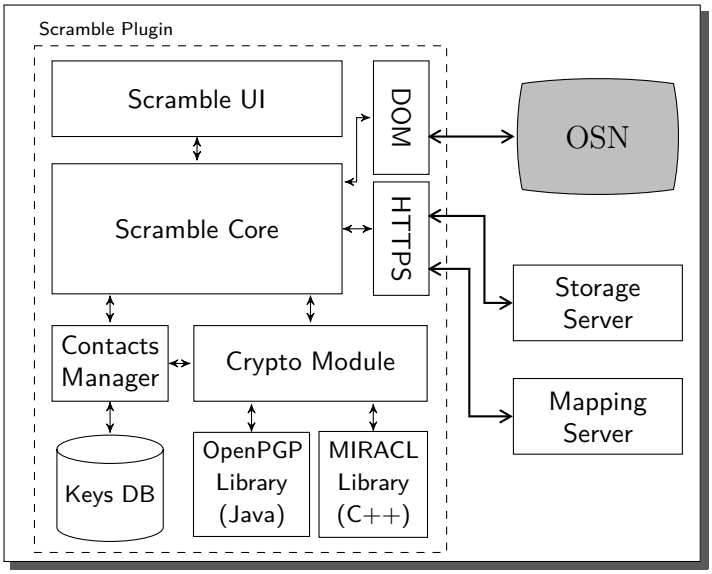


Figure A.2: Scramble! Architecture. The browser plugin communicates with the OSN, e.g., Facebook, using the Document Object Model (DOM) in the browser, such that the website is kept unaware of any changes. Whereas communication with the storage- and mapping server is done using HTTPS.

to other implementations, e.g., Chrome.

User Interface Module. Scramble is designed to work with existing OSNs, such as, Facebook, Twitter, Google+. Therefore, users interact with Scramble via the regular OSN web site, alongside an enhanced extension interface. Each operation is compliant with a normal web transaction, such that users perform them from their web browser. This component also allows users to manage their contact list, servers, and encryption keys.

Extra Entities. Scramble uses Dropbox for the storage server and TinyURL for the mapping server. Ultimately, users could select storage server from a list of different available servers, or even run their own. The TinyURL service allows users to choose a custom short URL to map to storage locations. However, similar URL shortening services, publishing services, or online blogs can also be used to store a public list of index-value pairs. In addition, to enhance privacy when accessing and generating the link users could use Tor Hidden services.

Despite the fact that Scramble defaults a generic storing server with no

authentication, users can optionally, create (automatically) a new Dropbox account with a random username, or set up their own server. All shared data is, subsequently, stored in the storing server in encrypted format. For instance, for the Dropbox option, Scramble uses the `Public` folder of the Dropbox account, accessible through a public URL.

A.2 Key Management

During installation Scramble generates a fresh OpenPGP public/private key pair (pk,sk), and an optional extra symmetric key `k` is used for enhanced protection towards possible curious mapping servers, as described in Chapter 7. The cryptographic keys are stored in the Keys database and along with the group definitions, controlled by the contacts manager.

Distribution and Verification. In general, end-users distribute and verify public keys using the twofold approach: (1) mutually trusted certification authority, or the Web-of-Trust (for OpenPGP keys), or (2) manual public key fingerprints verification through an out-of-band channel. In fact, obtaining certificates from certification authorities (CA) is a difficult, expensive and time consuming task for common users. Taking even tech-savvy users between 30 minutes to 4 hours to obtain a certificate from a public CA performing little to no verification [128]. Scramble distributes public keys using QR codes publishing them on the user's OSN profile, e.g., Facebook. Whereas, for the distribution of secret group keys `k` Scramble requires out-of-band channels.

Group Management. The contact list can be automatically retrieved from Facebook, using QR Codes for the OpenPGP implementation, while, using usernames for the IBE implementation. Users can then manage different groups locally or during the process of sharing content. The contact manager module controls the contacts and groups definitions, through XML and JSON, along with the associated keys.

Key Migration. Scramble securely migrates secret keys in two ways: exporting using out of band channel, and publishing into the storage server. The later, provides a weaker protection, but, arguably, higher usability. In order to publish to the storage server, users require a strong passphrase used in a key derivation function (KDF) to generate the key to symmetric (authenticated) encrypt the private key sk, and the list of the group keys `k`, for instance, using AES in CCM-mode [206].

A.3 Content Sharing Processes

Scramble! aims to protect user data without loss of website functionality, and keeping processes the most transparent as possible to the end-users. Therefore, all processes are transparent to viewers, while publishers must do an extra effort to publish encrypted content.

A.3.1 Sharing Protected Text

The user invokes Scramble! using the mouse cursor inside the input area, e.g., by a mouse right click menu. Scramble displays a pop-up dialog with a textbox where users can input plaintext, and choose the recipient set. Optionally, users can post directly the link to the secret message, or a text, independently from the secret message, as illustrated in Chapter 7.

Support for any Text Input Fields. Websites are becoming richer, and complex, thus, using complex Javascript calls and HTML code. As a result, text entry is no longer restricted to just a few HTML elements such as `<input type='text'>` and `<textarea>`. For example, Gmail input area for composing email messages is an editable `<html>` element within an `<iframe>`. Scramble handles special text input types by identifying the HMTL node containing the entered user input through the `document.popupNode` Firefox API call. Then, the inserted text is obtain from its `.value` attribute or `.innerHTML`, depending on the HTML node type.

Support for Rich Text Formatting. Web-based platforms increasingly encourage users to edit, annotate documents, and write HTML rich emails and blog entries. Thus, hardening the process of separation of user-generated content from page source data. For instance, Gmail couples specific HTML to email replies along with the previous emails within the same thread. Therefore, when users exchange secret messages and click the "Send" button, it is crucial to avoid reposting the initial thread secret content in plain. To achieve this, Scramble encrypts the whole message thread, including the Gmail reply headers, and the tags for rich HTML formatting are encrypted, replacing previous threads with dummy text.

A.3.2 Sharing Protected Images

Unlike text input, which can be implemented through a variety of means, pictures upload in the browser takes place exclusively through an `<input type='file'>` HTML element. At page load Scramble identifies all `file input` elements, and registers `change` event listeners. Consequently, for each file upload selection, Scramble requests users whether to protect the file, e.g., an image. Then, instead of publishing the encryption of the image into the OSN, Scramble retrieves open source images from different pages, and the DCT watermarking library[3] to embed the location of the secret. Unlike steganography, good image watermarks are resistant to typical image compression, cropping, and scaling techniques. Only intended recipients can use the secret key `k` to embed, and extract the watermark. Based on our experiments, to successfully embed a 20-digit long watermark on a random Flickr picture the DCT-watermark library takes an average of 0.3 *msec* with 54% success rate. However, a better chosen pool of pictures, and different libraries may increase success rate. Scramble automatically updates the watermarked image to the OSN, and the encrypted image to the storage server. For security reasons, unlike text, we display secret images through a pop-up window, as images stored locally cannot be embedded in a webpage hosted remotely by simply manipulating the value of the `src` attribute on an HTML `<img>` tag. In fact, manipulating the value of the `src` attribute on an HTML `<img>` tag is not possible, according to the strict origin security policy.[4] Although we describe the process for images, the same process can be applied to any other file type.

A.3.3 Extensible Page Parsing Rules

To support different platforms, Scramble uses simple XML rules that define and identify hidden data. In this way, support for one more platform boils down to the addition of new XML specification files. To specify the page structure on a generic form, Scramble uses XPath [196], a language used to navigate through elements and attributes in an XML document using path expressions to select nodes or node-sets. Therefore, Scramble uses XPath queries to identify (sender, message) pairs on a page, as exemplified in Figure A.3. The `region` query is used to restrict the search on the page to a single section containing published messages. Whereas, the execution of the `sender`, and `message` subqueries is restricted to the identified region. The identified sender is matched against contacts from the address book managed by the contact manager, containing email addresses, nicknames, and user IDs. To show the universal applicability of

[3]DTC-Watermark, by Christoph Gaffga: https://code.google.com/p/dct-watermark/
[4]Strict origin policy: http://kb.mozillazine.org/Security.fileuri.strict_origin_policy.

