KU LEUVEN

ARENBERG DOCTORAL SCHOOL
Faculty of Engineering Science

Chaff-based profile obfuscation

Ero Balsa

Supervisor:
Prof. dr. Claudia Diaz
Prof. dr. ir. Bart Preneel, co-supervisor

Dissertation presented in partial fulfilment of the requirements for the degree of Doctor in Engineering Science (PhD): Electrical Engineering

December 2019

Chaff-based profile obfuscation

Ero BALSA

Examination committee:
Prof. dr. ir. Hugo Hens, chair
Prof. dr. Claudia Diaz, supervisor
Prof. dr. ir. Bart Preneel, co-supervisor
Prof. dr. ir. Frank Piessens
Prof. dr. ir. Fréderik Vercauteren
Prof. dr. Seda F. Gürses
(Technische Universiteit Delft)
Prof. dr. Helen Nissenbaum
(Cornell Tech)
Prof. dr. Carmela Troncoso
(École polytechnique fédérale de Lausanne)

Dissertation presented in partial fulfilment of the requirements for the degree of Doctor in Engineering Science (PhD): Electrical Engineering

December 2019

Uitgegeven in eigen beheer, Ero Balsa, Kasteelpark Arenberg 10, bus 2452, 3001 Heverlee (Belgium)

A Abuelo

Os corazós dos homes
que ao lonxe espreitan,
feitos están
tamén
de pedra.

Celso Emilio Ferreiro, *Longa noite de pedra.*

Abstract

Driven by the economics of surveillance capitalism, online service providers *profile* users to infer information about them, feeding automated decision-making processes such as targeted advertising or user experimentation that prompt grave concerns about people's privacy, autonomy and democratic sovereignty, among other rights. Market forces and lack of state intervention have forestalled the emergence of privacy-preserving alternatives, forcing individuals to choose between being profiled or relinquishing online services altogether.

In response to this failure, researchers and developers have advanced the deployment of *privacy enhancing technologies* (PETs) and, in particular, PETs that rely on *obfuscation*. Obfuscation tools enable users to protect themselves against profiling by degrading the data profilers collect about them, thereby reducing the amount of information profilers learn from those data. Of special interest is utility-preserving obfuscation, enabling users to escape trade-offs between utility and privacy.

In this thesis we contribute to the advance of privacy engineering through utility-preserving obfuscation. We propose a conceptual framework to distinguish between utility-preserving and utility-degrading obfuscation, and identify personal and social utility requirements that inform the choice of either type of obfuscation. We study *chaff* as a utility-preserving obfuscation method and provide a model and analytical framework to inform and assist the design and analysis of *chaff-based profile obfuscation tools*, with a focus on defence strategies and usability. We illustrate the design and analysis of chaff-based profile obfuscation through two use cases: web search and online communication. We examine existing chaff-based private web search tools, uncovering systematic design flaws; we study and assist the design of obfuscation tools that conceal communication patterns, attending in particular to their deployment on social networking sites. Lastly, we propose a new privacy design pattern to systematise profile obfuscation through chaff and discuss further implications of our research, exposing gaps and identifying promising research avenues.

Samenvatting

Aangedreven door de economische imperatieven van het surveillancekapitalisme stellen online dienstverleners profielen op van hun gebruikers om informatie over hen af te leiden. Deze informatie dient voor geautomatiseerde besluitvormingsprocessen zoals gerichte reclame of experimenten met gebruikers welke aanleiding geven tot ernstige bezorgdheden over privacy, autonomie en andere rechten. Marktkrachten en een gebrek aan regulering door overheden hebben de opkomst van privacybeschermende alternatieven echter verhinderd. Mensen kunnen er dus enkel voor kiezen om geprofileerd te worden of te stoppen met het gebruik van online diensten.

Als reactie hierop zijn onderzoekers en ontwikkelaars gestart met het opstellen van privacyverbeterende technologieën. Met deze die steunen op obfuscatie kunnen gebruikers zich beschermen tegen profilering door ruis toe te voegen aan hun data. De informatie die profilers hieruit kunnen afleiden wordt zo verminderd. Van bijzonder belang is nutsbehoudende obfuscatie, waardoor gebruikers afwegingen tussen nut en privacy kunnen vermijden.

Deze thesis draagt bij aan de vooruitgang van privacy engineering door nutsbehoudende obfuscatie. We stellen een kader voor om onderscheid te maken tussen nutsbehoudende en nutsverminderende obfuscatie, en we identificeren persoonlijk en maatschappelijk nut als doelstellingen voor beide types obfuscatie. We bestuderen chaff als nutsbehoudende obfuscatiemethode en ondersteunen op chaff gebaseerde obfuscatietools met een focus op verdedigingsstrategieën en gebruiksvriendelijkheid. We illustreren dit aan de hand van twee toepassingen: online zoekopdrachten en communicatie. Voor de eerste onderzoeken we bestaande tools op basis van chaff die privézoekopdrachten toelaten en brengen we zo systematische ontwerpfouten aan het licht. Voor de tweede bestuderen we het ontwerp van obfuscatietools die communicatiepatronen verbergen, in het bijzonder voor socialenetwerksites. Tot slot stellen we een privacy-ontwerppatroon voor om profielobfuscatie op basis van chaff te systematiseren en bespreken we de gevolgen en toekomst van ons onderzoek.

Acknowledgements

It crossed my mind that all my childhood gardeners were women.

—Derek Jarman, *Modern Nature*

My deepest gratitude goes to my supervisor, Prof. Diaz. I would not have come this far without her support, patience and immense wisdom. She challenged me to do better and, in the process, to grow a thicker skin. If I have become any good at this thing called academia, Claudia is the first to blame.

Prof. Gürses is second in line. Working along her side was among the best things that could ever happen to me. I look up to Seda not only as the scholar I can only dream of becoming, but as a phenomenal human being too.

Prof. Troncoso scared me to death as a fresh PhD student. She still does. But whenever I wonder if my work is any good, I find myself asking: *what would Carmela do?* She is my personal yardstick for excellence.

Claudia, Seda and Carmela represent a sort of academic Trinity to me.

COSIC would not be the great research group it is without the leadership of my co-supervisor, Prof. Bart Preneel. I feel privileged to have had the opportunity to work with and learn from him, to be part of his team. I told Prof. Helen Nissenbaum and I will reiterate here that this thesis would literally not have been possible without her groundbreaking contribution to the study of obfuscation. I cannot be more grateful and honoured that she has accepted to be part of my jury. My sincere thanks also go to Prof. Frank Piessens and Prof. Fréderik Vercauteren for joining the jury and providing invaluable feedback to improve this dissertation. My gratitude also goes to Prof. Hugo Hens, for chairing the jury, and Prof. Bettina Berendt, even if she could not finally make it.

The CACTOS *crew*, Prof. Alessandro Acquisti and Prof. Laura Brandimarte. They and Veronica Marotta, among others, made my stay in Pittsburgh a

delightful experience, while Filipe Beato provided much needed assistance with *Scramble!*, his magnum opus. From west to east, I am indebted to Prof. Zhiguo Wan for leading me down the differential privacy rabbit hole and being a stupendous host at Tsinghua University.

Back to COSIC, I would like to thank all my colleagues for creating a great work environment, but I must specially mention the following. As my only office mate for a good couple of years, Fatemeh Shirazi had to put up with me more than most —a feat for which she deserves nothing short of canonisation. Abdelrahaman Aly was the best next-door neighbour I could wish for. If the Catalonian separatists are truly committed to independence, they should send Cristina Pérez-Solà and Marc Juarez on a charm offensive; not only are they brilliant researchers, they are the sweetest too. Thanks to Charlotte Bonte, who she sent me an amazing first version of the *samenvatting*. Carl Bootland, Ilia Iliashenko, Elena Andreeva and others for great banter over lunch. Francesca Pichierri, *Paca*, she brightened my days. Bekah Overdorf, I could not get enough of her. Rafa Gálvez, Ren Zhang and everyone else at the privacy group. The SPION team, Svetla Nikova and the Witdom team. Special thanks also go to Péla Noë, Wim Devroye and Elsy Vermoesen for all their help and assistance.

In Brussels, Koen Verswijver, the commoner that keeps snatching royal titles —chiefly among them dancing queen and drama queen— helped and supported me like no one else. I also owe the final version of the *samenvatting* to him. Ben Stevenson, the eye-rolling master, made Brussels feel like home for the first time. Alessandro Mancosu, the Sardinian diva, pushing my buttons like no other. At the KBR, Giordano Bottecchia and Thomas Vilquin never missed an opportunity to en-*courage!*-me. Gino Marchal, Jordi Noguera and others.

In Compostela, *às nenas*, especially dRihanna and HC. L and Oliver. Time and again they made me feel as if I had never left home. Alex, the Madrid lifeline.

I am greatly indebted to my family for their love and support. *Obrigado, mãe.* My grandma and my aunt never ceased to confront me with those much-needed *"a ver quando terminas!"*. My godmother Tamara, the baking goddess, thanks for sharing the passion and leading the way. To Olalla, because she always wanted to know more than most about the PhD —and actually listened. *Avidly.* Iria, Breixo, Lúa. My godfather and great uncles. I love you all.

This took way longer than expected. My grandfather, José Balsa, left us before I could muster the energy to pull myself together and write up this thesis. I know he would be proud. This is for him.

Ero Balsa
Brussels, 3 December 2019

Abbreviations

ACA attack-centred analysis. 12, 64, 65, 78, 82–85, 110, 126, 127, 133, 144, 147, 161, 167, 241

ACP access control problem. 207–210

API application programming interface. 205

CA certificate authority. 202–204, 206

C&W chaffing and winnowing. 49, 50, 53, 89

CBOR chaff-based profile obfuscator. 13, 220, 223–238, 242

CBPWS chaff-based private web search. xxix, 12, 13, 111, 115, 116, 118–120, 122–128, 130–134, 136–138, 140, 142–148, 150, 151, 153, 154, 156–167, 176–178, 215–217, 223, 234, 237, 242

CCT censorship circumvention tool. 54

CJEU Court of Justice of the European Union. 246

CPC communication profile confidentiality. 13, 111, 112, 172, 178, 180, 181, 184, 192, 199, 207, 215–218, 237, 242

CPO chaff-based profile obfuscation. 116, 119, 172, 217, 229

cProto communication profile obfuscation tool. xxix, 172, 181, 182, 185, 186, 192, 200, 202, 207, 214–218, 242

DCA dummy classification algorithm. xxviii, 122–124, 129, 130, 133, 134, 137, 154, 162

DGS dummy generation strategy. 12, 13, 53, 54, 59, 62, 77, 85, 87–89, 91, 93–96, 101, 102, 110, 112, 119, 121–124, 126, 127, 130, 132, 134, 136, 137, 140, 144–151, 154, 156–159, 161–165, 167, 172, 176, 180–182, 184, 186–192, 200, 216, 218, 226, 227, 229, 232, 234, 241–243, 247

DNS Domain Name System. 52

DoS denial of service. 54, 58, 111, 226

DP differential privacy. 67, 68, 77

E2EE end-to-end encryption. 13, 41, 97–99, 170, 171, 173, 174, 176, 177, 192, 193, 200–207, 209, 210, 214, 215, 218, 242

ECC error correction code. 50, 51

EEE expected estimation error. 81–83, 102, 126, 127, 129, 130

ϵPI ϵ-profile indistinguishability. 67, 91, 93

EU European Union. 41, 221, 246

GDPR general data protection regulation. 4, 41, 221, 246, 247

HbC honest-but-curious. 52, 53, 106, 107, 109–111, 121, 163, 167, 177, 216, 241

HCI human-computer interaction. 8, 221, 230, 237

IBE identity-based encryption. 203, 204

ID identifier. 44, 203, 204

IM instant messaging. 41, 169, 171, 172, 216

IQP Intent-aware query-obfuscation for privacy-protection. 159, 160

LBS location-based service. xxvii, 34, 35, 44, 52, 80

LDP local differential privacy. 67, 69, 70, 130

MAC message authentication code. 49, 50, 177

MCA mechanism-centred analysis. 12, 64, 65, 76, 82–85, 110, 125, 127, 134, 147, 167, 179, 241

MDS media delivery service. 31, 32

MI mutual information. 73

MitM man-in-the-middle. 202, 203, 206

MPC multiparty computation. 33, 37–40, 244

ObPET obfuscation-based privacy enhancing technology. 5–10, 15, 16, 21, 240, 241

OQF-PIR Optimized query forgery for private information retrieval. 151, 152, 154, 157, 163

OS operating system. 50

OSN online social network. xxix, 57, 172, 179, 181, 182, 186, 188, 191–194, 199, 201–206, 214, 215

OTR Off-the-Record Messaging. 171, 205

P2P peer-to-peer. 73

PbD privacy by design. 41, 222

PDP privacy design pattern. 220, 222, 233, 235

PDS Plausibly deniable search. 138, 144, 145, 148, 156, 157

PDS privacy design strategy. 235

PEL profile exposure level. 73

PET privacy enhancing technology. 3, 5, 7, 8, 15, 165, 202, 219, 222, 223, 235, 237, 240, 242, 244, 246

PFA profile filtering algorithm. 122–124, 149, 154

PIR private information retrieval. 31, 32, 39, 40, 115, 240

PKG private key generator. 203

POT protective optimisation technology. 53, 104, 106, 167, 245–247

PRAW Privacy model for the web (CBPWS tool). 145–150, 156, 229

Proto profile obfuscation tool. xxiv, 9–13, 58–68, 70–72, 74–91, 94–96, 98–112, 118, 137, 138, 167, 172, 174–180, 206, 207, 215–220, 222, 223, 235–238, 241–247

QoS quality of service. 38, 106, 111, 178, 226

SCA semantic classification algorithm. xxviii, 124, 130, 134, 135, 141, 145, 146, 157, 159

SCS shortest common supersequence. 94

SD statistical distance. 68

SDC statistical disclosure control. 37

SNP social network provider. 13, 173, 192, 201–207, 210, 214, 215, 218

SNR signal-to-noise ratio. 94

SNS social networking site. 13, 112, 169, 170, 172, 173, 200–204, 207, 215, 218

SUS system usability scale. 208, 212

TBB Tor Browser Bundle. 99

TMN TrackMeNot. 5, 6, 9, 142–145, 148, 229, 240

TPET third-party E2EE tool. 10, 112, 173, 202, 205–207, 210–212, 214, 215, 218, 242

UDO utility-degrading obfuscation. 11, 22, 26, 34, 36, 46, 57, 58, 166, 240, 243

UI user interface. 226, 228

UPO utility-preserving obfuscation. 11, 22, 26, 30, 33, 34, 36, 37, 44–46, 53, 57, 58, 232, 240, 241, 243

UX user experience. 95, 99–101, 205, 227, 228

VoIP voice over IP. 2, 169, 170

VSS verifiable secret sharing. 203

Nomenclature

$*$	Operation to combine a real and a dummy sequence, i.e. $\mathrm{q} = \mathrm{r} * \mathrm{d}$
A	Alice
$\mathcal{A}$	Adversarial gain function
b	Adversary's belief
β	Indicates adversary's belief as subindex or superindex
B	Bob
c	Counting function
C	Carol
C	Channel capacity
C_∞	Min-capacity
d	Dummy action
δ	Differential privacy parameter
Δ	Length of quantisation step
D	R.v. over dummy actions

$\mathcal{D}$	Universe of dummies d
d	Ordered sequence of dummy actions $[d_1, d_2, \ldots, d_m]$
D	R.v. over sequences d
D	Universe of sequences d
$\hat{d}$	Action an adversary classifies as dummy
D_{KL}	Kullback-Leibler divergence
ϵ	Indistinguishability parameter
E	A graph's edges
E	Expected estimation error
f	Data processing function of user interest
F	R.v. over number of friends. As subscript or superscript it denotes friends.
F	Frank
$\mathcal{G}$	Information gain
g	Data processing function of adversarial interest, e.g. profiling function
G	A graph
h	Data processing function of social interest
H	Shannon's entropy
H_∞	Min-entropy
I	Mutual information

ϑ	A vault
κ	Relationship classification function
l	Location
ℓ	Distance
$\mathcal{L}_\infty$	Min-entropy leakage
m	Number of messages
M	R.v. over number of messages
N	Deniability
o	Answer, response to a user action; system or function outcome
$\mathcal{O}$	Universe of system outputs/responses to user actions
o	Sequence of system outputs/responses
O	Universe of system output sequences
Ω	Obfuscator
P	Probability
$\mathcal{P}$	Program
q	Action, event, protocol message (e.g. query, message)
Q	R.v. over actions q_i
$\mathcal{Q}$	Universe of events q (from Chapter 3 onwards) Program (only Chapter 2)

q	Ordered sequence of actions $[q_1, q_2, \ldots, q_m]$
Q	R.v. over ordered sequences q
$\mathbf{Q}$	Universe of sequences q
r	Real action
R	R.v. of real actions
$\mathcal{R}$	Universe of r
r	Ordered sequence of real actions $[r_1, r_2, \ldots, r_m]$
R	R.v. of sequences of real actions
$\mathbf{R}$	Universe of sequences r
$\hat{r}$	Action an adversary classifies as real
$\hat{\text{r}}$	Sequence that the adversary classifies as real
ρ	Rate of dummies
s	A secret
S	Set
S_C	Cosine similarity
S_q	Set of queries
σ	Number of samples
t	Time
T	Period of time
τ	A topic
T	Universe of topics
θ	A threshold

u	User utility function
U	Undetectability
v	A user
V	Set of users, a (social) graph's vertices
w	Dummy profile component
$\mathbf{w}$	Dummy profile
$\mathbf{x}$	User real profile
X	R.v. over real user profiles
$\mathcal{X}$	Universe of $\mathbf{x}$
$\hat{\mathbf{x}}$	Profile that the adversary recovers
$\hat{X}$	R.v. over filtered profiles $\hat{\mathbf{x}}$
x	Real profile component
χ	R.v. over profile components
$\mathbf{y}$	User's observed profile
Y	R.v. of observed profiles
$\mathcal{Y}$	Universe of $\mathbf{y}$
$\mathbf{y}^t$	Target profile
y	Observed profile component
υ	R.v. over observed profile components
z	Number of posts (public messages)
Z	R.v. over number of posts (public messages)

Contents

List of figures

List of tables

Chapter 1

Introduction

> *You're all serfs. [...] The data is all out there, they take your stuff for free and monetise it for huge margins, they take over your life.*
>
> —Steve Bannon.

> *I was never one to acquiesce very easily to systems that felt wrong to me.*
>
> —Anohni.

> *Disobedience, in the eyes of anyone who has read history, is man's original virtue. It is through disobedience that progress has been made, through disobedience and through rebellion.*
>
> —Oscar Wilde, *The soul of man under socialism.*

> *Bring the noize when we run upon them!*
> *Bring the noize when we run upon them!*
>
> —M.I.A., *Matangi.*

Technological advances and innovations in the second half of the 20th century, as well as the cheap, mass production of devices such as personal computers and smartphones have ushered in a *digital revolution*, conducing to digital technology mediating an increasing number of human activities.

In 2019, we browse the Internet for both work and leisure, communicate over email and instant messaging, connect with people on social networking sites,

order food and taxis with our smartphones, rely on *"intelligent"* assistants for all sorts of tasks and manage our finances and tax payments online, among an ever widening range of activities.

The increasing digitisation of human life has many advantages. Computers dramatically facilitate information processing and storage, easing and simplifying tedious and complex tasks such as keyword search and document indexing. Computer networks further enable instantaneous and on-demand transmission of information, as the web, email and voice over IP (VoIP) exemplify.

Yet at the same time, the economic apparatus that has harnessed and steered the development of this digital revolution has led to the emergence of a *surveillance* and *datafied* society [362, 581]. Tech giants such as Google and Facebook have fostered and relied on the mass adoption of various digital services to build a system of behavioural data acquisition and exploitation whereby they monetise user data through what Zuboff refers to as *behavioural futures markets* [581]. Service providers entice people to use various services so as to extract their usage patterns, using such patterns to build individual *profiles* of user behaviour. User *profiling* enables providers to make predictions about users' future behaviour, selling those predictions as a product; e.g. Google and Facebook analyse users' data and metadata to predict which goods, services or topics users are most likely to be interested in, then sell those predictions to advertisers seeking to target their products or their ideas to potential buyers.

Profiling can bring many benefits to users, such as service improvement and personalisation, but it also poses multiple dangers, both for individuals and society at large. Beyond the obvious privacy problems that profiling poses, as profilers learn all sorts of sensitive and private information about the users they profile [497], profiling further informs (automated) decision making processes, prompting multiple deleterious outcomes over which users have no knowledge or control over [404]. Unauthorised predictions and inferences [144, 334], discrimination —racial, economic or otherwise— [584, 273, 513], social sorting [126, 579], filter bubbles [407, 428], manipulation [334, 511] or experimentation [65, 102, 184, 257] are some of the dangers that users face as a result of online profiling.

More fundamentally, profiling and the decision making processes that profiles inform alter the very fabric of society as, similarly to the effect Solove attributes to government surveillance, they *"not only frustrate the individual by creating a sense of helplessness and powerlessness, but also affect social structure by altering the kind of relationships people have with [those] that make important decisions about their lives. [...] The harms are bureaucratic —indifference, error, abuse, frustration, and lack of transparency and accountability"* [498]. Moreover, under the accumulation logic of *surveillance capitalism*, knowledge

and power is concentrated on a few surveillance capitalists that *"through illegible mechanisms of extraction, commodification, and control [...] effectively exile persons from their own behavior"* [580, 581], impairing *"parity of participation in social life"* and *"reinforcing [...] material and cultural power imbalances"* [83], thereby threatening *"a diverse range of intersecting rights, including autonomy, fairness, equality, democratic sovereignty, due process, and property"* [125].

In spite of these dangers, so far regulation of data acquisition and processing has been lax [144, 288, 404]. This is specially true and relevant for the US, as most tech companies are US-based and, therefore, appropriate regulation could have prevented the rise and dominance of business models based on online profiling [459, 581]. Scholars have attributed the reluctance of US regulators to restrain profiling practices to a historical juncture where several elements converged to prevent intervention. These elements include the global hegemony of neoliberal policies, pressing for deregulation and industry *self-regulation*; a global economic slowdown that motivates investors to find and encourage new, profitable markets; and the opportunistic exploitation of surveillant assemblages as part of the War on Terror that the US launched after the 9/11 attacks [77, 282, 393, 499, 581]. Consequently, US law has so far largely left the profiling industry to *self-regulate* itself through mechanisms of *notice and choice* whereby users must choose whether they consent to profiling or not [288].

Self-regulation has however long been considered a failure [233, 288, 341, 404]. Users' unawareness of privacy problems, feelings of having *'nothing to hide'* [7, 77, 317, 388, 498] or time-inconsistent trade-offs between immediate gratification and future harms, explain the dichotomy between users' attitude and behaviour in terms of privacy protection [5, 8], i.e. users report to care about privacy yet have adopted privacy-invasive systems en masse.

Moreover, tech companies' lack of incentives to stop profiling practices —as a result of the economic imperatives that capitalism imposes— in addition to weak regulation and lack of user awareness —which translates into little demand for privacy preserving systems— has led to no privacy-friendly alternatives entering the market [287, 457, 504].

Alternative systems and solutions that prevent user profiling do however exist, as the extensive literature on privacy enhancing technologies (PETs) demonstrates [115, 240]. The PETs research community of privacy experts have proposed and sought ways to redesign or adapt systems to prevent the ability of service providers to extract data and profile users while, at the same time, ensuring no or minimal degradation on the utility users derive from such systems. PETs represent systems and tools that reimagine or patch privacy-invasive systems in order to minimise users' privacy risks, e.g. alternative

implementations of web search and social networking platforms that would enable users to bypass Google and Facebook [161, 170, 508, 519].

The prohibitive social costs that giving up online services involves and the lack of realistic alternatives explain users' inability to escape profiling online [77, 404], with people voicing resignation and hopelessness in the aftermath of the Snowden revelations [168, 388, 418]. Both users and scholars agree that the market has not provided realistic alternatives to avoid profiling, that voluntariness is an *"illusion"* as users constantly face *take-it-or-leave-it* decisions; i.e. they either agree to profiling or renounce the use of online services [77, 288], while many of these services are increasingly necessary to perform vital tasks such as looking for a job or connect with friends and family.

In the EU, the recent general data protection regulation (GDPR) holds promise, specially because of the rules Article 22 sets on profiling and automated decision making [288, 494]. Legal scholars however differ as to the extent to which they believe the GDPR will *actually* limit current profiling practices [95, 381, 571, 583]. As Hoofnagle et al. argue in the context of service providers that require users' consent to profiling in exchange for service provision, *"under which circumstances take-it-or-leave-it choices are still acceptable has to become clear from enforcement"* [288].[1] Other scholars have advanced that owning to the EU's influence and bargaining power, the GDPR will eventually lead to a new, more stringent, *"global privacy standard"* [66, 238, 459].

In the meantime, however, users who wish to evade profiling are left with little choice. Either consent to profiling, fuelling the surveillance capitalist juggernaut, or give up on services which have become essential for the successful development of life and participation in society [581]. This state of user vulnerability and lack of alternatives has motivated a stream of work and research on tools that people can deploy on top of the services they use to protect themselves from privacy-invasive practices, as opposed to giving up on those services entirely. These tools follow a "take-matters-into-your-own-hands" philosophy, empowering users to defend themselves against profiling [99]. Prominent examples include privacy technologies such as Tor or PGP. Tor is an anonymous communication network that enables users to browse the web anonymously, thereby preventing profiling by rendering users' web transactions *unlinkable* [176]. People can download the Tor browser and use it (mostly) as they would use any other browser [520]. PGP is an email encryption tool that enables users to encrypt their emails to prevent email providers or network eavesdroppers from accessing their emails'

[1] Hoofnagle et al. also report that *"[t]he day the GDPR became enforceable, Max Schrems complained to Data Protection Authorities about the take-it-or-leave-it practices of Google, Instagram, WhatsApp, and Facebook"* [288]. At the time of writing, no resolution has been adopted on this complaint.

content [480]. Other popular PETs include various anti-tracking tools such as Ghostery, uBlock origin or NoScript [369, 471].

These PETs rely on a variety of mechanisms and strategies to protect user privacy. Among them, a particular subset relies on what is loosely termed *obfuscation*. A paradigmatic example of such a tool is TrackMeNot (TMN), a browser plug-in that generates fake queries on behalf of web search engine's users [523]. By injecting fake queries, TMN seeks that the search provider cannot distinguish a user's queries from the automatically generated and, thereby, that the provider cannot determine the user's search interests, as fake queries *degrade* or pollute the profiles providers build.

In many scenarios, obfuscation-based privacy enhancing technologies (ObPETs) like TMN represent a better or the only alternative to other PETs for a variety of reasons. Technical requirements and constraints that we examine in Sect. 2.3 prevent or discourage the use of other PETs, leaving ObPETs as the only viable option. Moreover, encryption tools such as PGP or anti-trackers such as Ghostery and NoScript rely on a set of complex mechanisms to achieve the protection they provide, mechanisms that the average user generally does not understand or is unaware of. PGP is notoriously hard to use and understand [551]. Similarly, researchers have reported severe user misconceptions about how anti-tracking tools work [471]. The technical complexity underlying many PETs renders them inaccessible or incomprehensible to the average user. In contrast, ObPETs like TMN relate to a far more approachable and easy to understand concept, one that users instinctively rely on for their protection online, namely, that of *privacy lies*.[2]

Previous work has shown that, among other *privacy-protective behaviours* online, users rely on data degradation or privacy lies, namely, they choose to strategically provide inaccurate or false information when they face requests that they deem intrusive, abusive or unnecessary [422, 466]. Two classic scenarios where users adopt this practice are inquisitive sign-up forms and publicly exposed social media profiles, which they fill with inaccurate or false information [224, 466]. ObPETs adopt the generation of privacy lies as a defence mechanism and *automate* it, i.e. they remove the need of user intervention, they free users from having to generate privacy lies themselves. Hence, instead of relying on technical information and methods that users ignore or have misconceptions about, ObPETs leverage a protection mechanism that users

[2]Obfuscation is too broad and vague a term to precisely delimit a particular set of ObPETs, as there is no universally-accepted formal definition of or consensus on what constitutes obfuscation [100]. We formalise in Chapter 2 the type of obfuscation tools we study in this thesis, namely, those that rely on obfuscation understood as *data degradation*; e.g. while some authors consider Tor's onion routing mechanism as obfuscation [100], it does not fall within the obfuscation as *data degradation* type of tools that we focus on in this thesis.

can easily understand and relate to, which in turn renders these tools more accessible to the average user [99].

Proof of obfuscation's accessibility and intuitiveness is the number of software developers, designers and artists with no expertise in computer security or privacy engineering that have resorted to obfuscation as a means to develop interventions against profiling. Examples of such interventions include web browser extensions such as *I like what I see* and *GoRando*. The former *"automatically clicks all 'Like' buttons on Facebook"*; the latter *"chooses one of the six 'reactions"'* every time a user clicks "Like" on Facebook [101, 293].

Indeed, both ObPETs users and designers may resort to obfuscation to *"avoid or neutralise a lurking but ill-understood threat"* [99], i.e. in the face of unknown, complex threats, both users and designers resort to intuitive, easy-to-understand defence strategies. A paradox however, as ultimately obfuscation represents a comprehensible defence to inherently complex privacy threats, leading both users and designers to misunderstand or overestimate the protection that ObPETs afford [42, 294].

It is because of this dual asymmetry, both in terms of power —as users have little control over profiling— and knowledge —as users do not know how they are being profiled or the consequences thereof— that motivates Brunton and Nissenbaum to refer to obfuscation as a *weapon of the weak*, borrowing from Scott's work on *everyday forms of resistance*, namely, *"stratagems deployed by a weaker party in thwarting the claims of an institutional or class opponent who dominates the public exercise of power"* [99, 478].

Data degradation or inaccuracy enables users to *modulate consent* in the absence of meaningful opt-out policies, this is, in the absence of granular policies that enable individuals to determine for which uses profilers can and cannot collect and process their data [99, 224]. Users can strategically obfuscate the data profilers collect to maximise utility in terms of what they consider the *primary use of data* and minimise utility in terms of the *secondary uses* they wish to preempt, e.g. TMN obfuscates users' search patterns in a way that allows users to obtain the search results they expect (the primary use) while preventing profilers to determine users' interests, which TMN users do not which to reveal (secondary use). As González Fuster argues, obfuscation *"can be interpreted as a preventive limitation against undesired secondary processing, a natural obstacle to further processing or a 'sticky policy' promoting compliance with the purpose limitation principle"*, enabling individuals to *"preserve [their] informational autonomy"* [224].

Howe further recognises three intertwined aims of ObPETs: protection, as they prevent profilers from gaining accurate information about users; expression, as

they enable users to openly contest and protest against profiling practices; and subversion, as they pollute profilers' data with noise —thereby crippling the economic machinery around profiling.

1.1 Motivation

Despite obfuscation being an intuitive and approachable solution, designing effective ObPETs to protect users against profiling is far from trivial. Sound ObPET designs require that we go beyond *adding some random noise*. ObPETs need to withstand strategic attacks, which means that we must subject them to proper security analyses. As Kocher et al. point out in the context of securing cryptosystems against side channel attacks, *"[d]esigners and reviewers must approach [...] obfuscation with great caution, however, as many techniques can be used to bypass or compensate for [it]"* [327]. Eliciting the privacy properties ObPETs must guarantee, determining how ObPETs must obfuscate, namely, how to introduce noise in the data profilers collect, evaluating the level of protection these tools offer, under which conditions, and in light of the epistemic asymmetry between ObPET designers and profilers; all these tasks require a systematic approach to ensure sound ObPET's design and evaluation.

Obfuscation itself as a concept is vague and often loosely used to denote a panoply of PETs that have little in common beyond an underlying notion of producing *"misleading, ambiguous and plausible but confusing information as an act of concealment or evasion"* to protect users' privacy [99, 100]. Whereas in terms of ethical and political theory it may be useful to lump these tools together under the category of obfuscation technologies, in terms of privacy engineering these tools rely on different principles and protection mechanisms that benefit from a separate conceptualisation and study of their own, further enabling a dialectics across different forms of obfuscation. Identifying and understanding the types of obfuscation privacy engineers use, their underlying protection mechanisms, the conditions that enable or prevent their deployment as well as the advantages and disadvantages of obfuscation compared to alternative mechanisms are some of the key pillars in the art and practice of engineering privacy through obfuscation. However, answers to these and other questions remain scarce or nonexistent in the literature.

In particular, we are interested in obfuscation mechanisms that enable users to protect themselves against profiling at no utility costs. In other words, obfuscation mechanisms that enable users to preserve all the utility they obtain from the *primary use of data* while preventing or frustrating any *secondary uses of data*. An engineering practice of privacy protection through obfuscation

needs to understand which mechanisms enable users to achieve that dual goal and under which conditions.

Brunton and Nissenbaum ask *"Is it possible to create a meaningfully quantified science of obfuscation?"* [99]. Privacy engineers need analytical frameworks and methodologies to design and evaluate obfuscation tools, to measure a tool's effectiveness, the level of privacy protection a tool offers. This requires the operationalisation of abstract privacy requirements into technical constraints and the selection of privacy measures that enable the quantification of privacy properties. It also requires modelling the set of assumptions under which such properties hold, especially those relating to the adversaries a tool must protect against. All these elements contribute to the advance of the *quantified science* of obfuscation that Brunton and Nissenbaum inquire about.