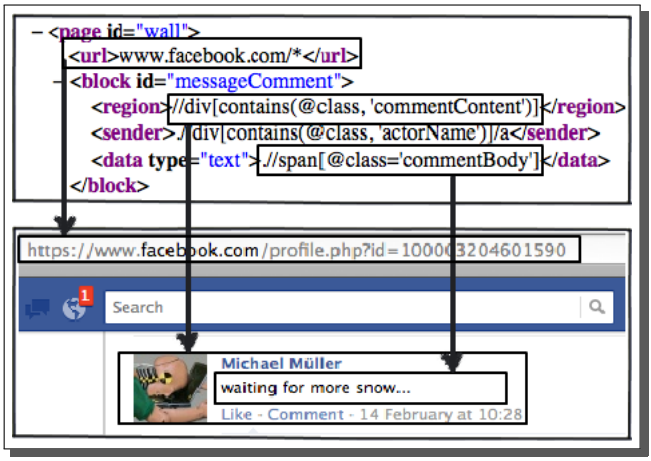


Figure A.3: Scramble! XPath Rules: identifies dummy messages candidates based on webpage-specific XPath parsing rules.

our solution, we defined parsing rules for communication over different platforms, such as, Gmail, Facebook, and Twitter. However, as the web interfaces of the supported platforms change, the XPath-based rules need to be updated.

A.4 Performance Evaluation

To evaluate the performance of Scramble, we run the Firefox Extension on a MacBook Pro laptop with an Intel Core i5 2.4GHz processor, and 4GB of memory over a wireless network.

We measured the time needed to retrieve, and display hidden messages on a Facebook page from the time the page is loaded in the browser. Note that only messages with senders in the contact list are candidates for protected communication. Processing a Facebook page with one hidden message (out of two candidates) took on average 0.9 *sec*, (N=10, *stdev*=0.2 *sec*). While, displaying a page with 10 hidden messages (out of 11 candidates) took 6 *sec* (N=10, *stdev*=0.6 *sec*). Hence, on average, retrieving hidden text messages takes 0.5 *sec* (N=25, *stdev*=70 *msec*), and processing messages holding no secret takes 0.06 *sec* (N=25, *stdev*=4 *msec*). Posting a hidden message took on average 0.67 *sec* (N=10, *stdev*=0.1 *sec*). Therefore, two users talking over a protected chat message system would experience a delay of approximately 1

second. However, the time to display a page increases linearly with the number of hidden messages, with and average memory consumption of 70MB.

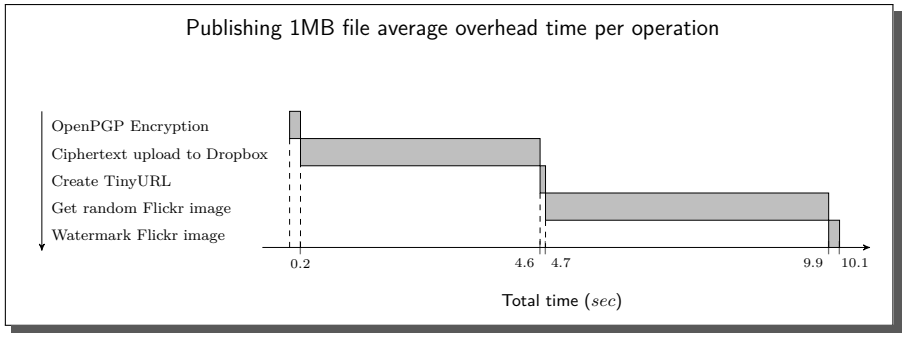


Figure A.4: Scramble!, step-by-step overhead when publishing a 1MB image, for 100 recipients. For optimization, steps could be run in parallel or be precomputed.

Figure A.4 displays the required time to execute each step when using OpenPGP along with the protocols from Chapters 6, and 7, in order to securely send a 1MB file to 100 contacts who share group shared keys. We present here only the extra security steps that must be preformed by Scramble, in comparison to the normal browser experience. The computation intensive tasks, file encryption and image watermarking, take very little time when compared to network operations when uploading the encrypted file to Dropbox and retrieving a random image from Flickr. Uploading an encrypted 1MB file to Dropbox takes on average 4.4 *sec* (*stdev*=0.6 *sec*, N=20), whereas the file encryption and size increases linearly with the number of contacts and file size taking in average 2 *sec* for a 100MB file and 500 contacts. The process to retrieve a Flickr image is automatically performed, and takes on average 5.2 *sec*, of which 3.8 *sec* were to open and search the image in the website. Once the image URL is identified, saving the image locally takes on average only 0.9 *sec* (N=50). Then, creating a TinyURL mapping the secret watermark to the encrypted Dropbox link took only 0.1 *sec* (N=20, *stdev*=0.01 *sec*). Although unrelated pictures may cause strange behavior on OSNs, users are allowed to select from the list of Flickr images one or many images to use.

Whilst executing all steps sequentially could account for slow browser response time and ultimately poor usability, implementation optimizations can make the process seem instantaneous. For instance, a pool of Flickr pictures could be retrieved, and stored locally beforehand. In addition, uploading the encrypted file could happen in parallel to other operations and finish after the upload of the watermarked image.

A.5 Summary

This chapter described Scramble, a tool that allows users to select and enforce access control rights over shared content on OSNs, independently from providers. Scramble provides different privacy benefits, mechanisms, and functionalities. However, it currently faces adoption issues, mainly due to its usability and lack of transparency entangled to the user's low knowledge on security, as demonstrated by Balsa *et al.* [12].

Bibliography

[1] Abdelberi, C., Chen, T., Cunche, M., Cristofaro, E. D., Friedman, A., and Kâafar, M. A. Censorship in the Wild: Analyzing Internet Filtering in Syria. In *IMC 2014* (Oct. 2014), C. Williamson, A. Akella, and N. Taft, Eds., ACM, pp. 285–298.

[2] Acquisti, A., Balsa, E., Berendt, B., Clarke, D., Wolf, R. D., Diaz, C., Gao, B., G´urses, S., Kuczerawy, A., Pierson, J., Piessens, F., Sayaf, R., Schellens, T., Stutzman, F., Alsenoy, B. V., and Vanderhoven, E. SPION Deliverable 2.1 State of the Art. COSIC internal report, 2011.

[3] Acquisti, A., and Gross, R. Imagined Communities: Awareness, Information Sharing, and Privacy on the Facebook. In *PETS 2006* (Jun. 2006), G. Danezis and P. Golle, Eds., vol. 2482 of *LNCS*, Springer, pp. 36–58.

[4] AlSabah, M., Bauer, K. S., Elahi, T., and Goldberg, I. The path less travelled: Overcoming Tor's bottlenecks with traffic splitting. In *PETS 2013* (Jul. 2013), E. D. Cristofaro and M. Wright, Eds., vol. 7981 of *LNCS*, Springer, pp. 143–163.

[5] Anderson, J., Díaz, C., Bonneau, J., and Stajano, F. Privacy-enabling social networking over untrusted networks. In *WOSN 2009* (Aug. 2009), J. Crowcroft and B. Krishnamurthy, Eds., ACM, pp. 1–6.

[6] Arrington, M. Is Facebook really censoring search when it suits them? In *Tech Crunch* (Nov. 2007). `http://tcrn.ch/1AeScfw`. Accessed: Feb 16, 2015.

[7] Back, A., Goldberg, I., and Shostack, A. Freedom systems 2.1 security issues and analysis. White paper, Zero Knowledge Systems, Inc., May 2001.

[8] Backes, M., Maffei, M., and Pecina, K. A security API for distributed social networks. In *NDSS 2011* (Feb. 2011), The Internet Society.

[9] Backstrom, L., Dwork, C., and Kleinberg, J. M. Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography. In *WWW 2007* (May 2007), C. L. Williamson, M. E. Zurko, P. F. Patel-Schneider, and P. J. Shenoy, Eds., ACM, pp. 181–190.

[10] Baden, R., Bender, A., Spring, N., Bhattacharjee, B., and Starin, D. Persona: an online social network with user-defined privacy. In *ACM SIGCOMM 2009* (Aug. 2009), P. Rodriguez, E. W. Biersack, K. Papagiannaki, and L. Rizzo, Eds., ACM, pp. 135–146.

[11] Balebako, R., and Cranor, L. F. Improving app privacy: Nudging app developers to protect user privacy. In *IEEE Security & Privacy* (Jan-Feb. 2014), vol. 12, IEEE Computer Society, pp. 55–58.

[12] Balsa, E., Brandimarte, L., Acquisti, A., Diaz, C., and Gürses, S. Spiny CACTOS: OSN users attitudes and perceptions towards cryptographic access control tools. In *USEC – NDSS 2014 Workshops* (Feb. 2014), M. Smith and D. Wagner, Eds., The Internet Society.

[13] Balsa, E., Troncoso, C., and Diaz, C. A metric to evaluate interaction obfuscation in online social networks. In *IJUFKS 2012* (Dec. 2012), K. Stokes and V. Torra, Eds., vol. 20, World Scientific, pp. 877–892.

[14] Bankston, K. Facebook's New Privacy Changes: The Good, The Bad, and The Ugly. In *Electronic Frontier Foundation* (Dec. 2009). `http://bit.ly/1EfUT14`, Accessed: Feb. 16, 2015.

[15] Barreto, P. S. L. M., Lynn, B., and Scott, M. Constructing elliptic curves with prescribed embedding degrees. In *SCN 02* (Sept. 2002), S. Cimato, C. Galdi, and G. Persiano, Eds., vol. 2576 of *LNCS*, Springer, pp. 257–267.

[16] Barth, A., Boneh, D., and Waters, B. Privacy in encrypted content distribution using private broadcast encryption. In *FC 2006* (Feb. / Mar. 2006), G. Di Crescenzo and A. Rubin, Eds., vol. 4107 of *LNCS*, Springer, pp. 52–64.

[17] Beato, F., Conti, M., and Preneel, B. Friend in the Middle (FiM): Tackling de-anonymization in social networks. In *IEEE SESOC – PERCOM 2013 Workshops* (Mar. 2013), T. Strufe and M. Önen, Eds., IEEE, pp. 279–284.

[18] Beato, F., Conti, M., Preneel, B., and Vettore, D. Virtualfriendship: Hiding interactions on online social networks. In *IEEE CNS 2014* (Oct. 2014), Y. Chen and R. Poovendran, Eds., IEEE, pp. 328–336.

[19] Beato, F., Cristofaro, E. D., and Rasmussen, K. B. Undetectable communication: The online social networks case. In *PST 2014* (Jul. 2014), A. Miri, U. Hengartner, N. Huang, A. Jøsang, and J. García-Alfaro, Eds., IEEE Computer Society Press, pp. 19–26.

[20] Beato, F., Ion, I., Čapkun, S., Preneel, B., and Langheinrich, M. For some eyes only: protecting online information sharing. In *ACM CODASPY 2013* (Feb. 2013), E. Bertino, R. S. Sandhu, L. Bauer, and J. Park, Eds., ACM, pp. 1–12.

[21] Beato, F., Kohlweiss, M., and Wouters, K. Enforcing Access Control in Social Networks. *HotPets – PETs 2009 Workshops* (Aug. 2009).

[22] Beato, F., Kohlweiss, M., and Wouters, K. Scramble! your social network data. In *PETS 2011* (Jul. 2011), S. Fischer-Hübner and N. Hopper, Eds., vol. 6794 of *LNCS*, Springer, pp. 211–225.

[23] Beato, F., Meul, S., and Preneel, B. Pratical Identity Based Encryption for Online Social Networks. COSIC internal report, 2014.

[24] Beato, F., and Peeters, R. Collaborative joint content sharing for online social networks. In *IEEE SESOC – PERCOM 2014 Workshops* (Mar. 2014), T. Strufe and M. Önen, Eds., IEEE, pp. 616–621.

[25] Bellare, M., Boldyreva, A., Desai, A., and Pointcheval, D. Key-privacy in public-key encryption. In *ASIACRYPT 2001* (Dec. 2001), C. Boyd, Ed., vol. 2248 of *LNCS*, Springer, pp. 566–582.

[26] Bernstein, D. J. Curve25519: New Diffie-Hellman speed records. In *PKC 2006* (Apr. 2006), M. Yung, Y. Dodis, A. Kiayias, and T. Malkin, Eds., vol. 3958 of *LNCS*, Springer, pp. 207–228.

[27] Bethencourt, J., Sahai, A., and Waters, B. Ciphertext-policy attribute-based encryption. In *IEEE Symposium on Security & Privacy 2007* (May 2007), IEEE Computer Society Press, pp. 321–334.

[28] Beye, M., Jeckmans, A. J. P., Erkin, Z., Hartel, P. H., Lagendijk, R. L., and Tang, Q. Literature overview - privacy in online social networks. Technical Report TR-CTIT-10-36, Oct. 2010.

[29] Bichsel, P., Camenisch, J., and Verdicchio, M. Recognizing your digital friends. In *Security and Privacy in Social Networks*, Y. Altshuler, Y. Elovici, A. B. Cremers, N. Aharony, and A. Pentland, Eds. Springer, 2013, ch. 3, pp. 27–46.

[30] Blaze, M., Feigenbaum, J., and Lacy, J. Decentralized trust management. In *IEEE Symposium on Security & Privacy 1996* (1996), IEEE Computer Society Press, pp. 164–173.

[31] Boneh, D., and Franklin, M. K. Identity-based encryption from the Weil pairing. In *CRYPTO 2001* (Aug. 2001), J. Kilian, Ed., vol. 2139 of *LNCS*, Springer, pp. 213–229.

[32] Bonneau, J., Anderson, J., Anderson, R. J., and Stajano, F. Eight friends are enough: social graph approximation via public listings. In *SNS 2009* (Mar. 2009), T. Stein and M. Cha, Eds., ACM, pp. 13–18.

[33] Bonneau, J., and Preibusch, S. The privacy jungle: On the market for data protection in social networks. In *EISP 2010* (Jun. 2010), T. Moore, D. Pym, and C. Ioannidis, Eds., Springer, pp. 121–167.

[34] Borges, F., Martucci, L. A., Beato, F., and Mühlhäuser, M. Secure and privacy-friendly public key generation and certification. In *IEEE TrustCom 2014* (Sept. 2014), Y. Liu, Ed., IEEE, pp. 114–121.

[35] Borisov, N., Goldberg, I., and Brewer, E. A. Off-the-record communication, or, why not to use PGP. In *WPES 2004* (Oct. 2004), V. Atluri, P. F. Syverson, and S. D. C. di Vimercati, Eds., ACM, pp. 77–84.

[36] Bos, J. N., and Chaum, D. Provably unforgeable signatures. In *CRYPTO'92* (Aug. 1992), E. F. Brickell, Ed., vol. 740 of *LNCS*, Springer, pp. 1–14.

[37] Bosker, B. Twitter To Censor Tweets In Some Countries. In *The Huffington Post* (Mar. 2012). `http://huff.to/1Mm5kWE`, Accessed: Feb. 16, 2015.

[38] Bosker, B. How Facebook Explains User Data Bug That Leaked 6 Million People's Information. In *The Huffington Post* (Feb. 2015). `http://huff.to/1A4Y5NG`, Accessed: Feb. 18, 2015.

[39] Boyd, D. Why youth (heart) social network sites: The role of networked publics in teenage social life. In *Digital Media and Learning* (2007), D. Buckingham, Ed., MIT Press, pp. 119–142.

[40] Boyd, D. M., and Ellison, N. B. Social Network Sites: Definition, History, and Scholarship. *JCMC 2007 13*, 1 (Oct. 2007), 210–230.

[41] Brands, S. Rapid demonstration of linear relations connected by boolean operators. In *EUROCRYPT'97* (May 1997), W. Fumy, Ed., vol. 1233 of *LNCS*, Springer, pp. 318–333.

[42] Burattin, A., Cascavilla, G., and Conti, M. SocialSpy: Browsing (Supposedly) Hidden Information in Online Social Networks.

[43] Cachin, C. An information-theoretic model for steganography. In *IH 1998* (Apr. 1998), D. Aucsmith, Ed., vol. 1525 of *LNCS*, Springer, pp. 306–318.

[44] Callas, J., Donnerhacke, L., Finney, H., Shaw, D., and Thayer, R. OpenPGP Message Format. In *RFC* (Nov. 2007), no. 4880, Internet Engineering Task Force, IETF.

[45] Camenisch, J., Hohenberger, S., Kohlweiss, M., Lysyanskaya, A., and Meyerovich, M. How to win the clonewars: Efficient periodic n-times anonymous authentication. In *ACM CCS 06* (Oct. / Nov. 2006), A. Juels, R. N. Wright, and S. Vimercati, Eds., ACM Press, pp. 201–210.

[46] Camenisch, J., Hohenberger, S., and Lysyanskaya, A. Compact e-cash. In *EUROCRYPT 2005* (May 2005), R. Cramer, Ed., vol. 3494 of *LNCS*, Springer, pp. 302–321.

[47] Camenisch, J., and Lysyanskaya, A. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *EUROCRYPT 2001* (May 2001), B. Pfitzmann, Ed., vol. 2045 of *LNCS*, Springer, pp. 93–118.