The design of privacy technologies requires solutions specifically tailored to particular contexts and users' needs [400]; no two obfuscation tools are likely to be the same. And yet, an engineering of obfuscation benefits from the abstraction of methods and solutions that privacy engineers can generally resort to as a guide in the design process. Engineering privacy through obfuscation requires a blueprint of the fundamental elements and concepts involved in the design of obfuscation tools. Such engineering practice includes not only the obfuscation mechanisms themselves but also the mechanisms that govern users' interactions with the tools that implement such mechanisms, namely, the human-computer interaction (HCI) problem. Understanding which design principles help individuals adopt and effectively utilise obfuscation tools, maximising their privacy protection, is a key aspect to ObPETs design.

Obfuscation practice can also benefit from a common language not only across obfuscation technologies, but also with respect to other privacy technologies. Understanding how concepts and methodologies that underlie other PETs can be adapted and repurposed in the context of ObPET design further promotes a dialogue across the privacy engineering community and contributes to advancing the discipline overall.

Lastly, even if obfuscation technologies enable users to protect their privacy online, it is unclear the extent to which ObPETs can insulate users against the unexpected, undesirable outcomes of profiling and under which conditions. Obfuscation tools can prevent an adversary from learning sensitive information about users, but not from using *misleading, ambiguous* or *confusing* information to make decisions that affect them. Moreover, we need to determine whether ObPETs can effectively undermine the conditions that underpin profiling practices in the first place. Howe identifies a trade-off between three ObPETs aims, namely, protection, expression and subversion [293]. We need to reason about the conditions under which ObPETs maximise either of those aims or

strike a balance between them, whether obfuscation technologies are ultimately an adequate instrument to resist and challenge surveillance capitalism.

1.2 Approach

Our first objective is to delimit the conceptual boundaries across obfuscation tools in computer security and privacy and develop a conceptual framework that enables us to focus on a particular subset of obfuscation tools, namely, those that protect users' privacy against profiling without taking a toll on user utility. To do so, we review previous work on *obfuscation* across the computer security and privacy engineering literature. We provide a critical analysis of the meanings researchers within particular subcommunities in computer security and privacy attribute to obfuscation and examine the privacy goals they pursue and the obfuscation methods they utilise.

Then, to further establish a conceptual separation across types of ObPETs, we rely on an abstract model of obfuscation as *data degradation*. This model enables us to distinguish between tools like, on the one hand, TMN or differentially private mechanisms and, on the other hand, Tor or tools that leverage *traffic morphing* [99]; thereby contributing to a first technical separation across ObPETs that enables us to further narrow down our subject of study to focus on the particularities of tools that rely on obfuscation as data degradation.

Furthermore, within this abstract model we define a set of concepts that enable us to distinguish between two types of obfuscation tools, namely, those that degrade utility to protect users' privacy and those that do not. This conceptualisation enables us to examine the conditions that call for either type of obfuscation tools and, as a result, to formulate a conjecture on the utility requirements that either require users to *necessarily* trade-off utility for privacy or that enable them to protect their privacy without incurring any utility loss.

Our second objective is to provide an analytical framework that enables us to systematise the study of obfuscation tools that rely on *chaff* to protect users from profiling at no cost for utility. To do so, we rely on classic security modelling to provide an abstract model of profile obfuscation tools (Protos) and of the reference adversary or *profiler* that Protos protect against. Moreover, we revisit privacy measures that previous authors have relied on to measure the level of protection that obfuscation tools afford. We categorise these measures according to their underlying adversarial assumptions and level of abstraction, thereby providing designers and reviewers with an analytical toolbox to design and evaluate Protos. In addition, we leverage classic computer security modelling to examine the set of adversarial assumptions that support each of

the three ObPETs' aims that Howe identifies —namely, protection, expression and subversion.

Our third objective is to develop a conceptual framework to assist the design of Protos. We leverage classic computer security and privacy engineering modelling to inform the selection and operationalisation of the privacy properties Protos must satisfy. Moreover, we rely on previous expert literature on Protos to propose a set of elementary conceptual tools that assist and inform tool design.

Our fourth objective is to further assist Proto design in terms of usability, namely, determine key principles that enable individuals to adopt and use Protos effectively. Because of the lack of studies that examine the usability of Protos, we revisit the literature on the usability of privacy and the behavioural economics of privacy to determine which existing principles and findings we can extrapolate to Proto design. In addition, we run a user study to test the reproducibility of some of these findings and assess the viability of third-party, end-to-end-encryption tools (TPETs) as a user-friendly platform on top of which designers can build Protos.

Our fifth objective is to demonstrate the viability and adequacy of the analytical and conceptual frameworks we propose, this is, to demonstrate how to apply these frameworks in practice. To do so, we implement the general model we propose in two particular scenarios, namely, web search and online communication —inspired by the flagship services of prominent surveillance capitalists Google and Facebook, respectively. We illustrate how to select and operationalise privacy properties in each of these contexts. We leverage our conceptual and analytical frameworks to revisit and critically examine previous Protos' designs. Moreover, we illustrate how to compute privacy measures in practice, proposing a set of simplifications that enable designers to speed up and thereby lower the cost of Protos' evaluation.

Our sixth objective is to set the basis of a design methodology for chaff-based profile obfuscation as well as to encourage a common language between and a dialogue with privacy engineers across a range of domains and techniques. To that aim, we resort to the fields of software and privacy engineering and their work on privacy design patterns. We propose a new privacy design pattern, namely, a *chaff-based profile obfuscator* and integrate previously proposed privacy design patterns within a hierarchy of related patterns.

Our seventh and last objective is to identify major gaps in our work as well as future avenues of inquiry. We reflect on the findings and limitations of our work, discuss the implications of Protos deployment and identify promising lines of future work.

1.3 Outline and contributions

In Chapter 2 we introduce the concept of obfuscation, highlighting how its multiple meanings and interpretations render it a vague and ambiguous concept to work with. To elucidate the particular type of obfuscation we focus on in this thesis, we provide an overview to the various understandings and uses of *obfuscation* across security and privacy research. We distinguish three subcommunities with their own particular understanding of obfuscation, namely, software engineers, whose definition relates to the *complexity* of reverse-engineering *obfuscated code*; cryptographers, whose definition relates to the *indistinguishability* between obfuscated code of two programs with equivalent functionality, and privacy engineers, whose definition relates to the *inaccuracy* and *imprecision* of data.

As privacy engineers, our working definition of obfuscation belongs to the last category. Hence, in Chapter 2 we propose an abstract model of data obfuscation and introduce the concepts of personal utility, privacy loss (or adversarial gain) and social utility. Through the concepts of personal utility and privacy loss we distinguish between two types of obfuscation, namely, utility-preserving obfuscation (UPO) and utility-degrading obfuscation (UDO). Both UPO and UDO seek to preserve user utility while minimising adversarial gains, i.e. privacy losses. However, we argue that UDO is inadequate or suboptimal to address trade-offs between personal utility and privacy in the absence of social utility requirements, as personal utility alone does not require information disclosure to adversarial parties. Conversely, as social utility requires disclosing information to adversarial parties, we argue that only UDO can provide robust privacy guarantees against adversaries with arbitrary background knowledge —by trading off social utility for privacy.

We further outline the technical constraints that motivate the use of obfuscation as opposed to cryptographic anonymity tools, highlighting the role of uncooperative providers that refuse to deploy cryptographic systems and the clash between anonymity and services that require persistent, even if pseudonymous, identities.

We introduce UPO through *chaff* and provide an overview of the use of chaff in security and privacy research. We distinguish between different types of chaff, identifying the goals, protection methods and adversarial assumptions underlying their deployment.

In Chapter 3 we introduce *"Protos"*, namely, chaff-based obfuscation tools that seek to prevent privacy losses that *profiling* causes. We introduce an abstract model that describes the profiling process as well as the type of adversary Protos respond to. In addition, we introduce a set of measures to assist the design and

evaluation of Protos, distinguishing two types of measures: those supporting a mechanism-centred analysis (MCA) and those supporting an attack-centred analysis (ACA). We propose this categorisation to distinguish between measures that abstract away from particular adversaries and attacks, measuring intrinsic properties of the Proto, i.e. as a function of its inputs and outputs alone (MCA), and measures that capture the performance of a particular adversary and attack, i.e. that rely on details external to Protos' design (ACA).

Moreover, we provide an overview to key aspects in Proto design, with a focus on Protos' dummy generation strategies (DGSs) and usability. A Proto's DGS is responsible for the generation of dummy, fake activity to protect users against profiling. Protos require usability to ensure that individuals are able to benefit from the antiprofiling protection they offer. Due to the shortage of usability studies of Protos (motivated in turn by a dearth of Protos implementations) we review the state-of-the-art on usability of security and privacy tools and propose a set of recommendations towards the design and implementation of usable Protos.

We finalise Chapter 3 with a discussion on the implications of assuming an honest-but-curious service provider as the reference adversary Protos defend against. We examine the impact of both naive and active adversaries, highlighting Protos' inability to both avoid unintended consequences from profiling and prevent an active adversary from degrading the quality of service Protos' users expect.

Chapter 3 draws on, extends and develops some of our previous findings and results [39, 41, 42, 261].

In Chapter 4 we study obfuscation tools that attempt to thwart profiling by online search engine providers. We instantiate the general Protos model to chaff-based private web search (CBPWS), illustrating how to operationalise relevant privacy properties to measure the effectiveness of CBPWS tools. Then, we illustrate how to leverage the CBPWS analysis framework by critically examining existing CBPWS tools. We uncover a series of vulnerabilities that render these tools ineffective and further articulate and systematise the culprits underlying these flawed designs. Through Chapter 4 we highlight, among several others, two main Protos' design challenges: one, the complexity of DGS design when the universe of user actions is too large to delimit and define and user behaviour is difficult to predict; two, the lack of information on an adversary's profiling practices.

Chapter 4 heavily draws on previous results published by Balsa et al., which we extend and elaborate on [42].

In Chapter 5 we propose a second use case and study obfuscation tools that attempt to thwart profiling in online communication services. We instantiate the

general Protos model to communication services, define *profile confidentiality* as the reference privacy property communication Protos are after, and operationalise profile confidentiality through the information-theoretic measures we propose in Chapter 3.

Online communication services enable us to highlight two main differences between the deployment of Protos for CBPWS and communication profile confidentiality (CPC). These differences further enable us to emphasise that despite the commonalities across Protos, each service and context requires tweaks and tailored solutions of its own.

Based on previous results by Balsa et al., we further propose a set of techniques to empirically compute the level of profile confidentiality that Protos afford to users who communicate on social networking sites (SNSs) [41]. We draw on a separate set of previous results by Balsa et al. to identify sources of side-channel information leakage in SNSs, informing the selection of metadata a Proto's DGS must consider [40].

We focus on the deployment of Protos in online social networks and include and update our previous analysis of the role social network providers (SNPs) play in the deployment of end-to-end encryption (E2EE) in SNSs, as encryption is key for Protos to guarantee content indistinguishability, a core DGS design requirement [38]. In addition, we resort to a previous user study by Balsa et al. to frame and discuss the dichotomy between tool integration and user engagement in the context of Protos for CPC in SNSs [39], showing that our results align and support previous and subsequent findings, namely, that users overwhelmingly prefer integrated privacy solutions.

In Chapter 6 we propose *chaff-based profile obfuscator* (CBOR), a new *privacy design pattern*. CBOR recasts the general Protos model we introduce in Chapter 3 as an abstract solution that privacy engineers can resort to to develop chaff-based profile obfuscation tools against profiling. Through CBOR we contribute to the advancement of privacy engineering as a discipline by, first, encouraging researchers and developers across domains to think about Protos collectively and, secondly, by developing a common language not only for Protos' developers but also for privacy engineers at large.

In Chapter 7 we conclude by revisiting the goals we have stated in this section and outlining the extent to which we have fulfilled them. Moreover, we discuss further implications of our research, exposing gaps and promising lines of research.

Chapter 2

On obfuscation

> *Obscurity wraps about a man like a mist; obscurity is dark, ample, and free; obscurity lets the mind take its way unimpeded. Over the obscure man is poured the merciful suffusion of darkness.*
> *None knows where he goes or comes. He may seek the truth and speak it; he alone is free; he alone is truthful; he alone is at peace."*
>
> —Virginia Woolf, *Orlando: A biography.*

In this chapter we provide a definition of what we refer to as *obfuscation* and *obfuscation technologies*. In particular, we focus on what we call obfuscation-based privacy enhancing technologies (ObPETs), i.e. technologies whose goal is to solve a privacy problem by relying on obfuscation. We may define a *privacy enhancing technology* (PET) as any type of *"technical means for protecting users' privacy"* [547]. The meaning of *obfuscation*, however, varies across research (sub)communities, meaning there is no standard definition or consensus on what obfuscation technically means or entails. There is a panoply of technologies and techniques whose proponents say to rely on or provide obfuscation. From privacy-preserving location based services [30, 186, 488] and private web search tools [42] to anonymous communication systems such as Tor [100] or directed-access databases [401] and software obfuscation [138, 137], these systems and tools rely on *some form* of obfuscation, yet they differ in the protection techniques they use and the privacy properties they pursue or guarantee. This wide variety of tools attests to the lack of a unique understanding or definition of what obfuscation involves or what an obfuscation tool is. Hence, if we define ObPET as tools that provide privacy through *obfuscation*, we need to consider a range of privacy tools that differ widely from each other.

Obfuscation is thus too vague a word to delimit the conceptual boundaries of the type of technologies we aim to study. Let us consider the definitions of *'obfuscation'* the Oxford dictionary[1] provides:

1. *The action of obfuscating something or someone; the condition of being obfuscated.*
    a. *Darkening or dimming of colour, light, or the sight; an instance of this.*
    b. *Concealment or obscuration of a concept, idea, expression, etc.*
    c. *Confusion of the mind, understanding, etc.; stupefaction, bewilderment.*
2. *Something that darkens or obscures a situation, facts, etc.; an instance of darkening or obscuration.*

Understanding obfuscation as *concealment* implies that any technology that conceals or hides information is obfuscation. Hence, according to this definition, encryption technologies are obfuscation. Similarly, anonymous communication systems seek to conceal the relationship between senders and receivers in a communication channel, leading to some scholars categorising them as obfuscation tools [100]. In fact, if we generalise this reasoning any technology that models privacy as confidentiality or hiding [259] may be considered an ObPET.

On the other hand, since inaccuracy and imprecision are *'something that darkens or obscures'* and *'an instance of darkening or obscuration'*, privacy technologies that intentionally make data less accurate or precise may also be considered obfuscation tools, as is the case of tools that leverage intentional data quality degradation as a *means* to provide location privacy [30, 186, 488], context privacy [555] or privacy-preserving collaborative filtering [70, 437].

In this thesis we focus on a particular subset of obfuscation tools that, because of the commonalities they share and their differences with other tools, we consider a category of their own. We note that this is not an attempt to limit or constrain the notion of obfuscation, but to illustrate the various understandings of what *obfuscation* means and entails across research communities.

In this chapter we firstly provide in Sect. 2.1 an overview of different understandings of what *obfuscation* means across computer security and privacy research. In Section 2.2, we introduce a conceptual framework for obfuscation tools that rely on *data degradation* to protect privacy. In Section 2.3 we examine

[1] Retrieved on 1 March 2018 from `www.oed.com`.

technologies other than those that rely on obfuscation as data degradation to illustrate the advantages and disadvantages of either type of tools. In Section 2.4 we introduce *chaff* as enabler of *utility-preserving obfuscation* and examine the use of chaff in security and privacy research, identifying different types of chaff and their uses. Lastly, we conclude in Sect. 2.5 with a summary of the contributions of this chapter.

2.1 Obfuscation in security and privacy research

We examine properties, goals and methods that the security and privacy research communities have denominated or referred to as *obfuscation*. We note that we focus on prominent and dominant conceptualisations of obfuscation, i.e. we do not attempt to capture niche understandings and uses that have not been broadly adopted.

2.1.1 Obfuscation as complexity. The software engineering perspective.

The most prominent use of obfuscation in the software engineering community relates to the *program obfuscation* problem —also known as *software* or *code* obfuscation. Program obfuscation seeks to generate a modified version of a program's code that, while guaranteeing the same utility as the original program, deters anyone from reverse-engineering it, learning its purpose or internal structure. The goal is hence *"to transform the program into a semantically equivalent program which is much harder to understand for an attacker"* [24]. Program obfuscation may target either source or machine code and finds application in scenarios such as intellectual property and data protection or anti-tampering and vulnerability discovery prevention [43].

Software engineers have proposed a range of methods for software obfuscation. These include lexical, control, data and semantic transformations of code that aim to hinder the efforts of an adversary at understanding or reverse-engineering the obfuscated program [43, 137].

Various authors have however either criticised or acknowledged that these methods are not grounded on rigorous definitions to guarantee any notion of provable security properties [24, 137, 338, 401, 474]. Program obfuscation progresses as *"an arms race between software developers and code analysts"* [474]: developers design new program obfuscation methods to defend against the latest debugging, disassembling and emulation tools [371] or any other type of code

analysis techniques available [43], exploiting their limitations to produce code that is, at that particular moment, hard to analyse [371]. Code analysts in turn respond by refining their analyses to break new obfuscation methods, thus raising the bar for program obfuscators to design better obfuscation. This highlights the lack of provable guarantees: software engineers' obfuscation methods target existent reverse-engineering techniques guaranteeing nothing about future code analysis developments.