[48] Cancelli, G., and Barni, M. Mpsteg-color: A new steganographic technique for color images. In *IH 2007* (Jun. 2007), T. Furon, F. Cayre, G. J. Doërr, and P. Bas, Eds., vol. 4567 of *LNCS*, Springer, pp. 1–15.

[49] Carminati, B., and Ferrari, E. Access control and privacy in web-based social networks. *IJWIS 4*, 4 (2008), 395–415.

[50] Castiglione, A., D'Alessio, B., and Santis, A. D. Steganography and secure communication on online social networks and online photo sharing. In *BWCCA 2011* (Oct. 2011), L. Barolli, F. Xhafa, K. F. Li, and A. Gentile, Eds., IEEE, pp. 363–368.

[51] Chandramouli, R., and Memon, N. D. Analysis of LSB based image steganography techniques. In *ICIP (2001)* (2001), vol. 3, IEEE, pp. 1019–1022.

[52] Chaum, D. Untraceable electronic mail, return addresses, and digital pseudonyms. In *Communications of ACM* (Feb. 1981), vol. 24, ACM, pp. 84–88.

[53] Chaum, D. Blind signature system. In *CRYPTO'83* (1983), D. Chaum, Ed., Plenum Press, New York, USA, p. 153.

[54] Chew, M., Balfanz, D., and Laurie, B. (Under)mining Privacy in Social Networks. In *W2SP – S&P 2008 Workshops* (May 2008), IEEE.

[55] Chor, B., Goldwasser, S., Micali, S., and Awerbuch, B. Verifiable secret sharing and achieving simultaneity in the presence of faults (extended abstract). In *26th FOCS* (Oct. 1985), IEEE Computer Society Press, pp. 383–395.

[56] Christodorescu, M. Private use of untrusted web servers via opportunistic encryption. In *W2SP – S&P 2008 Workshops* (May 2008), IEEE.

[57] Christofides, E., Muise, A., and Desmarais, S. Information disclosure and control on Facebook: are they two sides of the same coin or two different processes? *CyberPsychology & Behavior 12*, 3 (2009), 341–345.

[58] Cohen, J. E. What privacy is for. *Harvard Law Review 126* (2012), 1904.

[59] Constantinides, E., del Carmen Alarcón del Amo, M., and Lorenzo Romero, C. Profiles of social networking sites users in the Netherlands. In *HTSF 2010* (May 2010), University of Twente, NIKOS.

[60] Conti, G., and Sobiesk, E. An honest man has nothing to fear: User perceptions on web-based information disclosure. In *SOUPS 2007* (Jul. 2007), L. F. Cranor, Ed., ACM, pp. 112–121.

[61] Conti, M., Hasani, A., and Crispo, B. Virtual private social networks. In *ACM CODASPY 2011* (Feb. 2011), R. S. Sandhu and E. Bertino, Eds., ACM, pp. 39–50.

[62] Cramer, R., and Shoup, V. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *CRYPTO'98* (Aug. 1998), H. Krawczyk, Ed., vol. 1462 of *LNCS*, Springer, pp. 13–25.

[63] Cutillo, L. A., Molva, R., and Önen, M. Safebook: A distributed privacy preserving online social network. In *WOWMOM 2011* (Jun. 2011), M. Chatterjee and A. Passarella, Eds., IEEE Computer Society, pp. 1–3.

[64] Cutler, K.-M. Stats: Facebook made $ 9.51 in ad revenue per user last year in the U.S. and Canada. In *Tech Crunch* (May. 2012). `http://tcrn.ch/1QyTh9I`. Accessed: Dec 3, 2014.

[65] Dainotti, A., Squarcella, C., Aben, E., Claffy, K. C., Chiesa, M., Russo, M., and Pescapè, A. Analysis of country-wide internet outages caused by censorship. In *IMC 2011* (Nov. 2011), P. Thiran and W. Willinger, Eds., ACM, pp. 1–18.

[66] Danezis, G. Mix-networks with restricted routes. In *PETS 2002* (Apr. 2002), R. Dingledine and P. F. Syverson, Eds., vol. 2482 of *LNCS*, Springer, pp. 1–17.

[67] Danezis, G., and Clayton, R. Introducing traffic analysis. In *Digital Privacy: Theory, Technologies, and Practices* (Dec. 2007), A. Acquisti, S. Gritzalis, C. Lambrinoudakis, and S. di Vimercati, Eds., Auerbach Publications, pp. 95–117.

[68] Danezis, G., Diaz, C., Troncoso, C., and Laurie, B. Drac: An architecture for anonymous low-volume communications. In *PETS 2010* (Jul. 2010), M. J. Atallah and N. J. Hopper, Eds., vol. 6205 of *LNCS*, Springer, pp. 202–219.

[69] Danezis, G., Dingledine, R., and Mathewson, N. Mixminion: Design of a type III anonymous remailer protocol. In *IEEE Symposium on Security & Privacy 2003* (May 2003), IEEE Computer Society Press, pp. 2–15.

[70] Danezis, G., and Gürses, S. A critical review of 10 years of privacy technology. In *Surveillance Cultures: A Global Surveillance Society* (Apr. 2010).

[71] De Cristofaro, E., Kim, J., and Tsudik, G. Linear-complexity private set intersection protocols secure in malicious model. In *ASIACRYPT 2010* (Dec. 2010), M. Abe, Ed., vol. 6477 of *LNCS*, Springer, pp. 213–231.

[72] De Cristofaro, E., Manulis, M., and Poettering, B. Private discovery of common social contacts. In *ACNS 11* (June 2011), J. Lopez and G. Tsudik, Eds., vol. 6715 of *LNCS*, Springer, pp. 147–165.

[73] De Cristofaro, E., Soriente, C., Tsudik, G., and Williams, A. Hummingbird: Privacy at the time of twitter. In *IEEE Symposium on Security & Privacy 20012* (May 2012), IEEE Computer Society Press, pp. 285–299.

[74] De Cristofaro, E., and Tsudik, G. Practical private set intersection protocols with linear complexity. In *FC 2010* (Jan. 2010), R. Sion, Ed., vol. 6052 of *LNCS*, Springer, pp. 143–159.

[75] Diaz, C. *Anonymity and Privacy in Electronic Services*. PhD thesis, Leuven, Belgium, Dec. 2005. Bart Preneel and Joos Vandewalle (promotors).

[76] Diaz, C., Seys, S., Claessens, J., and Preneel, B. Towards measuring anonymity. In *PETS 2002* (Apr. 2002), R. Dingledine and P. F. Syverson, Eds., vol. 2482 of *LNCS*, Springer, pp. 54–68.

[77] Diaz, C., Troncoso, C., and Serjantov, A. On the impact of social network profiling on anonymity. In *PETS 2008* (Jul. 2008), N. Borisov and I. Goldberg, Eds., vol. 5134 of *LNCS*, Springer, pp. 44–62.

[78] Dierks, T., and Rescorla, E. The Transport Layer Security (TLS) Protocol Version 1.2. In *RFC* (Aug. 2008), no. 5246, Internet Engineering Task Force, IETF.

[79] Dingledine, R., Mathewson, N., and Syverson, P. F. Tor: The second-generation onion router. In *USENIX Security 2004* (Aug. 2004), M. Blaze, Ed., USENIX Security, pp. 303–320.

[80] Dwork, C., and Naor, M. An efficient existentially unforgeable signature scheme and its applications. *Journal of Cryptology 11*, 3 (1998), 187–208.

[81] Dwyer, C., Hiltz, S., Passerini, K., Dwyer, C., Passerini, K., and Hiltz, S. R. Trust and privacy: A comparison of Facebook and myspace. In *AMCIS 2007* (Aug. 2007), AISeL, p. 339.

[82] Dwyer, J. Four nerds and a cry to arms against Facebook. In *NY Times* (May 2013). `http://nyti.ms/1hc60kv`. Accessed: Dec 3, 2014.

[83] ElGamal, T. On computing logarithms over finite fields. In *CRYPTO'85* (Aug. 1985), H. C. Williams, Ed., vol. 218 of *LNCS*, Springer, pp. 396–402.

[84] Erickson, T., and Kellogg, W. A. Social translucence: An approach to designing systems that support social processes. In *TOCHI 2000* (May 2000), vol. 7, ACM, pp. 59–83.

[85] Fazio, N., and Perera, I. M. Outsider-anonymous broadcast encryption with sublinear ciphertexts. In *PKC 2012* (May 2012), M. Fischlin, J. Buchmann, and M. Manulis, Eds., vol. 7293 of *LNCS*, Springer, pp. 225–242.

[86] Feldman, A. J., Blankstein, A., Freedman, M. J., and Felten, E. W. Social networking with frientegrity: Privacy and integrity with an untrusted provider. In *USENIX Security 2012* (Aug. 2012), T. Kohno, Ed., USENIX Security, pp. 647–662.

[87] Feldman, P. A practical scheme for non-interactive verifiable secret sharing. In *28th FOCS* (Oct. 1987), IEEE Computer Society Press, pp. 427–437.

[88] Fiat, A., and Naor, M. Broadcast encryption. In *CRYPTO'93* (Aug. 1993), D. R. Stinson, Ed., vol. 773 of *LNCS*, Springer, pp. 480–491.

[89] Fischetti, M. Graphic science: Data theft: Hackers attack. *Scientific American 305*, 4 (Oct. 2011), 100–100.

[90] Freedman, M. J., and Morris, R. Tarzan: a peer-to-peer anonymizing network layer. In *ACM CCS 02* (Nov. 2002), V. Atluri, Ed., ACM Press, pp. 193–206.

[91] FreedomHouse.org. A Global Assessment of Internet and Digital Media. In *Freedom on the net 2012* (Sept. 2012), S. Kelly, S. Cook, and M. Truong, Eds. `http://bit.ly/1zRNUvE`.

[92] Freedomhouse.org. A Global Assessment of Internet and Digital Media. In *Freedom on the net 2013* (Oct. 2013), S. Kelly, M. Truong, M. Earp, L. R. nd Adrian Shahbaz, and A. Greco-Stoner, Eds. `http://bit.ly/1vQBVg3`.

[93] Freedomhouse.org. Tightening the net: Governments expand online controls. In *Freedom on the net 2014* (Oct. 2014), S. Kelly, M. Earp, L. Reed, A. Shahbaz, and M. Truong, Eds. `http://bit.ly/1vQBVg3`.

[94] Fujisaki, E., Okamoto, T., Pointcheval, D., and Stern, J. RSA-OAEP is secure under the RSA assumption. In *CRYPTO 2001* (Aug. 2001), J. Kilian, Ed., vol. 2139 of *LNCS*, Springer, pp. 260–274.

[95] Galbraith, S., Paterson, K., and Smart, N. Pairings for cryptographers. Cryptology ePrint Archive, Report 2006/165, 2006. `http://eprint.iacr.org/2006/165`.

[96] Gentry, C. Practical identity-based encryption without random oracles. In *EUROCRYPT 2006* (May / June 2006), S. Vaudenay, Ed., vol. 4004 of *LNCS*, Springer, pp. 445–464.

[97] Gentry, C., and Waters, B. Adaptive security in broadcast encryption systems. Cryptology ePrint Archive, Report 2008/268, 2008. `http://eprint.iacr.org/2008/268`.

[98] Gerstein, R. S. Intimacy and privacy. *Ethics* (Jan. 1978), 76–81.

[99] Gill, P., Erramilli, V., Chaintreau, A., Krishnamurthy, B., Papagiannaki, K., and Rodriguez, P. Follow the money: understanding economics of online aggregation and advertising. In *IMC 2013* (Oct. 2013), K. Papagiannaki, P. K. Gummadi, and C. Partridge, Eds., ACM, pp. 141–148.

[100] Goffman, E. In *The Presentation of Self in Everyday Life* (Jun. 1959), vol. 1, Anchor.

[101] Golbeck, J., and Hendler, J. A. Accuracy of metrics for inferring trust and reputation in semantic web-based social networks. In *EKAW 2004* (Oct. 2004), E. Motta, N. Shadbolt, A. Stutt, and N. Gibbins, Eds., vol. 3257 of *LNCS*, Springer, pp. 116–131.

[102] Goldberg, I. Privacy-enhancing technologies for the internet, ii: Five years later. In *PETS 2002*, R. Dingledine and P. F. Syverson, Eds., vol. 2482 of *LNCS*. Springer, Apr. 2002, pp. 1–12.

[103] Goldberg, I., Wagner, D., and Brewer, E. Privacy-enhancing technologies for the internet. In *IEEE Spring COMPCON* (Feb. 1997), IEEE Computer Society Press.

[104] Goldreich, O., Goldwasser, S., and Micali, S. How to construct random functions. *Journal of the ACM 33*, 4 (Oct. 1986), 792–807.

[105] Goldwasser, S., Micali, S., and Rackoff, C. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing 18*, 1 (1989), 186–208.

[106] Greschbach, B., Kreitz, G., and Buchegger, S. The devil is in the metadata - New privacy challenges in Decentralised Online Social Networks. In *IEEE SESOC – PERCOM 2012 Workshops* (Mar. 2012), T. Strufe and M. Önen, Eds., IEEE, pp. 333–339.

[107] Gross, R., and Acquisti, A. Information revelation and privacy in online social networks. In *WPES 2005* (Nov. 2005), V. Atluri, S. D. C. di Vimercati, and R. Dingledine, Eds., ACM, pp. 71–80.

[108] Guha, S., Tang, K., and Francis, P. Noyb: privacy in online social networks. In *WOSN 2008* (Aug. 2008), C. Faloutsos, T. Karagiannis, and P. Rodriguez, Eds., ACM, pp. 49–54.

[109] Günther, F., Manulis, M., and Strufe, T. Cryptographic treatment of private user profiles. In *RLCPS – FC 2011 Workshops* (Feb. 2011),

G. Danezis, S. Dietrich, and K. Sako, Eds., vol. 7126 of *LNCS*, Springer, pp. 40–54.

[110] Guo, H., Zhang, Z., and Zhang, J. Proxy re-encryption with unforgeable re-encryption keys. In *CANS 14* (Oct. 2014), D. Gritzalis, A. Kiayias, and I. G. Askoxylakis, Eds., vol. 8813 of *LNCS*, Springer, pp. 20–33.

[111] Gürses, S. *Multilateral Privacy Requirements Analysis in Online Social Network Services*. PhD thesis, Leuven, Belgium, May 2010. Bettina Berendt and Bart Preneel (promotors).

[112] Gürses, S., and Berendt, B. The social web and privacy: Practices, reciprocity and conflict detection in social networks. In *PAKD 2010* (Dec. 2010), E. F. Francesco Bonchi, Ed., Chapman & Hall/CRC Press, pp. 395–433.

[113] Gürses, S., and Diaz, C. Two tales of privacy in online social networks. In *IEEE Security & Privacy* (May-Jun. 2013), vol. 11, IEEE Computer Society, pp. 29–37.

[114] Gürses, S., Troncoso, C., and Diaz, C. Engineering privacy by design. *CPDP* (Jan. 2011).

[115] Gürses, S. F., Rizk, R., and Günther, O. Privacy design in online social networks: Learning from privacy breaches and community feedback. In *ICIS 2008* (Dec. 2008), Association for Information Systems, p. 90.

[116] Haggerty, K. D., and Ericson, R. V. The surveillant assemblage. In *The British Journal of Sociology* (Dec. 2000), vol. 51, pp. 605–622.

[117] Hess, F., Smart, N., and Vercauteren, F. The eta pairing revisited. Cryptology ePrint Archive, Report 2006/110, 2006. `http://eprint.iacr.org/2006/110`.

[118] Hirose, S., and Yoshida, S. An authenticated Diffie-Hellman key agreement protocol secure against active attacks. In *PKC'98* (Feb. 1998), H. Imai and Y. Zheng, Eds., vol. 1431 of *LNCS*, Springer, pp. 135–148.

[119] Hohenberger, S., and Waters, B. Attribute-based encryption with fast decryption. In *PKC 2013* (Feb. / Mar. 2013), K. Kurosawa and G. Hanaoka, Eds., vol. 7778 of *LNCS*, Springer, pp. 162–179.

[120] Hopper, N. J., Langford, J., and von Ahn, L. Provably secure steganography. In *CRYPTO 2002* (Aug. 2002), M. Yung, Ed., vol. 2442 of *LNCS*, Springer, pp. 77–92.

[121] Horner, W. G. A new method of solving numerical equations of all orders, by continuous approximation. In *Philosophical Transactions* (Jul. 1819), Royal Society of London, pp. 308–335.

[122] Invernizzi, L., Kruegel, C., and Vigna, G. Message in a bottle: sailing past censorship. C. Payne, Ed., ACM, pp. 39–48.

[123] Jahid, S., Mittal, P., and Borisov, N. EASiER: encryption-based access control in social networks with efficient revocation (short paper). In *ASIACCS 11* (Mar. 2011), B. S. N. Cheung, L. C. K. Hui, R. S. Sandhu, and D. S. Wong, Eds., ACM Press, pp. 411–415.