Collberg et al. evaluate code obfuscation techniques according to their *obscurity* (measured as the additional time an adversary requires to reverse-engineer or understand an obfuscated program), their *resilience* (the possibility to design an automated tool that undoes obfuscation), their *stealth* (the ability of an adversary to detect obfuscated code), and *cost* (the additional computation and memory resources required to run the program) [137]. Anckaert et al. propose a set of *software complexity metrics* such as instruction counts, cyclomatic numbers or knot counts in the program's *control flow graph* that instead of targeting a particular code analysis technique (or combination thereof) attempt to capture principles that apply to code analysis *complexity* in general [24]. The rationale behind this proposal is that any code analysis technique is slowed down or hindered by an increasing number of program instructions or control flow graph complexity; hence their utility as a proxy measure of obfuscation. Schrittwieser et al. alternatively rate the robustness of similar classes of obfuscation methods against classes of code analyses (such as pattern matching and static, dynamic and human-assisted analysis) by the ability of the latter to defeat the protection the former provide [474].

Common to all these measures is the notion that obfuscation depends upon code *complexity*. Program obfuscation methods seek to increase the complexity of the reverse-engineering process so as to, even if unable to guarantee any formal definition of security, significantly raise reverse-engineering costs, forcing in turn an adversary to spend as many resources on de-obfuscation as possible [371]. Moreover, while there is no universal or fixed set of measures that software engineers rely on to evaluate software obfuscation, there is the underlying notion of a continuous spectrum of effort/protection: the more code complexity the developer introduces, the more or better obfuscation obtains as a result.

2.1.2 Obfuscation as indistinguishability. The cryptography research perspective.

The cryptography community has long seen *program obfuscation* as a cryptographic *"master tool"* that would by design provide any cryptographic functionality, such as public-key cryptography, secure multiparty computation

or zero-knowledge proofs [44]. However, the software engineering *'arms race'* approach to program obfuscation has long received criticism from the cryptography community due to the lack of formal definitions and provable security guarantees in their methods and solutions [137, 338, 401].

In addressing this, Hada and Barak et al. provide the first formal definitions of program obfuscation [46, 265]. Barak et al. propose the "virtual black box" model, whereby an obfuscated program $\mathcal{P}'$ must provide the same functionality as the original, non-obfuscated program $\mathcal{P}$, while anything we can efficiently compute from $\mathcal{P}'$ we must be able to efficiently compute given oracle or black-box access to $\mathcal{P}$ too [46].

Barak et al. demonstrate that such a definition of obfuscation is in general impossible to satisfy due to the existence of a family of *inherently unobfuscatable* functions [46], the intuition being that an obfuscated program reveals code providing a given functionality (that may be input to other programs and reused at will) and that is inherently more informative than no code at all, i.e. a black-box.[2] To circumvent this impossibility result, Barak et al. propose a weaker notion of obfuscation, *indistinguishability obfuscation* (IO), whereby an adversary is unable to distinguish obfuscated programs $\mathcal{P}'$ and $\mathcal{Q}'$ of functionally equivalent programs $\mathcal{P}$ and $\mathcal{Q}$.

The "cryptographic notion" of obfuscation is therefore more akin to a binary property: an obfuscator Ω is either able to satisfy the definition (any of the above) or not. This differs from the software engineering perspective, where measures of complexity denote a range of obfuscation potency or resilience: the more code transformations, the more obfuscation.

Barak further equates the approaches to program obfuscation of software engineers and cryptographers to *security by obscurity* and *security by simplicity*, respectively [44]. The reason is that the former rely on adding new complex transformations that existent code analysis techniques are not aware of or prepared for. However, as soon as code analysts discover these transformations and refine their methods, they can bypass obfuscation. Modern cryptography on the other hand predicates its security on open designs and well-known hard problems such as integer factorisation, the discrete logarithm problem or one-way functions [45]. Instead of relying on new techniques adversaries are yet unaware of and need to adapt and find a way around, cryptographers rely on open and well-known cryptographic primitives for which no efficient attacks exist or are yet known.

[2] Matthew Green provides a layman's explanation of the different cryptographic definitions of program obfuscation in his blog *"A Few Thoughts on Cryptographic Engineering"* [247].

In spite of this shift from *security by obscurity* to *security by simplicity*, still the underlying notion to the approaches of both software developers and cryptographers to program obfuscation is that of *complexity*. Software developers' obfuscation methods rely on new, complex code transformations existent code analysis techniques do not account for. Cryptographers' current approaches to program obfuscation rely on hard, complex problems for which no efficient solutions are yet known [387]. Granted that the difference between the software engineering and cryptographic approaches is stark: cryptographers provide provable guarantees through reductions to well-known hard problems such as the discrete logarithm, whereas the software engineering approach does not: it simply hopes that obfuscation will be hard to undo, offering no formal guarantees over the proposed obfuscation mechanisms. Still, Barak indicates that there is *"no strong evidence"* that problems such as the discrete logarithm or integer factorisation are actually hard; hence, there is no reason to *"assume the nonexistence of a[n] algorithm for these problems"* [45]. The advent of quantum computing has cast further doubts over the security of current cryptosystems in the future, especially those relying on public-key cryptography [72, 118].

If it is just a matter of *time* before *crypto analysts* find a way —either through more efficient algorithms or new technology— to break the cryptosystems underlying cryptographers' obfuscation, then the difference in approaches between the cryptography and the software engineering communities seems to shrink. This connects with an alternative, loose understanding and use of the word *obfuscation*, namely, that of "bad" or *"faulty encryption"*.

A note on "obfuscation as *'bad encryption'*". The implicit vagueness of the term *obfuscation* has led security experts to liberally use it to denote a host of data transformations that, while not secure enough to be deemed *encryption* (i.e. in accordance to modern cryptographic standards in that no feasible, efficient attack is currently known), do impose nevertheless a certain amount of effort —even if minimal— required to undo them. We highlight however that there is no consensus with respect to this terminology (i.e. referring to insecure forms of encryption as obfuscation) nor a formalisation of this understanding of insecure encryption as obfuscation.

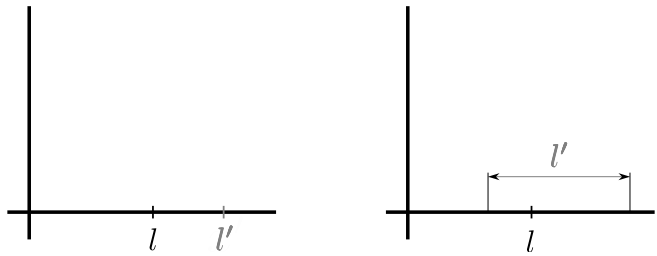
Researchers have denoted as obfuscation earlier (insecure) proposals of order-preserving encryption [150]; insecure, deterministic data masking (e.g. by repeatedly XOR-ing data with a constant string [367]); weak encryption [20], or simply insecure security practices involving encryption such as storing decryption keys in easily accessible files [316].

We note that this notion of obfuscation represents *security through obscurity* rather than the *security through simplicity* or *open security* principles underlying

modern cryptography and thus shares more similarities with the notion of obfuscation in software engineering.

2.1.3 Obfuscation as inaccuracy and imprecision. The privacy engineering perspective.

We mention at the beginning of this chapter that any privacy technology that conceptualises privacy as confidentiality or hiding may potentially be considered an ObPET. In practice however, the most prominent understanding of obfuscation in the privacy community is that of techniques that rely on a host of data operations that modify data to render it less accurate or precise, aiming to lessen in turn the amount of sensitive information an adversary acquires from them. Such data operations include, among others, randomisation, addition and suppression, generalisation, shuffling and swapping [11]. By **inaccuracy** we refer to deviations from data values akin to *errors* (random variability) whereas by **imprecision** we refer to coarser granularity values encompassing the original value [186]. Figure 2.1 provides a graphical representation of inaccuracy and imprecision.



(a) Accuracy: l' less accurate than l.

(b) Precision: interval l' less precise than l.

Figure 2.1: A graphical representation of (in)accuracy and (im)precision

An example that further illustrates these concepts in the context of location data is the following. Let us consider a location l, e.g. a set of GPS coordinates $(l_\phi, l_\lambda) = (50.862469\text{ N}, 4.686821\text{ E})$, pointing to a location on Campus Arenberg at the University of Leuven. An *inaccurate location* entails a different set of GPS coordinates l', pointing to another location entirely (even if arbitrarily close to the actual location l), e.g. $l' = (l'_\phi, l'_\lambda) = (50.869364\text{ N}, 4.692250\text{ E})$, a nearby location on campus, or $l' = (l'_\phi, l'_\lambda) = (50.88\text{ N}, 4.70\text{ E})$, a different location in the centre of Leuven. An *imprecise location* on the other hand entails granularity coarser than GPS coordinates (e.g. at the street, city

or country level) yet including the actual location $l = (l_\phi, l_\lambda)$, e.g. in order of decreasing granularity: $l' = \{\text{Campus Arenberg}\} \subset \{\text{Leuven-Heverlee}\} \subset \{\text{Vlaams-Brabant}\} \subset \{\text{Belgium}\}$.

Privacy engineers turn to *data obfuscation* to provide database anonymity [364, 512], search privacy [42, 235], location privacy [30, 186, 490], privacy preserving personalisations and recommendations [427, 429], communication profile confidentiality [41] or privacy-preserving data mining [14], among other privacy goals. We however note that the use of the word *obfuscation* to denote these data operations is not equally established within each of these subcommunities. Whereas location privacy researchers typically refer to the operations they perform on location data as *obfuscation*, database privacy researchers rarely do so, favouring alternative terms such as *perturbation* or more specific terminology such as *randomisation* or *generalisation*, even if the set of techniques both use is oftentimes the same or analogous. The database privacy community moreover distinguishes between operations that seek to trade-off data accuracy for privacy, offering no guarantees of data *'truthfulness'*, such as randomisation, and those that trade-off data precision for privacy, resulting in truthful yet imprecise data, such as generalisation [31].

In spite of the variety of data operations we may denote as obfuscation and the wide set of privacy problems researchers seek to address with them, they all share one common feature and underlying assumption, that of *gradual protection* and *monotonicity*. Obfuscation enables privacy engineers to tune the degree of privacy protection by choosing to add *more* or *less* obfuscation. More obfuscation entails greater data degradation. This, in turn —assuming a sound obfuscation strategy is in place— enables better privacy protection.

We further distinguish between two different approaches to data obfuscation. On the one hand, *utility-degrading* obfuscation (UDO) addresses trade-offs between utility and privacy by corroding data quality in ways that are detrimental to both adversaries and non-adversarial users. These methods apply obfuscation that makes data less useful both to adversaries —thereby inhibiting their ability to breach data subjects' privacy— and to non-adversarial users —who also see the quality of the service they receive diminish. When deploying these methods, the ultimate goal of the privacy engineer is to strike a balance between utility and privacy loss [351, 465].

On the other hand, *utility-preserving* obfuscation (UPO) seeks to prevent adversaries from invading data subjects' privacy while retaining all utility for non-adversarial users. When deploying these methods, the ultimate goal of the privacy engineer is to prevent adversaries from acquiring *any* privacy-sensitive information while preserving all utility for non-adversarial users. The ability to deploy either type of obfuscation depends on both the desired functionality

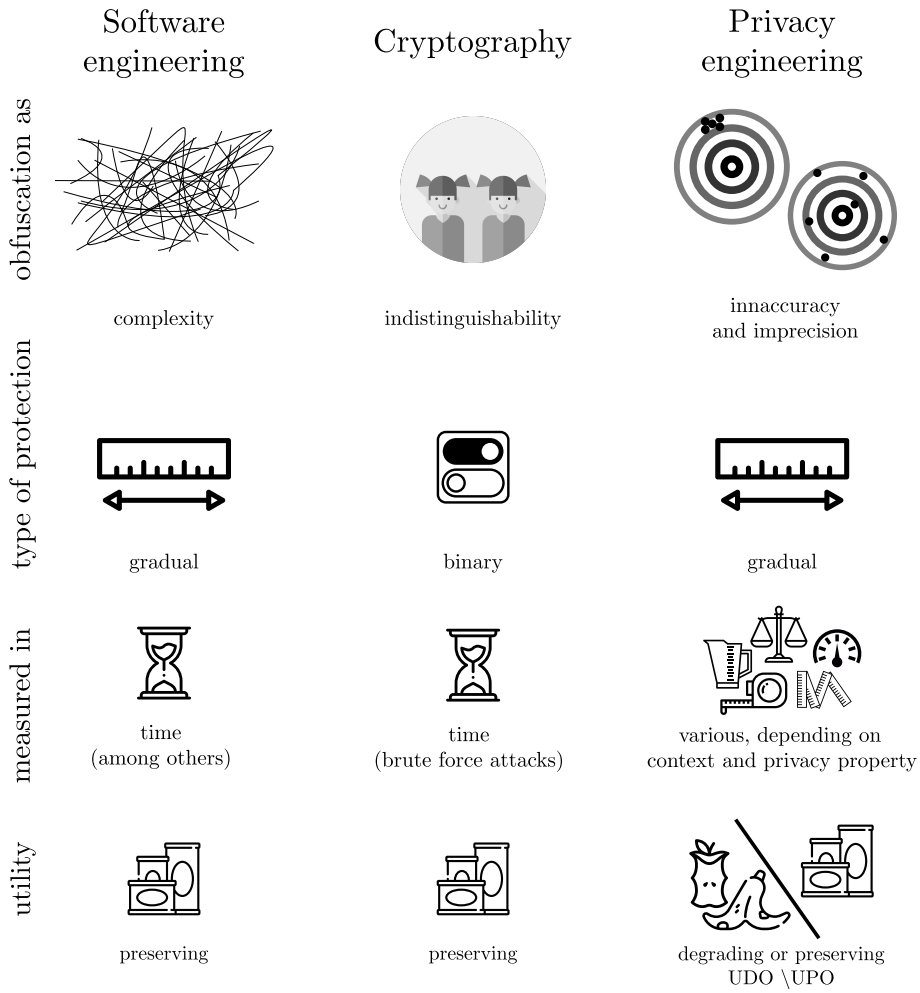


Figure 2.2: A comparison between notions of obfuscation across security and privacy research.

Image includes the following icons from `www.flaticon.com`: Twins, sand timer, meter, canned food and banana (author: Freepik), ruler (author: Good Ware), switch (author: Smashicons), pitcher (author: phatplus), balance, tape measure (author: monkik), speed meter (author: Yannick) and eaten apple (author: Those Icons).

and the system model in ways that a privacy engineer cannot always control, e.g. if the information system is already in place and running. We delve into the separation between utility-degrading and utility-preserving obfuscation in Sect. 2.2.1.

Lastly, we note the notion of program obfuscation in both the software engineering and cryptography communities represents *utility-preserving obfuscation*, as the goal is to produce code that minimises what an adversary learns while preserving the program's functionality. Moreover, the software engineering and cryptography communities measure obfuscation quality in *time* (that it may take to undo obfuscation and brute-force a cryptosystem, respectively) while the privacy community has a wide variety of metrics that depend on the particular privacy property we aim to provide with obfuscation.

Figure 2.2 summarises the notions and uses of obfuscation across security and privacy research as described in this section.

2.2 Data obfuscation

In this section we examine in more detail privacy tools that rely on obfuscation as data degradation. To that end, let us consider a bare-bones model of a privacy invasion. Let us consider an individual that uses an online service that offers a set of functionalities $\{f_i\}$, such as sending a message to a friend, browsing the Internet, posting a blog entry or searching the web. We refer to this individual as the *user* or *data subject* and we abstract from the particular type of service she uses; we simply consider that the user performs a series of service requests r_i obtaining as a result a set of responses $\{o_i\}$ so that $o_i = f_i(\mathrm{r}_i)$, with functionality f:

$$f : \mathcal{R} \to \mathcal{O} \tag{2.1}$$

where $\mathcal{R}$ represents the universe of user service requests r and $\mathcal{O}$ the universe of responses o the system provides. A user request r includes not only the designated service the user is interested in, but also input data that the execution of the service requires, e.g. the message a user sends or the URL of the website she wishes to visit. Hence, we more generally refer to r as the user input and to o as the system output. Moreover, we denote the *sequence* of user inputs as $\mathrm{r} = [r_1, r_2, \ldots, r_n]$ and the sequence of responses as $\mathrm{o} = [o_1, o_2, \ldots, o_n]$.

An adversarial entity is interested in the user input data r, but allowing the adversary to obtain r invades the user's privacy. Figure 2.3 depicts an information flow conceptualisation of this model.

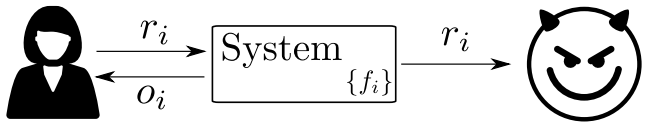


Figure 2.3: Process flow conceptualisation of a privacy invasion.

We account for the utility users derive and the privacy loss (or adversarial gain) they incur from revealing r to an adversary. We represent utility and adversarial gain through functions u and $\mathcal{A}$, respectively. The user obtains utility u_f from functionalities f_i so that

$$u_f : \mathbf{O} \rightarrow \mathbb{R}^+ \tag{2.2}$$

where $\mathbf{O} = \{\mathrm{o}_i\}$ represents the universe of output sequences.

The adversary on the other hand obtains information by processing users' input data with a set of functions g_i so that

$$g : \mathbf{R} \rightarrow \mathcal{X} \tag{2.3}$$

where $\mathbf{R}$ represents the universe of user input sequences r and $\mathcal{X}$ the universe of outputs $\mathbf{x}$ the adversary obtains from g, i.e. what the adversary wishes to learn about the user.

We model the privacy loss $\mathcal{A}$ that users incur by revealing r as:

$$\mathcal{A} : \mathcal{X} \rightarrow \mathbb{R} \tag{2.4}$$

Lastly, we consider *social utility*, namely, utility a user provides to others (including herself) by revealing r to adversarial parties. We consider that social utility derives from a set of functions $h_i(\mathrm{r})$ so that:

$$h = \mathbf{R} \rightarrow \mathcal{O}_h$$

where $\mathcal{O}_h$ represents the universe of outputs that produce social utility, e.g. film recommendations or traffic-aware driving directions based on the user's film

reviews and position on the road while driving, respectively. Hence, we obtain social utility as:

$$u_h : \mathcal{O}_h \to \mathbb{R}^+$$

Users may exchange and reveal data among a trusted group of relatives, friends or coworkers. We do not consider that exchanges among trusted peers contribute to social utility. Rather, we consider that the utility users derive from disclosing r to trusted peers is part of their personal utility, whereas social utility exclusively derives from data disclosure to potentially adversarial users. Whereas this decision may seem arbitrary,[3] it enables us to conjecture a crisp separation between the utility requirements that enable either UPO or UDO, as we discuss later in Sect. 2.2.2. Figure 2.4 represents the inclusion of utility and privacy leakage in the model. Table 2.1 summarises the notation we have introduced so far.

Symbol	Meaning	Symbol	Meaning
r	User input/request	r	Sequence of user inputs $[r_1, r_2, \ldots, r_n]$
$\mathcal{R}$	Universe of user inputs	$\mathbf{R}$	Universe of user input sequences
f	System functionality	u_f	Personal utility function
o	System output/response, $o = f(r)$	o	Sequence of system outputs $[o_1, o_2, \ldots, o_n]$
$\mathcal{O}$	Universe of system outputs	$\mathbf{O}$	Universe of system output sequences
g	Function of interest to the adversary	$\mathcal{A}$	Privacy loss / Adversarial gain
$\mathbf{x}$	Adversarial outcome, $\mathbf{x} = g(\mathrm{r})$	$\mathcal{X}$	Universe of adversarial outcomes
h	System functionality	u_h	Social utility function
Ω	Obfuscation function		

Table 2.1: Overview of data obfuscation model notation.

[3]Deciding which peers or entities a user trusts is in fact entirely at the user's or system designers' own discretion.

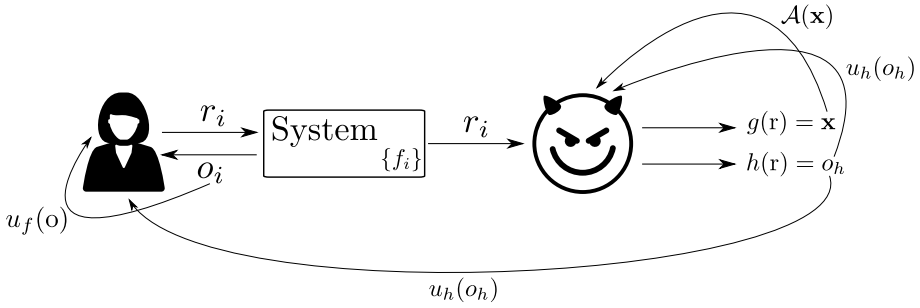


Figure 2.4: Privacy loss and utility as functions over data exposed to an adversary.

Data obfuscation as a form of protection modifies the input data r users provide to the system to minimise the privacy loss that derives from revealing that data to an adversary. Disclosing an obfuscated r′ instead of r seeks to minimise privacy loss, this is, turn $\mathcal{A}(g(\mathrm{r}')) = \mathcal{A}(\mathbf{x}) \to 0$, while keeping intact or maximising user utility, this is, $u_f(f(\mathrm{r}')) = u_f(\mathrm{o}') \to u_f(\mathrm{o})$ and social utility, i.e. $u_h(\mathrm{o}'_h) \to u_h(\mathrm{o}_h)$. Figure 2.5 depicts an information flow conceptualisation of privacy tools that rely on data obfuscation.

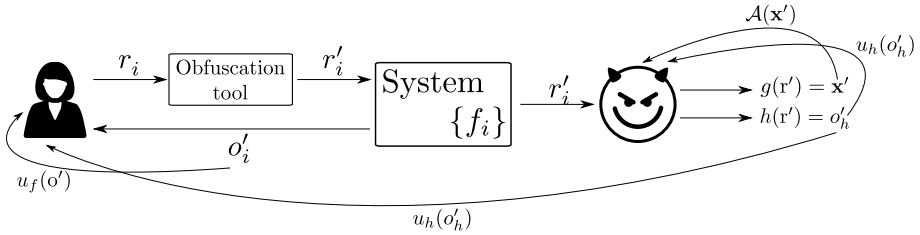


Figure 2.5: Process flow of data obfuscation tools

Data obfuscation is *syntax-preserving* [360].[4] Obfuscated data takes on values that the system has been designed to process, even if these values are less accurate or precise. A defining feature of data obfuscation is therefore that it requires no modifications of the system it is applied to. A user or system administrator may unilaterally deploy data obfuscation without the need for any system alterations.

[4]Note however that a system may only admit values on a predetermined precision scale, e.g. GPS coordinates. In such cases, the type of obfuscation is restricted to inaccurate data, rather than imprecise *"yet truthful"* data.

Moreover, obfuscation represents a *gradual* protection mechanism as we may degrade more or less the accuracy and precision of users' input data so that, ideally —assuming a sound obfuscation mechanism is in place— we obtain higher or lower privacy protection, respectively.

Hence, we may distinguish between tools whose underlying privacy protection mechanisms hinge upon data degradation (i.e. modifying r into r′) and those that do not, such as encryption and onion routing [243]. Encryption does not rely on introducing inaccuracy or imprecision, but on a complete reencoding of a data item (e.g. a message) that should bear no relation to the original content (other than in terms of size). The protection a *secure* encryption scheme[5] affords is therefore *"binary"*: an unauthorised party is either able to decrypt the message or not. Encrypted content may in fact be considered perfectly obfuscated (as in statistically indistinguishable from random noise), precisely because one does not gradually increase or decrease protection by adding more or less obfuscation. While longer keys increase the amount of time an adversary needs to break them, once broken the message can be recovered in its entirety. This contrasts with data obfuscation tools, featuring gradual levels of data degradation that hopefully translate into equivalent gains or losses of utility and privacy. On the other hand, anonymous communication systems such as Tor [176] attempt to hide the IP address of an Internet user (i.e. the r in our model) from an external observer or the end server by routing the message through a series of intermediary relays rather than making the IP less accurate or imprecise so that the more obfuscated the IP is, the less accurate or imprecise the IP the adversary observes.[6] Hence, we restrict our definition of obfuscation tools to those that derive privacy protection from data degradation.

2.2.1 Utility-preserving and utility-degrading obfuscation

We distinguish between two approaches to data obfuscation according to their impact on user utility: *utility-degrading* and *utility-preserving*.

Utility-degrading obfuscation.

Utility-degrading obfuscation alters r to mitigate users' privacy loss at the expense of utility loss. Utility-degrading obfuscation is unavoidable when no

[5] By *secure* encryption we mean a semantically secure cryptosystem [244] whose implementation is free from software bugs and resistant to side-channel attacks [264, 327].

[6] Referring to *accurate* or *imprecise* IPs requires the definition of a metric space over which we can evaluate the distance between IPs, e.g. distance over IP address numbers or distance between the actual geographical location of the machines to which each IP maps.

obfuscation mechanism can enable privacy gains without a drop in utility. Formally:

$$\nexists\, \Omega(\mathrm{r}) = \mathrm{r}' : \mathcal{A}(\mathbf{x}) > \mathcal{A}(\mathbf{x}') \wedge u(\mathrm{o}) = u(\mathrm{o}') \tag{2.5}$$

This further entails that g depends on a subset of r —or relationships therein— that f depends on, so that any changes to r_i that have an impact on $\mathbf{x}$ affect o_i too.

Whereas utility-degrading obfuscation may be a design choice for privacy engineers for reasons such as cost, convenience or personal preference, it is however unavoidable when utility derives from information disclosure to adversarial entities, making the trade-off between utility and privacy inescapable. We explore the conditions that may impose utility-degrading obfuscation in Sect. 2.2.2.

Utility-preserving obfuscation.

Utility-preserving obfuscation alters r to mitigate users' privacy loss while preserving utility, i.e. $\mathcal{A}(\mathbf{x}') < \mathcal{A}(\mathbf{x})$ while $u(\mathrm{o}') = u(\mathrm{o})$.

Utility-preserving obfuscation is only possible when users derive their utility from a set of functions $\{f_i\}$ that are not affected by changes in input data r that the set of adversarial functions $\{g_i\}$ is sensitive to. In lay terms, this means that the user and adversary goals are independent to the extent that it is possible to effect changes in the input limiting the impact of such changes to $\{g_i\}$ alone. Utility-preserving obfuscation is possible when there exists an obfuscation mechanism that enables privacy gains without a drop in utility. Formally:

$$\exists\, \Omega(\mathrm{r}) = \mathrm{r}' : \mathcal{A}(\mathbf{x}) > \mathcal{A}(\mathbf{x}') \wedge u(\mathrm{o}) = u(\mathrm{o}') \tag{2.6}$$

In practice, this means that f_i depend at most on a proper subset of the data (or relationships therein) that g_i depend on, so that it is possible to effect changes on r that have an impact on g but not on f. We explore the conditions that impose utility-degrading obfuscation in the next section.

A note on cost. In this thesis we separate between utility as the set of functionalities the user aims to achieve (i.e. $\{f_i\}$ and $\{h_i\}$) and the cost or expense at which these come. Cost has practical implications for the design of

privacy technologies in general and obfuscation tools in particular, as increased costs may trump utility and make a system unusable.

In defining utility as independent from cost, we render our definition of UPO oblivious to the latter, leading to the paradox where a UPO mechanism may protect privacy at prohibitive cost for users, thereby rendering the purported utility of the system meaningless. Hence, in practice, UPO design viability depends on additional considerations relating to deployment costs that we abstract from in this thesis.

2.2.2 The role of personal and social utility.

Figure 2.4 distinguishes between individual and social utility as characterised by functions f_i and h_i, respectively.

Personal utility refers to the utility an individual revealing r_i obtains exclusively for herself or for others she trusts. We refer to individuals that exclusively seek to obtain personal utility from a system as *utility consumers*. Consider an individual that uses a public transport app, such as the SNCB app offering information on public transportation in Belgium. This kind of app enables people to find the nearest public transport station, plan routes between two locations and obtain information about delayed trains, trams and buses. To obtain this set of functionalities a user needs to provide her location, desired origin and destination and chosen means of transport, respectively. If we consider that a user intends the input she feeds to the service provider to be useful to no one besides herself and those she trusts, then that user exclusively seeks *personal utility*, she is a *utility consumer*, e.g. a user provides her location to the SNCB app to find the nearest train station and she does not intend her location to be of utility to any adversarial party; similarly, a user that sends a private message to a friend intends that information to be of utility to no one besides herself and her friend, whom she trusts.

Social utility on the other hand refers to utility an individual provides to others (including herself) by revealing r_i to adversarial entities. We refer to individuals that provide social utility as *utility producers*. Consider an individual that uses a medical research app, such as EpiWatch [289] or mPower [390] for epilepsy and Parkinson's patients, respectively. These kind of tools not only let users track their own symptoms and improve their treatment but also share that information with medical researchers to help them better study and understand the condition under treatment. Whereas users of these apps obtain personal utility from the potential for recognition of symptoms and earlier diagnoses —enabling in turn better treatment—, they also contribute with their data to improve the treatment of everyone (including themselves), thereby producing

social utility. However, we consider other patients or medical researchers non-trusted and potentially adversarial, i.e. they may leak or abuse the information they have collected from utility producers.[7]

Either type of utility imposes opposite data exposure requirements. Utility consumers should not need to reveal or expose their data to adversarial parties; the rationale being that if only the data subject and other trusted parties benefit from her data, then no adversarial party should require access to the data. Data exposure requirements depend therefore on the *system architecture* rather than on the intrinsic nature of the functionality itself. To illustrate this, consider a media delivery service (MDS), such as YouTube. In the commonplace, privacy-unfriendly system architecture, the service provider requires users to disclose the list of videos they desire to watch. This represents a privacy problem for users concerned about what the MDS provider learns about them from their media consumption patterns. Moreover, let us assume MDS users seek to obtain *only* personal utility and do not wish to produce any social utility i.e. they simply wish to watch the requested media. In this case, it is the system architecture that forces users to disclose their data to meet their utility requirements, not the utility function itself. In fact, a private information retrieval (PIR) implementation of the MDS enables users to receive the media they desire to watch without the need for them to disclose the list of items to the service provider [258].

A trivial implementation that illustrates why personal utility does not require users' data disclosure relies on the service provider sending all data and code to run the service to each user. Equipped with everything they require to run the service, users do not need to disclose any usage information to the service provider. This is in fact how providers offered their services prior to the Internet, through *"shrink-wrap software"* that users bought and enjoyed in the privacy of their non-Internet connected homes [263]. Whereas such an implementation of a service like YouTube would prove infeasible, it highlights that some of the benefits of thin-client-based online services (for both users and providers) come at the expense of privacy loss.

Personal utility requirements alone do not, *theoretically*, impose any disclosure requirements on their users. Therefore, it *should* be possible to architect the system in a way that escapes trade-offs between utility and privacy, i.e. to enable

[7]Deeming a user's friend as trusted and medical researchers and other patients as adversarial simply reflects the trust assumptions implicit in the particular threat model we choose to consider in this example, i.e. one may argue that a user does not trust her friend, fearing he may publish their private photos and conversations online, while medical researchers' stringent security measures and high ethical standards fully warrant her trust. What is relevant here is that our data obfuscation model only considers privacy losses stemming from adversarial entities and implicitly assumes that trusted parties pose no threat to privacy, regardless of who we assume to be either trustworthy or adversarial.

personal utility without giving privacy away. Still, what is theoretically possible may not always be achievable in practice due to cost, available infrastructure or usability (e.g. unacceptable delays), among other constraints. Whereas it may be feasible to deploy a PIR implementation of an MDS offering a small number of media files (e.g. in the few thousands) such as Netflix [258], this may not scale well to services such as Google search, indexing *"hundreds of billions of webpages"* and with stringent latency requirements [29, 246, 515].

Conversely, utility producers *always* need to at least partially reveal or expose their data, either directly (i.e. the raw data r_i they input to the system) or indirectly (i.e. through the product of computations performed on their data $h(\mathrm{r}) = o_h$); the rationale being that whoever benefits from the utility producer's data must have access to at least some function over those data. Data exposure requirements are therefore *unavoidable* as they depend on the intrinsic nature of social utility itself. Changes to the system architecture may reduce the privacy risk of providing social utility but they cannot completely prevent it.

To illustrate this, consider an MDS that offers personalised recommendations to users on what to watch next, i.e. other media items viewers may find interesting. Recommendation systems typically rely on probabilistic modelling to determine relationships between media items that viewers often like, e.g. whether viewers that like film A also often like film B. To determine such relationships, recommendation systems require sample data to train a model, i.e. samples of what people request to the MDS. One way of building a recommendation system is to enable a central party (e.g. the service provider) to collect users' viewing histories. The central party collects all viewers' histories to obtain the recommendation model, then offers personalised suggestions to each individual viewer. This however poses a privacy problem for viewers concerned about sharing their viewing histories with the central party.

An alternative to this centralised system architecture is the following. Each user trains the model locally and does not disclose her viewing history to anyone. A viewer thus obtains recommendations based on her own history, but not from other viewers' histories. Hence, whereas this alternative protects users' privacy, it also destroys all social utility. Users do not enjoy predictions based on other people's viewing histories.