[124] Jernigan, C., and Mistree, B. F. T. Gaydar: Facebook friendships expose sexual orientation. In *First Monday* (Oct. 2009), vol. 14.

[125] Johnson, A., Wacek, C., Jansen, R., Sherr, M., and Syverson, P. F. Users get routed: traffic correlation on tor by realistic adversaries. In *ACM CCS 13* (Nov. 2013), A.-R. Sadeghi, V. D. Gligor, and M. Yung, Eds., ACM Press, pp. 337–348.

[126] Joux, A. A new index calculus algorithm with complexity $l(1/4 + o(1))$ in very small characteristic. Cryptology ePrint Archive, Report 2013/095, 2013. `http://eprint.iacr.org/2013/095`.

[127] Jr, J. C. B. Meet Facebook's Mr. Nice: At Facebook, Creating Empathy Among Cyberbullying. In *NY Times* (Oct. 2014). `http://nyti.ms/1A2PkDI`. Accessed: Feb 3, 2015.

[128] Kapadia, A. A Case (Study) For Usability in Secure Email Communication. In *IEEE Security & Privacy* (Mar. 2007), vol. 5, IEEE Computer Society, pp. 80–84.

[129] Kim, Y., Perrig, A., and Tsudik, G. Simple and fault-tolerant key agreement for dynamic collaborative groups. In *ACM CCS 00* (Nov. 2000), S. Jajodia and P. Samarati, Eds., ACM Press, pp. 235–244.

[130] Kincaid, J. This is the second time a Google engineer has been fired for accessing user data. In *Tech Crunch* (Sep. 2010). `http://tcrn.ch/LUH6Hz`. Accessed: Dec 3, 2014.

[131] Kincaid, J. Mobile as it starts sharing user counts by country. In *Tech Crunch* (Sep. 2013). `http://tcrn.ch/JHZ0fE`. Accessed: Jan 10, 2015.

[132] King, J., Lampinen, A., and Smolen, A. Privacy: is there an app for that? In *SOUPS 2011* (Jul. 2011), L. F. Cranor, Ed., ACM, pp. 1–20.

[133] Krawczyk, H. SIGMA: The "SIGn-and-MAc" approach to authenticated Diffie-Hellman and its use in the IKE protocols. In *CRYPTO 2003* (Aug. 2003), D. Boneh, Ed., vol. 2729 of *LNCS*, Springer, pp. 400–425.

[134] Krishnamurthy, B. I know what you will do next summer. *ACM Computer Communication Review 40*, 5 (2010), 65–70.

[135] Krishnamurthy, B., Naryshkin, K., and Wills, C. E. Privacy leakage vs. Protection measures: the growing disconnect. In *W2SP – S&P 2011 Workshops* (May 2011), IEEE.

[136] Krishnamurthy, B., and Wills, C. E. On the leakage of personally identifiable information via online social networks. *ACM Computer Communication Review 40*, 1 (2010), 112–117.

[137] Laurie, B., Langley, A., and Kasper, E. Certificate Transparency. In *RFC* (Jun. 2013), no. 6962, Internet Engineering Task Force, IETF.

[138] Leech, M., Ganis, M., Lee, Y., Kuris, R., Koblas, D., and Jones, L. SOCKS Protocol Version 5. In *RFC* (Mar. 1996), no. 1929, Internet Engineering Task Force, IETF.

[139] Lewis, D. iCloud Data Breach: Hacking And Celebrity Photos. In *Forbes Online* (Sept. 2014). `http://onforb.es/1Cmngvl`, Accessed: Jan. 6, 2015.

[140] Libert, B., Paterson, K. G., and Quaglia, E. A. Anonymous broadcast encryption: Adaptive security and efficient constructions in the standard model. In *PKC 2012* (May 2012), M. Fischlin, J. Buchmann, and M. Manulis, Eds., vol. 7293 of *LNCS*, Springer, pp. 206–224.

[141] Liu, H., Vasserman, E. Y., and Hopper, N. Improved group off-the-record messaging. In *WPES 2013* (Nov. 2013), A.-R. Sadeghi and S. Foresti, Eds., ACM, pp. 249–254.

[142] Lucas, M. M., and Borisov, N. Flybynight: mitigating the privacy risks of social networking. pp. 1–8.

[143] Luo, W., Xie, Q., and Hengartner, U. Facecloak: An architecture for user privacy on social networking sites. In *CSE 2009* (Aug. 2009), IEEE Computer Society, pp. 26–33.

[144] Malandrino, D., Petta, A., Scarano, V., Serra, L., Spinelli, R., and Krishnamurthy, B. Privacy awareness about information leakage: who knows what about me? In *WPES 2013* (Nov. 2013), A.-R. Sadeghi and S. Foresti, Eds., ACM, pp. 279–284.

[145] Manulis, M., Poettering, B., and Tsudik, G. Affiliation-hiding key exchange with untrusted group authorities. In *ACNS 10* (June 2010), J. Zhou and M. Yung, Eds., vol. 6123 of *LNCS*, Springer, pp. 402–419.

[146] Mao, H., Shuai, X., and Kapadia, A. Loose tweets: an analysis of privacy leaks on twitter. In *WPES 2011* (Oct. 2011), Y. Chen and J. Vaidya, Eds., ACM, pp. 1–12.

[147] Matyszczyk, C. If your account is subpoenaed, Facebook sends police, well, everything. In *CNET* (Apr. 2012). `http://cnet.co/1zILKNn`. Accessed: Dec 3, 2014.

[148] Menezes, A. J., van Oorschot, P. C., and Vanstone, S. A. *Handbook of Applied Cryptography*. The CRC Press series on discrete mathematics and its applications. CRC Press, 2000 N.W. Corporate Blvd., Boca Raton, FL 33431-9868, USA, 1997.

[149] Merkle, R. C. *Secrecy, Authentication, and Public Key Systems*. PhD thesis, Stanford, CA, USA, Jun. 1979.

[150] Miners, Z. End-to-end encryption needs to be easier for users before Facebook embraces it. In *PCWorld* (Mar. 2014). `http://bit.ly/1iCBi2i`, Accessed: Oct. 27, 2014.

[151] Moghaddam, H. M., Li, B., Derakhshani, M., and Goldberg, I. SkypeMorph: protocol obfuscation for Tor bridges. In *ACM CCS 12* (Oct. 2012), T. Yu, G. Danezis, and V. D. Gligor, Eds., ACM Press, pp. 97–108.

[152] Möller, U., Cottrell, L., Palfrader, P., and Sassaman, L. Mixmaster Protocol — Version 2, Jul. 2003.

[153] Murdoch, S. J., and Danezis, G. Low-cost traffic analysis of tor. In *IEEE Symposium on Security & Privacy 2005* (May 2005), IEEE Computer Society Press, pp. 183–195.

[154] Nagy, M., Cristofaro, E. D., Dmitrienko, A., Asokan, N., and Sadeghi, A. Do I know you?: efficient and privacy-preserving common friend-finder protocols and applications. In *ACSAC 2013* (Dec. 2013), C. N. P. Jr., Ed., ACM, pp. 159–168.

[155] Naor, M., and Pinkas, B. Efficient trace and revoke schemes. In *FC 2000* (Feb. 2000), Y. Frankel, Ed., vol. 1962 of *LNCS*, Springer, pp. 1–20.

[156] Narayanan, A., and Shmatikov, V. De-anonymizing social networks. In *IEEE Symposium on Security & Privacy 2009* (May 2009), IEEE Computer Society Press, pp. 173–187.

[157] News, S. Twitter admits peeking at address books, announces privacy improvements. In *Fox News* (Feb. 2012). http://fxn.ws/1BiRp01, Accessed: Feb. 16, 2015.

[158] Nguyen, D. H., and Mynatt, E. D. Privacy mirrors: understanding and shaping socio-technical ubiquitous computing systems. Tech. rep., 2002.

[159] Nilizadeh, S., Jahid, S., Mittal, P., Borisov, N., and Kapadia, A. Cachet: a decentralized architecture for privacy preserving social networking with caching. In *CoNEXT 2012* (Dec. 2012), C. Barakat, R. Teixeira, K. K. Ramakrishnan, and P. Thiran, Eds., ACM, pp. 337–348.

[160] Nojoumian, M., Stinson, D. R., and Grainger, M. Unconditionally secure social secret sharing scheme. Cryptology ePrint Archive, Report 2009/207, 2009. http://eprint.iacr.org/2009/207.

[161] Oremus, W. Facebook sued for "reading" your private messages. In *Slate* (Jan. 2014). http://slate.me/1evQXN1, Accessed: Oct. 27, 2014.