To remedy this, users may share their own local models with the central party instead of their viewing histories. The central party then averages the individual models into a global model that users download to update their local model and keep on refining. This is the idea behind *federated learning* as used by Google to train Gboard in a distributed fashion [376]. Still, from the perspective of the user concerned about a central party collecting all data, this does not entirely address the privacy problem, as the central party obtains individual models that

leak information about the training data [32, 489]. Federated learning thus adds a level of indirection and complexity that prevents less sophisticated adversaries from breaching the users' privacy; however, it still leaks information about users from the outcome of the function computed over their data, i.e. $h(\mathrm{r})$.

Users may as well get rid of the central party by running a secure multiparty computation (MPC) protocol. In this scenario users still compute their own individual model locally. Then, instead of sending it to the central party, they run a distributed communication protocol among themselves to average the individual models into a global model in a way that no party gets access to any individual user model [148]. Whereas MPC prevents the participants from obtaining the individual model of any individual, the resulting global model still presents biases from each individual model, thus leaking information about the input individual predictive models. Of particular concern are well-informed adversaries, such as the classic *"knows-all-records-but-one"* or *"has-arbitrary-knowledge"* adversaries considered in the *differential privacy* literature [322]: MPC cannot offer any privacy guarantees against $n-1$ colluding parties out of n participants. Let us consider an adversary that observes the output $h(\mathrm{r}_v, \{\mathrm{r}_2, \ldots, \mathrm{r}_n\}) = o_h$, where r_v represents the input of the target user and the remaining r_j the inputs of the $n-1$ colluding parties. Knowing h, the colluding parties' r_j and the function output o_h, the adversary can either determine the actual value of input r_v or the set of values that could not have possibly lead to o_h, therefore learning *something* about the actual user value r_v —the exception being that h is insensitive to the user's input r_v, in which case the user input is irrelevant, i.e. social utility does not depend on it.

In fact, from Dwork's result on the impossibility of absolute disclosure prevention, it follows that *any useful* data disclosure leads to unbounded privacy losses against adversaries with arbitrary background knowledge [187]. By requiring that users expose (the result of a function over) their data, the provision of social utility entails some privacy loss. Hence, unless a user's input data r to social utility function h_i is completely randomised, it is impossible to prevent disclosure; however, fully randomising r leads to the destruction of social utility, because if any random input r to h produces social utility, the input itself is irrelevant and the user is unnecessary for the social utility production process, i.e. the user does *not* produce utility, she is *not* a utility producer.

Hence, we conjecture that against adversaries with arbitrary background knowledge, it is not possible to deploy UPO to satisfy social utility requirements. Social utility requires the exposure of *useful* (i.e. non-fully randomised) user data, which in turn leads to the impossibility of absolute disclosure prevention, which means that it must be impossible to satisfy $\mathcal{A}(\mathbf{x}') < \mathcal{A}(\mathbf{x}) \wedge u_h(o'_h) = u_h(o'_h)$. Social utility requires that we trade utility off for privacy. This is why Dwork et al. propose to shift from absolute guarantees to relative guarantees of privacy

and use differentially-private mechanisms that rely on UDO to trade utility off for privacy [187, 189, 191].

The disclosure requirements that either personal or social utility impose therefore have implications for both system design and privacy engineering, raising questions regarding the adequacy of obfuscation. Since personal utility does not inherently require users' data disclosure, we may posit that we must only rely on UPO to address users' privacy concerns, i.e. personal utility should not require users to degrade r_i in a way that reduces utility, as personal utility alone does not impose the disclosure of r_i in the first place. The use of UDO to balance personal utility and privacy therefore points to two possible pitfalls: either a suboptimal design, this is, the design unnecessarily relies on UDO instead of UPO; or a constraint in the system design, i.e. the system architecture forces users to give away utility in exchange for privacy protection.

Moreover, the *gradual* protection that we may expect from obfuscation tools in that *"the more obfuscation the more privacy protection"* is at odds with the fact that personal utility alone does not fundamentally require any disclosure at all. Obfuscation subjects utility consumers to unnecessary privacy risks whenever protection is less-than-perfect, i.e. whenever $\mathcal{A}(\mathbf{x}) > 0$. This ultimately questions the legitimacy of relying on obfuscation tools to engineer utility consumers' privacy. However, as we show below in Sect. 2.2.3, obfuscation provides a mechanism of response to privacy-invasive system architectures.

Conversely, since social utility mandates users' data disclosure, UDO seems to be *the only solution to fully address* users' privacy concerns against adversaries with arbitrary background knowledge. Since social utility implicitly requires disclosing *some* data to adversaries, there is no workaround to the trade-off between utility and privacy.

We illustrate the interplay between UPO, UDO and personal and social utility in the next section.

2.2.3 Obfuscation-based privacy-preserving location-based services

To illustrate the interplay between utility-preserving obfuscation (UPO), utility-degrading obfuscation (UDO) and personal and social utility, we provide an example in the context of location-based services (LBSs).

We consider an entity that offers LBSs online, e.g. such as Google Maps[8]. We refer to this entity as the service provider. The provider offers, among

[8] See `https://www.google.com/maps`

other services, geolocation (i.e. displaying the current location of the user on a map), driving or walking directions in real time (from the users' location) and suggestions for nearby restaurants, shops or any other type of public or private establishment users may be interested in. Users send *queries* $\{q_i, (l_i, t_i)\}$ to request the service q_i of their choice together with their actual location (l_i) and time (t_i). Figure 2.6 provides a depiction of a generic LBS.

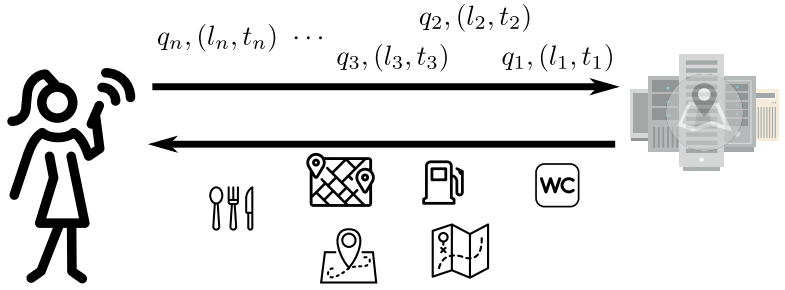


Figure 2.6: Interactions between user and service provider on a typical LBS. Image includes the following icons from `www.flaticon.com`: Woman on phone, cutlery, map, gasoline pump, WC sign (author: Freepik), map location, map (author: Smashicons), map (author: Vectors Market).

We further consider three use cases depending on the extent to which the provider leverages the users' input data on the services it offers:

No personalisation. Services depend on public information and individual user queries alone, e.g. the provider computes routes for each user independently from previous queries, be these queries from the same user or others.

Individual personalisation. Services depend on each individual user's previous queries, e.g. the provider may offer personalised driving directions based on the types of roads a user often chooses (from previously proposed itineraries) and the kind of instructions and map schematisations she finds easier to understand (e.g. based on previous erroneous turns and total amount of time needed to complete a journey).

Collective inputs and personalisation. Services depend on the usage patterns of all users, i.e. the provider relies on positive or negative feedback from users to improve the service for all. Moreover, the provider also leverages users' real time location information, e.g. to take traffic congestion into account (based on users' current position on the roads) when suggesting driving directions .

We assume that users consider the service provider to be *adversarial*, i.e. in spite of the benefits users find in personalisation and collective inputs, they are concerned the provider processes their location and usage information for other purposes, e.g. to offer them targeted advertising or lure them into spending more time using the service. Hence, users wish to minimise the amount of information the provider learns about them.

To that end, users turn to two types of location obfuscation solutions. On the one hand, solutions that rely on *dummy queries* [320, 321, 358, 565]. These tools automatically generate fake queries d_j with strategically chosen locations (l_j, t_j) and add them to the stream of *unmodified* real queries q_i with actual locations (l_i, t_i). These tools depend on achieving *indistinguishability* between real and dummy queries, compelling the provider to respond to dummy queries even if the tool filters such responses from users' view. We note that tools that generate dummy queries do not degrade real user queries. Instead, they aim to degrade the outcome of a function $g(\mathrm{r}')$ the adversary computes on those queries, e.g. a querying profile $\mathbf{x}$. Hence, under certain conditions we review below, these tools provide utility-preserving obfuscation (UPO). On the other hand, users may turn to solutions that rely on spatial and temporal degradation of user queries before they are sent out to the provider [27, 186, 232, 253]; i.e. sending queries with a *modified* location and time (l'_i, t'_i) that are less accurate, imprecise (or both) than the original $(l_i t_i)$. We note that since these tools degrade user queries' accuracy and precision, they often represent utility-degrading obfuscation (UDO) solutions.

Users that seek *no personalisation* or *individual personalisation* are strictly utility consumers, i.e. they share their data with the provider to benefit from the services on offer while expecting no one else to benefit from their data. Users have no incentives to share information related to their location or usage (e.g. type and frequency of queries they issue) other than to obtain the services they seek. As mentioned earlier, if users solely seek personal utility then there is no inescapable need for data disclosure. For example, in a trivial privacy-preserving implementation of the service that entails no user data disclosure, the service provider ships to users the code and maps they need to run the service themselves and reveal nothing. Less trivial implementations rely on cryptographic protocols such as private information retrieval [417], private equality testing [366] or homomorphic cryptography [544]. Cryptographic solutions however require modifications on the side of the service provider and, as we discuss in Sect. 2.3, this may not always be possible. Obfuscation tools on the other hand can be deployed without any changes on the side of the provider. Privacy engineers should therefore in this context aim to design obfuscation tools that satisfy $u(\mathrm{o}') = u(\mathrm{o})$ and $\mathcal{A}(\mathbf{x}') < \mathcal{A}(\mathbf{x})$ —ideally $\mathcal{A}(\mathbf{x}') = 0$.

Obfuscation tools that rely on dummy queries satisfy that requirement. Let us first consider the *no personalisation* use case. An obfuscation tool that sends dummy queries on behalf of the user without interfering with the user's real queries preserves utility by ensuring that the user gets the same responses she expects without dummy queries. Moreover, assuming the tool deploys a sound obfuscation strategy, the adversary is unable to tell which queries are real and therefore unable to determine the user's position or usage patterns. As a result, obfuscation tools that rely on dummy queries provide UPO; since $u(\mathrm{o}') = u(\mathrm{o})$ and, considering *perfect obfuscation*, $\mathcal{A}(\mathbf{x}') = 0$. In practice however perfect obfuscation is hardly ever possible, thus $\mathcal{A}(\mathbf{x}') < \mathcal{A}(\mathbf{x})$, still satisfying the UPO requirement.

However, this obfuscation tool does not satisfy the UPO requirement in the *individual personalisation* use case because dummy queries obfuscate the real usage patterns the service provider relies on to provide personalisation. Worse still, rather than losing personalisation and *defaulting* to the *no personalisation* utility baseline, they obtain spurious personalisation from dummy queries. It is still possible however to provide UPO with dummy traffic, only at a higher cost and level of complexity. Let us assume the user has the ability to create multiple accounts on the service, obtaining personalisation in each of those accounts individually, i.e. services get personalised for each account based on previous usage on that account alone. Let us further consider a tool that populates with dummy queries a number of these alternative user accounts confining all real user activity to one account in which no dummy traffic is added. This tool preserves personalisation on the user account without revealing to the adversary which account contains the real user behaviour; the user still obtains utility from personalisation in her account whereas an adversary needs to determine who among the multiple simulated selves is the real one. Still, users may not always be able to create multiple personal accounts, thereby imposing an architectural restriction that undermines the deployment of UPO.

Users that consider *collective inputs and personalisation* are both utility consumers *and* producers, i.e. they share their data with the provider to benefit from the services on offer and contribute to improve them for others. Users thus have incentives to share information related to their location or usage —even if some users may overlook such incentives and become *free-riders*, an issue Brunton and Nissenbaum discuss in their political and ethical study of obfuscation [100]. Since *social utility* requires them to contribute with their data to other people's services, they need to trade off social utility for privacy. Cryptographic tools such as multiparty computation (MPC) enable users to provide social utility without the need of a third party such as a centralised service provider that collects inputs from every user. However, they still require *statistical disclosure control* (SDC) tools to limit the amount of information

recipients of social utility may learn about utility producers from the results of the MPC functions.

Social utility and privacy are at odds because social utility requires users to disclose information to potentially adversarial users. Even the obfuscation tool we propose to preserve individual personalisation degrades social utility (for all users, including the one with the obfuscation tool), as the provider may either discard the input of the obfuscating individual or include one or several dummy accounts as input to the collective personalisation algorithm. Similarly, the provider cannot determine where obfuscating users really are. Services that depend on collective inputs such as real-time traffic congestion information thus suffer from spurious input data. If the provider removes obfuscating users from the dataset, it underestimates traffic congestion, whereas including all or a set of dummy accounts conversely overestimates it. We note however that quality of service (QoS) degradation may be minimal, depending on factors such as the percentage of obfuscating users or how uniformly distributed across the dataset the impact of obfuscation is, e.g. in the particular example of traffic congestion reports, whether all obfuscating users drive in a particular area or distribute themselves uniformly across the area under observation.

2.3 Why data obfuscation?

In this section we examine *technical* requirements and constraints that motivate the use of obfuscation tools over other privacy technologies, such as cryptographic and anonymity tools.[9]

Data obfuscation design features.

First and foremost, obfuscation is *syntax-preserving* [360]. It modifies the data values in a system without effecting any changes the system is unable to process, enabling privacy engineers to apply data obfuscation in a running system without requiring or imposing changes to a system's design.

Secondly and as a counterpoint to being syntax-preserving, obfuscation *degrades data integrity*. While obfuscation does not require or impose system design changes, it alters the data systems process, thus potentially altering their operations and outcomes.

[9]We examine the technical constraints that motivate the use of obfuscation from a computer science perspective. For a more philosophical account of the reasons that animate and justify the use of obfuscation, we refer the reader to Brunton and Nissenbaum's study of obfuscation as a tool for user privacy and protest [100].

Lastly, and in part as a result of it being syntax-preserving, data obfuscation can be deployed *unilaterally*. Data obfuscation modifies users' individual data inputs to a system, hence enabling users to deploy obfuscation tools on their own, without the assistance of the service provider.

These features provide a number of advantages to rely on obfuscation as an alternative to or in combination with other privacy technologies.

Cryptographic solutions.

The model of data obfuscation introduced in Sect. 2.2 describes systems where users obtain utility from revealing data to an adversarial entity and how obfuscation tools modify those data to reduce the information adversaries acquire while attempting to preserve as much as users' utility as possible.

Cryptographic tools recast this model by *hiding* intervening user data from the adversary while guaranteeing users' utility. These tools rely on computations over encrypted data that adversaries are unable to decrypt and thus have no access to. This highlights in turn why obfuscation tools cannot always be replaced with cryptography. Since the very purpose of cryptography is to hide data from adversaries, cryptographic solutions cannot entirely address the disclosure requirements implicit in the provision of social utility, even if they can be engineered to reveal no more than what is strictly necessary [560], e.g. as long as the function a set of individuals compute through an MPC protocol is deterministic, it is impossible to establish any type of privacy guarantees against adversaries with arbitrary background knowledge, as per Dwork et al.'s result on the impossibility of absolute disclosure prevention [190]. At the same time, obfuscation cannot replace cryptographic solutions where data accuracy and precision are critical, such as in user authentication [129, 464].

When it comes to the provision of personal utility as defined in Sect. 2.2 however, cryptography represents the ideal solution: it preserves all utility for users while hiding their data from adversaries —even and especially if the latter are service providers. Private information retrieval (PIR) is a prime example of such a cryptographic solution. PIR enables users to obtain information from a database without disclosing to the database holder which information they retrieve [231, 419, 564]. Users thus obtain the same utility as from a non-privacy preserving database, namely, they retrieve the documents they are after, while the database holder learns nothing about the documents they retrieve.

Other examples include homomorphic encryption [234, 527] and multiparty computation [85, 141, 148, 242]. While currently inefficient for most practical applications, homomorphic encryption enables computation over encrypted

data, hence further allowing a user to outsource private data processing to untrusted, adversarial parties [328], e.g. users may send data to *"the cloud"* and use its computing power to operate on those data without the cloud provider being able to decrypt (and therefore pry into) the data it hosts and enables computations over. Secure multiparty computation (MPC) on the other hand enables a set of mutually distrusting parties to compute a function over their private inputs, yet these parties implicitly trust the output's recipients with the output's value —or implicitly assume that it is infeasible to recover individual inputs from the output alone, disregarding the threat of auxiliary information. Applications of MPC include, among others, secure distributed voting [54, 283], and private auctions [86, 85].

What these cryptographic solutions have in common is that they *require changes* to the system model, and thus the *involvement of multiple parties that agree to jointly run a cryptographic protocol.* Individuals cannot deploy these cryptographic solutions unilaterally. Whereas it is possible for an individual to send her data encrypted to the cloud without the involvement of the service provider or any other party, she still requires the service provider to enable homomorphic computations over her data. Similarly, a user cannot unilaterally deploy PIR; it is the service provider who must deploy the infrastructure and interface for users to query their services privately.

Hence, cryptographic solutions may not always be available. Other factors that further discourage the deployment of cryptographic solutions include companies' vested interest in data collection, both users' and service providers' unawareness of privacy problems and crypto solutions, as well as the additional cost of deployment. We review these factors below.

Vested interest in data collection. Service providers that obtain a benefit from data collection have no incentives in deploying cryptographic solutions that effectively starve them of users' data. Today's dominant online business model whereby companies obtain their revenue through the monetisation of user data explains the lack of incentives for service providers to offer cryptographic privacy protection online. Narayanan argues that *"[crypto-for-privacy's] goal can be thought of as roughly to prohibit secondary use of data"* and that *"misaligned economic incentives"* explain why *"secondary use is in fact a business imperative"* [399, 400].

Unawareness. Even if providers do not have a vested interest in collecting user data, they may be unaware of security and privacy issues in the service they provide, as well as the availability of cryptographic solutions that address those issues [400]. Furthermore, service providers often see privacy problems as requiring privacy policies, terms of use and other legal instruments that, even if necessary, do not essentially make the

service more privacy-friendly [400]. Moreover, a lack of institutional regulation requiring strong, technical privacy protection instead of data protection policies does little to encourage the adoption of cryptographic solutions [472, 329].[10]

If service providers have little pressure from regulators to deploy cryptographic privacy protections, users' or market pressure is weaker still. Narayanan argues that for people to rely on cryptographic tools, they need to be aware not only of their utility and existence, but also the underlying problems these tools address [400]. Besides, even if *some* users may be aware of the underlying privacy problems of relying on a particular online service, only a critical mass of users can effect enough pressure to encourage the service provider to deploy these solutions.

Users' unawareness and lack of technical understanding of cryptography further discourages service providers to adopt cryptographic solutions, as users may see no additional value in a secure system. Abu-Salma et al. have shown in the context of secure messaging that people do not understand what *end-to-end encryption* (E2EE) implies, that they assess reliability and security from the quality of service they experience and regard secure services as futile [2], further supporting Narayanan's argument against the classic trust model that assumes users can control and trust their devices [400].

Usability concerns have traditionally been put forward as an explanation for the dearth of *crypto-for-privacy* online, with infamously hard-to-use solutions such as PGP as a prominent example of crypto's lack of usability [551]. However, recent user-friendly cryptographic solutions such as OTR [19, 506] and most importantly the deployment of E2EE in popular instant messaging (IM) apps such as Facebook's Messenger, *Telegram* and *Whatsapp* [202] call into question long-held assumptions on the usability of cryptography, which Abu-Salma et al. dismiss as a relevant obstacle to adoption [2]. Still, research has shown users' inability to assess E2EE's reliability and security in those apps, thus questioning how usable E2EE in IM really is despite world-wide adoption [281, 475].

Cost. Service providers willing to deploy cryptographic solutions to protect their users' privacy face higher costs and challenges brought about by the deployment of cryptography. Deploying cryptography requires human expertise, dedicated hardware and software, and imposes an additional

[10] In the European Union (EU), the general data protection regulation (GDPR), requiring that companies implement privacy by design (PbD) [204], represents an exception to globally weak regulation over the collection and processing of personal data online. Legal experts expect that the EU, as the biggest single market in the world, will exert its influence setting a *gold standard* on privacy regulation around the world [18, 459]. The extent to which the GDPR encourages the adoption of strong privacy solutions in practice remains to be determined.

computational and communication burden, all of which contribute to higher costs in running a service.

Expertise in cryptography is a critical issue since few software developers have the necessary training and skills that cryptography demands. According to Belovin, *"the major issue [of deploying crypto online] has been one of cryptographic engineering: turning academic papers into a secure, implementable specification. But there is missing science as well, especially when it comes to efficient implementation techniques"* [62]. Narayanan further supports Belovin's argument by claiming that *"the idea that a developer who isn't a crypto expert could read a modern paper and understand and implement the protocol in a bug-free way is laughably unrealistic"* [400]. In-house developers thus seldom have the necessary skills to, first, identify privacy problems and, secondly, to deploy the set of cryptographic solutions that address them. Training personnel or hiring highly demanded crypto experts is also expensive, thus increasing operating costs of running the service. Narayanan points to *"misaligned incentives"*; cryptography for privacy is hard to sell and expensive to implement [400].

Moreover, cryptography also imposes an additional communication and computation burden. Whereas many efficient cryptographic solutions are available [19, 71, 194, 575], others still require advances and optimisations before they can be efficiently deployed at scale, like in the case of fully homomorphic encryption [491], private set intersection [325] and functional encryption [230]. Companies may not always have the computational resources available or be willing to invest in additional resources to execute fast enough the more expensive cryptography protocols. Furthermore, small, low-power devices such as implantable medical devices and "smart" user devices are resource-constrained and potentially unable to handle complex and expensive cryptography. Indeed, a whole field of research is devoted to the development of *lightweight* cryptography solutions [84]. Efficiency limitations may thus further hinder the adoption of cryptography.

In short, obfuscation arises as an alternative to cryptographic solutions where the latter are not or cannot be deployed. Still, we note that cryptography and obfuscation are not mutually exclusive. We illustrate in Chapter 5 how cryptography can assist achieving *indistinguishability*, a key obfuscation engineering requirement.

Anonymous communication.

Anonymous communication systems such as Tor and I2P enable Internet users to communicate or request services online without revealing their identity (sender anonymity) or who they communicate with (recipient anonymity) [176, 434, 570]. Where obfuscation modifies the input data r_i users reveal to adversaries, anonymous communication breaks the link between the source of the data (i.e. the user) and the data itself. Hence, adversaries either collect data on someone whose identity cannot be determined (within a sender anonymity set [434]), or on someone whose identity is known but whose communication partners and requested services cannot be determined (within the recipient anonymity set).

Anonymity thus tackles privacy concerns by breaking the link between communicating entities and their identities. Users may request and receive services *anonymously*, so that service providers and network eavesdroppers cannot determine what these users do. One may therefore argue that anonymity systems provide, within certain limits and assumptions,[11] a solution for users to conceal their identity or activities online.

Anonymity is however not always desirable or possible. For online services which require persistent interactions or user authentication, such as social media platforms, anonymity degrades to pseudonymity at best [445]. Users may log in to a web service through an anonymous communications channel to (a) hide their location from the service provider and (b) hide the services they log in to from entities monitoring the users' local network. However, even if users hide their IP from the service they log in to and authenticate themselves using a pseudonym, their activity in the system may be sufficient to re-identify them. Rao and Rohatgi point to two types of information that may enable identification of pseudonymous users: *syntactic* and *semantic* [445]. Syntactic information relates to features of text users write such as *"vocabulary, sizes of sentences and paragraphs, use of punctuation, frequency of blank lines, common misspellings, etc"*. Semantic information relates to the themes and concepts users express in their communications. Recent advances in *adversarial stylometry* have indeed shown that people have unique writing styles that enable identification the way fingerprints do [94, 374]. Other non-linguistic sources of information such as location data, social interactions or metadata further compromise pseudonymity [164, 245, 402].

[11] Tor for instance offers no resistance to global passive adversaries or targeted traffic confirmation attacks [176]. High latency anonymity systems such as Mixmaster and Mixminion [154] on the other hand aim to resist these attacks, but at the cost of high delays that make them unsuitable for real-time online activities such as web browsing [394].

Yet even if users remain pseudonymous there are privacy concerns that go beyond identity. Both commercial and political marketers are often uninterested in what most would consider users' identifying information such as name or ID number. Instead, they hope to obtain the necessary information to assign people to a relevant market or population segment that determines, for commercial marketers, which products (if any) they should target to them and, for political microtargeting, which messages they should expose a particular population segment to in order to rally their support. In this sense, pseudonymity is of little help to prevent targeted advertising and political microtargeting and the privacy problems that come with them, such as influencing future purchases through various marketing manipulations, *typecasting* and lack of oversight over unethical campaigning practices [537, 556].

Obfuscation responds to these threats by polluting user data so that they lose their potential for identification and do not bear the profiling and predictive power that adversaries expect to acquire. Both anonymous communications and obfuscation thus may be leveraged to tackle complementary aspects of privacy protection.

Obfuscation as additional protection. Obfuscation offers *an additional layer of protection* to both cryptographic tools and anonymous communication networks. Cryptosystems may include *backdoors*, fall prey to faulty implementations or use proprietary systems and implementations that impede independent reviewing, undermining the trust users place in these systems. If the cryptosystems users rely on are compromised, obfuscation helps mitigating the effects of the breach, offering, in cybersecurity parlance, increased *resilience* against security breaches [269]. Similarly and in addition to the reasons that motivate the use of both anonymity systems and obfuscation, obfuscation tools offer, in certain contexts, additional protection when anonymous communications fail, e.g. search engine users who fell prey to Cargenie Mellon University's infamous large-scale identification attack on Tor could have benefited from an extra layer of protection by using a tool that, by sending *fake search queries*, obfuscated their web search activity [177] (q.v. Chapter 4).

2.4 Utility-preserving obfuscation and *chaff*

Utility-preserving obfuscation (UPO) requires that users' online activities are either left intact or replaced by activities that produce identical or equivalent user utility. Of these two alternatives, we have already hinted at a solution that falls within the former, namely, in the location-based service (LBS) scenario we

have examined in Sect. 2.2.3, the generation of fake online activity (i.e. fake queries) pollutes the data adversaries obtain on users while ensuring users still receive the same responses to their actual queries (assuming no personalisation, that is). The second alternative involves replacing users' activities with modified versions that produce identical or equivalent results while minimising privacy losses, a process commonly referred to in the literature as *sanitisation*.

We recall that the data obfuscation model we introduce in Sect. 2.2 proposes two sets of functions: those that provide personal utility, $\{f_i\}$, and those that lead to privacy loss, $\{g_i\}$. The replacement or sanitisation strategy entails the removal or destruction of sensitive or private features from the data a user provides to an adversary in exchange for utility. In other words, to modify the user activity by degrading the data features that $\{g_i\}$ requires while retaining the necessary features that $\{f_i\}$ requires to preserve utility, e.g. by adding noise *"in directions along which private features are concentrated"*, to achieve *"full privacy without sacrificing utility"*, namely, UPO [296]. However, it is unclear whether sanitisation alone can achieve any rigorous guarantee of privacy and fully preserve utility at the same time [296].

Moreover, the design of replacement strategies requires sufficient knowledge of the mapping between users' actions and the corresponding outputs, this is, of the set of functions $\{f_i\}$ that provide utility, as designers must ensure that any noise addition strategy does not degrade $\{f_i\}$ outcomes. However, such knowledge may not always be available, e.g. in a search engine with millions of input word combinations and millions of retrievable webpages, retrieving the full mapping inputs-outputs requires exhaustive search over all possible combinations. Similarly, service providers may be unwilling to disclose the set of functions $\{f_i\}$ to retain intellectual property rights over them [524].

Moreover, having access to the set of functions $\{f_i\}$ questions the legitimacy of information disclosure in the first place, as one could theoretically have users computing $\{f_i\}$ locally and privately, on their own devices, without disclosing the input data r_i to an adversary. Xu et al. justify the use of sanitisation mechanisms as opposed to having users computing $\{f_i\}$ themselves in situations where, e.g. *"the software* [that computes $\{f_i\}$] *may be too big and require special hardware"* [559].

Conversely, obfuscation strategies that rely on the generation of additional, separate, fake user activity while leaving user actions intact need only to rely on the assumption that *each output o_i exclusively depends on its corresponding input r_i*; in other words, the output o_i to any input r_i depends on that particular input alone. Under this assumption, we do not require previous knowledge on the particular set of utility-bearing functions $\{f_i\}$ to ensure UPO, as fake activity has no impact over the utility users obtain from their own activity.

At the same time, under the assumption of *indistinguishability* between real and fake activity —the critical requirement for any such obfuscation strategy to work—, the adversary's gain function depends on a *polluted* set of input data r' that the adversary should be unable to de-obfuscate, in turn (hopefully) degrading the adversary's gain. Hence, this type of obfuscation strategy, which we refer to as *chaff*, represents a UPO mechanism.

We devote the remainder of this thesis to the study of *chaff-based obfuscation*. We note that privacy engineers may also choose to deploy chaff for UDO. However, in this thesis we do not examine the trade-offs between utility and privacy that arise in chaff-based UDO.

Chaff.

Chaff designates *fake*, *dummy* actions automatically generated on behalf of the user bearing no relation to the utility users expect to obtain from a service. While *useless* with respect to users' functional requirements [259], chaff has the potential to obfuscate usage patterns when intertwined with users' real activities —provided that adversarial observers are unable to *"separate the wheat from the chaff"*. Chaff may pollute adversaries' observations to the point of rendering data collection and processing futile, as adversaries cannot determine which fraction of the observed activities users generate and which fraction obfuscation tools are responsible for.

Historically, the origins of *chaff* as countermeasure (and etymology of its security-related meaning) date back to the chaff strategy and submarine decoys independently developed by both Allies and Axis powers during World War II to overwhelm enemy radar and sonar systems with false signals [391, 452]. WWII chaff consisted of small aluminium strips that planes seeking to escape the enemy's radars would drop to jam the signal radars expected to detect to locate targets. Hence, upon successful chaff release, radars would obtain a noisy signal and be unable to precisely locate the position of the enemy plane.[12] WWII teems with examples of military dummies and *decoys* deployment as a way to hide information from the enemy, deceive the enemy into attacking or prevent it from attacking, with some of these military strategies carrying up to today [456].

[12] Rieback et al. note that *"The German countryside became littered with chaff, which people used to decorate their Christmas trees"* [452]. In an ironic twist of history, after the war, Alcoa, the company that supplied chaff to the US military, went on to repurpose chaff in the manufacture of aluminium Christmas trees —in spite of the risk of electrocution during decoration. Indeed, many of the companies supplying the war effort resorted to Christmas items to repurpose their output [284].

Both terms *dummy* and *decoy* denote fake items or actions strategically designed to resemble their real counterparts, hence both terms are interchangeable in many contexts. The computer security literature however often favours either of these terms for particular uses. Dummies denote fake actions or entities whose purpose is to either hide or disguise information. Decoys represent traps and lures, particular types of dummies intended to attract, misguide, deceive and learn information about adversaries as part of a strategy to protect information. In this thesis we focus on the study of dummies as a means to hide information through obfuscation, rather than through disguise or deception. We shed further light on these terminological nuances through an overview of past work on dummies in computer security and privacy.

2.4.1 Digital chaff: dummy traffic and other dummies

Computer security researchers pioneered the use of chaff for privacy protection in communication systems by proposing the use of *dummy traffic* as a traffic analysis countermeasure. In a 1964 paper, Baran proposes to use *"a 'dummy' or filler stream of bits [to conceal] traffic loading"* in the context of security and secrecy of distributed communications [47]. Later in 1977, Kent proposes the use of dummy traffic to defend against traffic analysis in *"terminal-host communication"* [318], i.e. to hide *"the frequency, length and origin-destination patterns of message traffic"* in otherwise link-to-link or end-to-end encrypted communications. Baran and Kent point to privacy concerns stemming not from communication-over-insecure-channels content disclosure —as content may be encrypted and thus effectively hidden from unauthorised access— but from *communication patterns*, this is, the fact that who, when and how often people communicate leaks information about their communication. The introduction of dummy traffic seeks to prevent this leakage by burying real communication patterns in a fog of fake communication, ultimately providing communication *undetectability* [434], namely, an adversary monitoring network traffic cannot (ideally) determine whether observed traffic patterns are real or fake and, as a result, cannot determine who, when and how often people communicate.[13]

Relying on dummy traffic to defend against traffic analysis in digital networks indeed becomes a popular idea through the 80s and early 90s [116, 405, 536]. Of special relevance to our work is Chaum's 1981 seminal paper on *untraceable electronic mail* [116], marking the beginning of the research field of anonymous communications and more generally privacy technologies research [152, 156]. Chaum proposes the use of *mixes*, this is, computers in a communication network

[13] Achieving this level of privacy, along the lines of Shannon's perfect secrecy [483], requires however full network padding, i.e. adopting predefined dummy patterns independent from real ones in every transmission link in the network [218].

that relay messages hiding the correspondence between inputs and outputs by changing their appearance and order (i.e. re-encrypting messages and choosing a random output order, respectively) to hide the link between communication source and destination. Chaum however notes that mixes do not hide the number of messages a user sends or receives, hence he proposes, as a remedy, that senders generate randomly addressed dummy messages that recipients discard upon reception. Neither can mixing alone protect against powerful, global adversaries in low-latency anonymous communication systems [117, 159, 349], motivating the introduction of dummy traffic to prevent traffic analysis attacks [75, 171]; the rationale is analogous to the motivation behind Baran and Kent's proposals, i.e. to prevent an adversary from determining whether a particular instance of traffic is real or not, thereby introducing further uncertainty about the correspondence between a mix's inputs and outputs.