[162] Paterson, K. G., and Srinivasan, S. Security and anonymity of identity-based encryption with multiple trusted authorities. In *PAIRING 2008* (Sept. 2008), S. D. Galbraith and K. G. Paterson, Eds., vol. 5209 of *LNCS*, Springer, pp. 354–375.

[163] Pedersen, T. P. Non-interactive and information-theoretic secure verifiable secret sharing. In *CRYPTO'91* (Aug. 1991), J. Feigenbaum, Ed., vol. 576 of *LNCS*, Springer, pp. 129–140.

[164] Pedersen, T. P. A threshold cryptosystem without a trusted party (extended abstract) (rump session). In *EUROCRYPT'91* (Apr. 1991), D. W. Davies, Ed., vol. 547 of *LNCS*, Springer, pp. 522–526.

[165] Perez, J. C. Facebook's beacon more intrusive than previously thought. In *PCWorld* (Nov. 2007). http://bit.ly/1AtYYBo, Accessed: FEB. 17, 2015.

[166] Pfitzmann, A., and Köhntopp, M. Anonymity, unobservability, and pseudonymity - A proposal for terminology. In *DIAU 2000* (Jul. 2000), H. Federrath, Ed., vol. 2009 of *LNCS*, Springer, pp. 1–9.

[167] Prosser, W. L. Privacy. In *California Law Review* (Aug. 1960), vol. 48, p. 383–423.

[168] Protalinski, E. 600 million of Facebook's 1 billion users are mobile. In *The Next Web* (Oct. 2012). http://tnw.co/1zNq9Ve. Accessed: Dec 3, 2014.

[169] Raimondo, M. D., Gennaro, R., and Krawczyk, H. Secure off-the-record messaging. In *WPES 2005* (Nov. 2005), V. Atluri, S. D. C. di Vimercati, and R. Dingledine, Eds., ACM, pp. 81–89.

[170] Randall, D., and Richards, V. Facebook can ruin your life. And so can MySpace, Bebo... In *The Independent* (Mar. 2008). `http://bit.ly/1AHXW36`, Accessed: Feb. 16, 2015.

[171] Riederer, C., Erramilli, V., Chaintreau, A., Krishnamurthy, B., and Rodriguez, P. For sale : your data: by : you. In *HOTNETS 2011* (Nov. 2011), H. Balakrishnan, D. Katabi, A. Akella, and I. Stoica, Eds., ACM, p. 13.

[172] Rosen, E., and Rekhter, Y. BGP/MPLS IP Virtual Private Networks (VPNs). In *RFC* (Feb. 2006), no. 4364, Internet Engineering Task Force, IETF.

[173] S. Kelly, S. F. Using HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 with IPsec . In *RFC* (May. 2007), no. 4868, Internet Engineering Task Force, IETF.

[174] Salowey, J., Choudhury, A., and McGrew, D. AES Galois Counter Mode (GCM) Cipher Suites for TLS. In *RFC* (Aug. 2008), no. 5288, Internet Engineering Task Force, IETF.

[175] Scott, M. Miracl–multiprecision integer and rational arithmetic c/c++ library. *Shamus Software Ltd, Dublin, Ireland, URL* (2003).

[176] Scott, M. On the efficient implementation of pairing-based protocols. In *13th IMA International Conference on Cryptography and Coding* (Dec. 2011), L. Chen, Ed., vol. 7089 of *LNCS*, Springer, pp. 296–308.

[177] Semitsu, J. P. From Facebook to Mug Shot: How the Dearth of Social Networking Privacy Rights Revolutionized Online Government Surveillance. *Pace Law Review 31*, 1 (Jan. 2011), 95.

[178] Serjantov, A., and Danezis, G. Towards an information theoretic metric for anonymity. In *PETS 2002* (Apr. 2002), R. Dingledine and P. F. Syverson, Eds., vol. 2482 of *LNCS*, Springer, pp. 41–53.

[179] Shakimov, A., Lim, H., Cáceres, R., Cox, L. P., Li, K. A., Liu, D., and Varshavsky, A. Vis-à-vis: Privacy-preserving online social networking via virtual individual servers. In *COMSNETS 2011* (Jan. 2011), D. B. Johnson and A. Kumar, Eds., IEEE, pp. 1–10.

[180] Shamir, A. How to share a secret. *Communications of the Association for Computing Machinery 22*, 11 (Nov. 1979), 612–613.

[181] Shamir, A. On the security of DES. In *CRYPTO'85* (Aug. 1985), H. C. Williams, Ed., vol. 218 of *LNCS*, Springer, pp. 280–281.

[182] Simoens, K., Yang, B., Zhou, X., Beato, F., Busch, C., Newton, E., and Preneel, B. Criteria Towards Metrics for Benchmarking Template Protection Algorithms. In *IAPR ICB 2012* (Mar.-Apr. 2012), A. K. Jain, A. Ross, S. Prabhakar, and J. Kim, Eds., IEEE, pp. 498–505.

[183] Slepak, G. DNSChain + okTurtles. In *White Paper* (Apr. 2014), okTurtles. `https://okturtles.com/other/dnschain_okturtles_overview.pdf`, Accessed: Jan. 10, 2015.

[184] Solove, D. J. A taxonomy of privacy. *University of Pennsylvania Law Review 154*, 3 (Jan. 2006), 477–564.

[185] Solove, D. J. "I've Got Nothing to Hide" and Other Misunderstandings of Privacy. *San Diego Law Review 44* (2007), 745.

[186] Song, J., Poovendran, R., and Lee, J. The AES-CMAC-96 Algorithm and Its Use with IPsec. In *RFC* (Jun. 2006), no. 4494, Internet Engineering Task Force, IETF.

[187] Squicciarini, A. C., Paci, F., and Sundareswaran, S. PriMa: an effective privacy protection mechanism for social networks (short paper). In *ASIACCS 10* (Apr. 2010), D. Feng, D. A. Basin, and P. Liu, Eds., ACM Press, pp. 320–323.

[188] Squicciarini, A. C., Shehab, M., and Wede, J. Privacy policies for shared content in social network sites. *VLDB 19*, 6 (2010), 777–796.

[189] Stark, E., Hamburg, M., and Boneh, D. Symmetric cryptography in javascript. In *ACSAC 2013* (Dec. 2009), C. Payne and M. Franz, Eds., IEEE Computer Society, pp. 373–381.

[190] Stiegler, H. G. A structure for access control lists. *Software: Practice and Experience 9*, 10 (Oct. 1979), 813–819.

[191] Sturm, C., and Amer, H. The Effects of (Social) Media on Revolutions – Perspectives from Egypt and the Arab Spring. In *HCI 2013* (Jul. 2013), M. Kurosu, Ed., vol. 8006 of *LNCS*, Springer, pp. 352–358.

[192] Sun, Q., Simon, D. R., Wang, Y.-M., Russell, W., Padmanabhan, V. N., and Qiu, L. Statistical identification of encrypted web browsing traffic. In *IEEE Symposium on Security & Privacy 2002* (May 2002), IEEE Computer Society Press, pp. 19–30.

[193] Tierney, M., Spiro, I., Bregler, C., and Subramanian, L. Cryptagram: Photo privacy for online social media. In *COSN 2013* (2013), A. E. Abbadi and B. Krishnamurthy, Eds., ACM, pp. 75–88.

[194] Tootoonchian, A., Saroiu, S., Ganjali, Y., and Wolman, A. Lockr: better privacy for social networks. J. Liebeherr, G. Ventre, E. W. Biersack, and S. Keshav, Eds., ACM, pp. 169–180.

[195] Ugander, J., Karrer, B., Backstrom, L., and Marlow, C. The Anatomy of the Facebook Social Graph. *CoRR abs/1111.4503* (Nov. 2011).

[196] Urpalainen, J. An Extensible Markup Language (XML) Patch Operations Framework Utilizing XML Path Language (XPath) Selectors. In *RFC* (Sept. 2008), no. 5261, Internet Engineering Task Force, IETF.

[197] van den Berg, B., and Leenes, R. Audience segregation in social network sites. In *PASSAT 2010* (Aug. 2010), A. K. Elmagarmid and D. Agrawal, Eds., IEEE Computer Society, pp. 1111–1116.

[198] van den Berg, B., and Leenes, R. Masking in Social Network Sites – Translating a Real-World Social Practice to the Online Domain. In *Information Technology* (Jan. 2011), P. Molitor, Ed., vol. 53, De Gruyter, pp. 26–33.

[199] van den Berg, B., Pötzsch, S., Leenes, R., Borcea-Pfitzmann, K., and Beato, F. Privacy in social software. In *Privacy and Identity Management for Life* (Jul. 2011), J. Camenisch, S. Fischer-Hübner, and K. Rannenberg, Eds., Springer, pp. 33–60.

[200] van der Aalst, W. M. P. *Process Mining: Discovery, Conformance and Enhancement of Business Processes*, vol. 1. Springer, 2011.

[201] Vu, L., Aberer, K., Buchegger, S., and Datta, A. Enabling Secure Secret Sharing in Distributed Online Social Networks. In *ACSAC 2009* (Dec. 2009), C. Payne and M. Franz, Eds., IEEE Computer Society, pp. 419–428.

[202] Wang, Y., Norcie, G., Komanduri, S., Acquisti, A., Leon, P. G., and Cranor, L. F. "I regretted the minute I pressed share": a qualitative study of regrets on Facebook. In *SOUPS 2011* (Jul. 2011), L. F. Cranor, Ed., ACM, pp. 10–16.

[203] Warren, S. D., and Brandeis, L. D. The right to privacy. In *Harvard Law Review* (Dec. 1890), vol. 4, pp. 193–220.

[204] Washington Post. NSA slides explain the PRISM data-collection program. In *Washington Post* (Jun. 2013). http://wapo.st/J2gkLY. Accessed: Dec 3, 2014.

[205] Westin, A. *Privacy and freedom.* Atheneum, 1970.

[206] Whiting, D., Housley, R., and Ferguson, N. Counter with CBC-MAC (CCM). In *RFC* (Sep. 2003), no. 3610, Internet Engineering Task Force, IETF.

[207] Winter, P., and Lindskog, S. How the great firewall of china is blocking Tor. In *FOCI – USENIX 2012 Workshops* (Aug. 2012), R. Dingledine and J. Wright, Eds., USENIX Security.

[208] Wright, C. V., Coull, S. E., and Monrose, F. Traffic morphing: An efficient defense against statistical traffic analysis. In *NDSS 2009* (Feb. 2009), The Internet Society.

[209] Wu, H., and Preneel, B. AEGIS: A fast authenticated encryption algorithm. In *SAC 2013* (Aug. 2013), T. Lange, K. Lauter, and P. Lisonek, Eds., vol. 8282 of *LNCS*, Springer, pp. 185–201.

[210] Zhang, C., Sun, J., Zhu, X., and Fang, Y. Privacy and security for online social networks: challenges and opportunities. *IEEE Network 24*, 4 (Jul. 2010), 13–18.

[211] Zhao, Z., Ahn, G.-J., Hu, H., and Mahi, D. SocialImpact: Systematic analysis of underground social dynamics. In *ESORICS 2012* (Sept. 2012), S. Foresti, M. Yung, and F. Martinelli, Eds., vol. 7459 of *LNCS*, Springer, pp. 877–894.

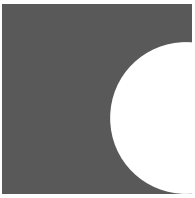
[212] Zhu, Y., Hu, Z., Wang, H., Hu, H., and Ahn, G. A collaborative framework for privacy protection in online social networks. In *CollaborateCom 2010* (Oct 2010), IEEE, pp. 1–10.

Curriculum Vitæ

Filipe Beato was born on September 17 in Lisbon, Portugal. He obtained a Master of Science degree in Computer Science from the University of Bristol, UK in 2008 and a Computer and Electrical Engineering Degree from the New University of Lisbon, Portugal in 2005.

In December 2008, he joined the COSIC (COmputer Security and Industrial Cryptography) research group as a research assistant to work on the EU-Primelife and NIST Biometric projects. He started his PhD in April 2011 sponsored by the FRH/BD/70311/2010 grant from the Fundação para a Ciência e Tecnologia (FCT). During his PhD he spent six months visiting Prof. Gene Tsudik at UC Irvine, US, in 2012, and several weeks visiting Prof. Mauro Conti at University of Padua, Italy, in 2013 and 2014.

Prior to COSIC, he spent some time as a research assistant at HP Labs Palo Alto, US, and HP Labs Bristol, UK, and worked as a software engineer at Critical Software, Portugal.



List of Publications

International Conferences

[1] Beato, F., Conti, M., Preneel, B., and Vettore, D., Virtualfriendship: Hiding interactions on online social networks. In *IEEE CNS 2014* (Oct. 2014), Y. Chen and R. Poovendran, Eds., IEEE, pp. 328–336.

[2] Beato, F., Cristofaro, E. D., and Rasmussen, K. B., Undetectable communication: The online social networks case. In *PST 2014* (Jul. 2014), Ali Miri and Urs Hengartner and Nen-Fu Huang and Audun Jøsang and Joaquín García-Alfaro, Eds., IEEE Computer Society Press, pp. 19–26.

[3] Borges, F., Martucci, L. A., Beato, F., and Mühlhäuser, M., Secure and privacy-friendly public key generation and certification. In *IEEE TrustCom 2014* (Sep. 2014), Y. Liu, Ed., IEEE, pp. 114–121.

[4] Beato, F., Ion, I., Čapkun, S., Preneel, B., and Langheinrich, M., For some eyes only: protecting online information sharing. In *ACM CODASPY 2013* (Feb. 2013), E. Bertino, R. S. Sandhu, L. Bauer, and J. Park, Eds., ACM, pp. 1–12.

[5] Simoens, K., Yang, B., Zhou, X., Beato, F., Busch, C., Newton, E., and Preneel, B., Criteria Towards Metrics for Benchmarking Template Protection Algorithms. In *IAPR ICB 2012* (Mar.-Apr. 2012), A. K. Jain, A. Ross, S. Prabhakar, and J. Kim, Eds., IEEE, pp. 498–505.

[6] Beato, F., Kohlweiss, M., and Wouters, K., Scramble! your social network data. In *PETS 2011* (Jul. 2011), S. Fischer-Hübner and N. Hopper, Eds., vol. 6794 of LNCS, Springer, pp. 211–225.

Peer-Reviewed Workshops

[7] Balsa, E., Beato, F., and Gürses, S., Why Can't Online Social Networks Encrypt?. In *W3C Workshop on Privacy and UserCentric Controls 2014* (Nov. 2014).

[8] Balsa, E., Beato, F., Diaz, C., and Preneel, B., Scramble. In *EFF Crypto Usability Prize (EFF CUP) Workshop 2014* (Jul. 2014).

[9] Beato, F., and Peeters, R., Collaborative Joint Content Sharing for Online Social Networks. In *IEEE SESOC 2014* (Mar. 2014), IEEE, pp. 616-621.

[10] Beato, F., Conti, M., and Preneel, B., Friend in the Middle (FiM): Tackling De-Anonymization in Social Networks. In *IEEE SESOC 2013* (Mar. 2013), IEEE, pp. 279–284.

[11] Beato, F., Kohlweiss, M., and Wouters, K., Enforcing Access Control in Social Networks. In *HotPets 2009* (Jul. 2009).

Book Chapters

[12] Beato, F., Borcea-Pfitzmann, K., Leenes, R., Potzsch, S., and Van den Berg, B., Privacy in Social Software. In *Privacy and Identity Management for Life* (2011), J. Camenisch, S. Fischer-Huebner, and K. Rannenberg, Eds., Springer-Verlag, pp. 33-60.

Miscellaneous

[13] Beato, F., Meul, S., and Preneel, B., Practical Identity Based Broadcast Encryption for Online Social Networks. In *COSIC internal report* (2014).

[14] Mavrogiannopoulos, N., Beato, F., and Reparaz, O., Secure and minimal security-modules in Internet security protocols. In *COSIC internal report* (2013).

[15] Beato, F., Borcea-Pfitzmann, K., Kuczerawy, A., Leenes, R., Olislaegers, S., Pekárek, M., Potzsch, S., Roosendaal, A., and Van den Berg, B., D1.2.1 - Privacy Enabled Communities. In *Primelife Deliverable* (2010), 218 pages.

[16] Beato, F., Borcea-Pfitzmann, K., De Ruiter, J., Leenes, R., Potzsch, S., and Wahrig, H., D1.2.2 - Privacy-enabled Communities Demonstrator. In *Primelife Deliverable* (2010), 23 pages.

FACULTY OF ENGINEERING SCIENCE
DEPARTMENT OF ELECTRICAL ENGINEERING
COMPUTER SECURITY AND INDUSTRIAL CRYPTOGRAPHY
Kasteelpark Arenberg 10, bus 2452
3001 Heverlee
filipe.beato@esat.kuleuven.be
http://www.esat.kuleuven.be