Anonymous communication networks may also benefit from the generation of dummy traffic at the end hosts to address the threat of website fingerprinting, whereby a local adversary monitors the encrypted connection of a user to the anonymous communication network and attempts to determine the website the user visits based on features such as total connection time, direction bandwidth and data bursts [192]. The addition of dummy traffic pollutes websites' identifying traffic features, mitigating the risk of website fingerprinting [105, 192, 309, 424, 546].

The use of dummies as a mechanism to hide patterns goes beyond anonymity systems. In wireless sensor networks [78, 254], the addition of both *dummy nodes* and dummy traffic helps preventing adversaries from locating critical nodes in the network, forcing adversaries not only to discriminate real from dummy communication but also real from dummy nodes. Zhou et al. use dummy data blocks and dummy read and write accesses in an encrypted file system in an attempt to conceal the location of user data therein [525, 576], i.e. an adversary should not ideally distinguish which blocks in the file system host random bits and which ones host user data. Dummy traffic has also found its way into early proposals of *private database joins* as a mechanism to leverage yet protect against untrusted servers [352]. In this scenario, an untrusted server hosts two databases with encrypted records to be combined if one or more of their attributes match, e.g. to combine those records both databases hold on a matching set of individuals. A trusted but resource-constrained secure computation component (SCC) reads two records (one from each database) at a time, determines if they must be joined and outputs the result back to the untrusted server. To hide from the untrusted server the result of the join operation, Li et al. propose to *pad* with dummy CPU cycles the time the SCC takes in determining whether or not there is a match, as matches take longer computing time. Moreover, to conceal the existence of a match from read

and write accesses (as in a *non-private join* only matches would be written on the server) the SCC is to output both matching and non matching records, adding dummy records for every non-matching record. Since all the records the SCC outputs are encrypted, only those parties with legitimate access to the outcome of the join operation are able to decrypt them and separate wheat (the matching, joined records) from the chaff (the dummy ones) [352]. CPU cycles and memory accesses as sources of information leakage are reminiscent of classic *side-channel* vulnerabilities in cryptographic hardware such as timing and power consumption analysis [308, 327]. Indeed, countermeasures to these side-channel attacks include flattening the power consumption signal through fixed time implementations or the addition of random delays and dummy operations [23, 327]. Whereas we acknowledge the analogies in the use of dummy operations and padding in both hardware security and privacy technologies research, we do not study such analogies in this thesis.

The scenarios above show how security researchers have leveraged the use of dummies —whether as traffic, sensors, CPU cycles or memory accesses— to hide patterns. However, not only are dummies useful to hide patterns and metadata but also represent an alternative to provide content confidentiality whenever the use of encryption is neither permitted nor available.

A prominent example of using dummies to provide *content confidentiality* dates back to the *"Crypto Wars"* of the 1990s [217]. In the midst of a policy debate over whether law enforcement should have surreptitious access to the content of encrypted communications (i.e. a backdoor to the decryption key), Rivest proposed *chaffing and winnowing* (C&W), a scheme that provides communications confidentiality *without encryption*, i.e. relying on authentication alone [453].[14] Rivest argued that since C&W provides confidentiality through authentication messages alone it should bypass any legal restrictions on encryption.[15] Chaffing and winnowing works by appending a message authentication code (MAC) to every block of (plaintext) data two communicating parties exchange and injecting in-between blocks of dummy data with random, fake MACs. Communicating parties agree on a secret authentication key that allows them to discard dummy messages upon reception, as the MACs of dummy blocks do not match the (plaintext) data. An adversary however, having no access to the authentication key, cannot distinguish valid

[14] A decade later, UK's Regulation of Investigatory Powers Act 2000 (RIPA) rekindled academics' interest in this scheme [132].

[15] In their security analysis of Rivest's scheme, Bellare and Boldyreva argue that chaffing and winnowing *is* encryption as it provides security properties equivalent to those of symmetric encryption; highlighting in turn divergences of what cryptographers and policy makers understand as encryption, i.e. the former defining encryption in terms of outcomes and security properties, while the latter vaguely referring to the mechanisms underpinning traditional encryption mechanisms rather than to what they achieve [60].

from invalid MACs, as a secure MAC should in fact be indistinguishable from a random tag. Still, since C&W sends data blocks in plaintext, adversaries exploit the content of messages to distinguish real from dummy ones, highlighting the challenge of generating dummy messages that pass as real ones. C&W overcomes this problem by splitting the data in single bits and sending as dummies each bit's complementary, foiling any attacks that rely on content and providing in turn *perfect indistinguishability* —and therefore perfect secrecy. We note however that C&W does not by itself conceal the who, when and how often anonymous communication systems aim to hide, i.e. C&W hides content, not communication metadata.

Similarly, Herley and Florêncio study the problem of password protection from keyloggers in untrusted end-user devices [277]. Keyloggers are pieces of malware that capture all user keystrokes in a device, thereby being able to obtain passwords and any other data users type on their keyboards. Herley and Florêncio note that keyloggers capture keystrokes at a low OS call level; keyloggers are able to capture everything the user types but lack context on where and why the user has typed it beyond the active OS window, i.e. keyloggers can tell whether the user has typed text in her email client or her browser, but not where within those programs or the specific purpose of the input. Conversely, browsers only capture keystrokes they can interpret such as shortcuts keys or text written in form fields, dropping any other keystrokes they do not know how to interpret. Herley and Florêncio leverage this divergence in inputs processing to pollute the sequence of keystrokes on the browser that keyloggers capture without impacting the browser's behaviour: they propose to inject random keystrokes between every key of the user's password. After injection, keyloggers no longer recover the characters of the user password but a long string of random characters with the actual password keys interspersed in between them; recovering the actual password's sequence of characters thus becomes intractable as the number of random keystrokes increases.

The use of dummies for content confidentiality has also found prominent use in privacy-preserving biometrics with error correction codes (ECCs) and so-called *chaff points* central to the design of fuzzy extractors [113]. In their seminal paper on *fuzzy vaults*, Juels and Sudan propose a scheme to hide a secret s in a public *vault* ϑ_S so that only those who know an underlying set of elements S are able to obtain s [312]. The vault is *fuzzy* because it also allows those that produce a set S' close to S (*"close"* in that they only differ in a previously defined number of elements) to open the vault. Juels and Sudan's fuzzy vault scheme works as follows. They encode the secret s as a polynomial p (e.g. through an embedding of s in p's coefficients) over a single variable ϕ. Then, they evaluate the polynomial over the set of elements in S (which they treat as ϕ-coordinate values) producing a *codeword* as a set of pairs (ϕ_i, λ_i), where ϕ_i

represents the elements in S and $\lambda_i = p(\phi_i)$. The number i of elements depends on the maximum distance between S and S' the fuzzy vault must tolerate. The codeword (ϕ_i, λ_i) enables an ECC to recover the polynomial p and therefore the secret s. However, only those in possession of a set S' close to S must be able to open the vault. To prevent unauthorised access to s, they *obfuscate* the codeword (ϕ_i, λ_i) with random chaff points $\{(\phi_j, \lambda_j)\}$ that lie outside p. This ensures that an adversary unable to produce a set S' close to S feeds a mix of genuine and random chaff points to the ECC, preventing in turn the ECC from being able to recover the polynomial p —and thus s. The security of the scheme depends on the additional number of chaff points, with larger amounts of chaff points providing higher security as the number of alternative polynomials other than p the ECC recovers increases.

Hence, a user of an online service may store a biometric *template* (a set of points that represent e.g. her fingerprint) in a third-party server for subsequent authentication with the online service. To protect the template after unauthorised leakage or theft, either the user or the third-party server may choose to hide the original template among a set of chaff points. While legitimate users can produce at any time a set of points close to the original template (the template derives from their own bodies after all), adversaries obtain an obfuscated template from which they cannot recover the original. Moreover, biometrics are noisy and inconsistent, i.e. with every new reading, a user's fingerprint is likely to be similar but not entirely the same due to measuring errors and misalignments, hence the convenience of a *fuzzy* scheme.

The division across both content and metadata confidentiality may not always be obvious or relevant in the study of chaff, with location privacy a prominent example of this dichotomy [63]. We may classify location privacy as a content confidentiality problem whenever users explicitly provide their location, such as in location based services where users request driving directions between two geographical points or search the nearest restaurants to their current position. Conversely, whenever users implicitly provide their location, such as through their current IP address or through the request of services in mobile edge clouds [276], location privacy becomes a metadata confidentiality problem. Regardless of this classification, what matters in terms of privacy engineering is whether or not information about users' locations is required to sustain the provision of utility users expect and whether system designers face trade-offs between utility and privacy.

Location privacy researchers have proposed the use of dummy locations or trajectories as a mechanism to conceal users' current location or moving habits [276, 321, 358, 482, 565]. The underlying rationale behind using dummies to provide location privacy is analogous to the scenarios above: mixing true and

fake locations so that an adversary collecting users' data cannot learn about their actual locations or moving habits.

We may also generate dummies to conceal both content and metadata. Whereas the generation of dummies for content confidentiality may implicitly obfuscate certain metadata, preventing information leakages from the latter requires dummy generation strategies specifically tailored to that goal. A representative example of the use of dummies for protection of both content and metadata is *private web search*. Search engines such as Google and Bing have the ability to build search profiles on their users, who send their queries for relevant websites on the Internet. Search engines can build *profiles* from individual queries by classifying these queries according to topic, time frame or source (e.g. origin IP). Users' search profiles capture what users are interested in as well as the level and evolution of their interests, potentially revealing sensitive information and enabling further inferences on their personality and habits. Profiles thus leverage not only queries' content but also all sorts of metadata. Consequently, there are a number of solutions that propose to generate *dummy queries* to protect against web search profiling [182, 197, 294, 397, 426, 448, 563]. We devote Chapter 4 to the study of these solutions.

Similarly, since DNS resolvers query DNS directory services on behalf of users, they stand in an unparalleled position to monitor the domain names users request and are capable of logging users' browsing histories. To protect against privacy-invasive DNS resolvers, researchers have proposed the use of *range queries*, i.e. sets of $k-1$ hostnames simultaneously sent with the user's actual requested hostname [207]. Zhao et al. claim that if we randomly generate the set of $k-1$ hostnames with each name's probability similar and "no special trait", the DNS resolver can only correctly guess the true target with a probability inversely proportional to the range query set size, i.e. $1/k$ [572]. Zhao et al.'s scheme assumes an honest-but-curious adversary, namely, the DNS resolver provides a response for each of the k hostnames; the requester then simply filters out (discards) the $k-1$ dummy hostnames it added to her query. Further refinements of this scheme require changes on the DNS protocol with the cooperation of two non-colluding DNS resolvers [573], sending dummy queries to several servers [111, 112] and deploying DHT-based DNS [359].

Lastly, AdNauseam seeks to enable users to conceal which advertisements they click on from advertising networks by automatically clicking on all advertisements present in the websites they visit [295].

Query-response services such as LBSs, web search and privacy preserving-DNS we have just reviewed necessarily assume an honest-but-curious (HbC) adversary, this is, the adversary attempts to filter out dummy queries from the user profiles it builds but does not actively disrupt service provision, i.e. an adversarial

service provider sends responses to all queries (both real and dummy) back to the user, it does not ignore queries nor does it refuse to provide responses. Moreover, these schemes also generally assume an uncooperative service provider and users uninterested in service personalisation dependent on previous queries, as otherwise the design needs to face trade-offs between the data users need to reveal for personalisation and the resulting privacy loss.

In fact, the use of chaff or dummies in all the scenarios above represents UPO because dummies do not interfere with what designers consider provides utility to users, e.g. they omit service personalisation based on previous user activity. Dummy traffic in anonymous communication networks may increase delays by increasing traffic load in the network, but does not preclude the user from surfing the web in any way (this is, assuming acceptable delays that prevent a usability nightmare). Rivest's C&W similarly increases the load on the network, but does not undermine or hinder the ability of two people to communicate.

The assumption that dummies have no effect on user utility (UPO) and that adversaries are honest-but-curious further explains why in the scenarios we describe above DGS designers are unconcerned, *as long as adversaries cannot filter dummies out*, about the particular mix of reals and dummies adversaries retrieve. On the one hand, dummies' lack of impact on user utility precludes any motivation to craft DGSs beyond pattern and content hiding, as the particular combination of both reals and dummies a DGS generates is only valuable insofar as it withstands adversarial filtering. In fact, if dummies had an effect on user utility and we attempted to account for their impact by selectively generating a particular set of dummies, strategic adversaries may filter them out from their observations, unfolding a cat-and-mouse game between DGS designers and adversaries that evokes adversarial learning [357] —with DGS designers playing the role of the attacker— and protective optimisation technologies (POTs) [261], where users try to game optimisation systems by selectively changing inputs and constraints to these systems. Hence, when neither UPO nor HbC assumptions hold, i.e. if upon detecting obfuscation or other interventions against data collection adversaries degrade the quality of service or block users altogether, privacy engineers may resort to alternative solutions to game, mislead or bypass these adversaries.

Morphing and mimicry. Steganographic hiding and tool undetectability.

The deployment of dummies in a service works as privacy-enhancing UPO insofar as adversaries powerful enough to disrupt the service (such as the provider) tolerate them, as otherwise users become vulnerable to quality of service degradation or being cut off from the service.

State censors are an archetypal example of this type of adversaries; they block citizens from accessing websites or using online services that run against the state's interests. Whereas citizens may attempt to circumvent censorship by hiding the websites they access behind an anonymous communication network such as Tor, the censor may choose in turn to block services that enable users to circumvent it, e.g. the censor may block Tor altogether, preventing all citizens to use it [552].

Censorship circumvention tools (CCTs) necessarily rely on *undetectability* to succeed, this is, ensuring that a censor cannot distinguish accesses to allowed websites from those to forbidden ones [319, 344]. Researchers attempt to achieve undetectability by, among other strategies, concealing users' accesses to services that enable and mediate access to forbidden websites, such as Tor [97, 211, 291, 552]. These strategies rely on some sort of traffic shaping to make accesses to forbidden resources look like accesses to permitted services such as cloud storage and VoIP or like random packets whose patterns do not match protocols the censor routinely blocks. In particular, a subset of these strategies relies on *mimicry*, this is, shaping a blacklisted communication protocol's traffic so that it looks like a whitelisted protocol's packets [319, 386, 543, 550]. To match the traffic patterns of a whitelisted protocol, these solutions rely on traffic shaping operations such as split and splice, delays, padding and dummy packets.

Using dummies for mimicry reveals a fundamental shift in threat models that, whereas we do not study in detail in this thesis, need to consider in the design of obfuscation tools —as we discuss in Sect. 3.4. The systems we review earlier defend against abusive data collection, injecting dummies to spoil the quality of the data adversaries wish to collect and exploit. We often assume that these adversaries are honest-but-curious thus unlikely to interfere with quality of service while doing their best to filter away as many dummies as possible. In contrast, mimicry as deployed by CCTs seeks to evade adversarial detection, e.g. to prevent blocking and denial of service (DoS) or simply to go unnoticed and avoid grabbing an eavesdropper's attention.

The divergence in threat models further motivates differing dummy generation strategies. In defending against data collection, more dummies (should) translate into better protection by burying real actions into an ever greater level of noise.[16] Besides, successful adversarial filtering causes lower privacy levels, not DoS. In contrast, in enabling mimicry, the protocol or pattern we aim to mimic dictates the amount and type of dummies we require and successful adversarial filtering may prompt DoS, as adversaries may have incentives not only to filter dummies, but to discard utility-bearing data as well, e.g. blocking protocol packets carrying real web browsing requests.

[16] Under the assumption that a sound *dummy generation strategy* is in place.

Lastly, these two threat models point to two different albeit tightly linked approaches to privacy protection: obfuscation and *steganography* [306]. The former protects secrets by degrading the information made available about them. The latter protects secrets by hiding them within other data that ideally makes them invisible, attempting to conceal the existence of the secret itself. Designers need not restrict themselves to one type of strategy and may deploy both dummies that seek steganographic stealth and obfuscation. A paradigmatic example of this dual strategy is StegoTorus, which seeks to circumvent censorship by *padding* Tor traffic to *mimic* an *"innocuous"* cover protocol while generating *dummy* traffic to other, unrelated hosts [550].

Dummies or padding? A note on terminology. Various authors loosely refer to both *adding dummies* or *padding* indistinctly to denote the same set of practices [309, 353, 424, 487, 546] or choose either term to refer to different types of practices [105, 192, 290], e.g. *padding* to denote modifying the size of real messages (e.g. appending zero- or random-valued bytes) and *dummies* to denote adding fake messages to a stream of real messages.

In this thesis we propose and adopt the following convention to refer to either padding or dummies. We say that *padding* denotes the addition of non-information bearing bits (e.g. random or zeroed) to a real entity, i.e. be it bytes to a packet, characters to a message or different weights and components to a *profile* resulting from processing various sources of data. We say that a *dummy* denotes an independent entity shaped in the form of a real one that attempts to function as and elicit the same kind of responses that the real entity it represents does, e.g. a dummy packet, a dummy message or a dummy profile. Padding extends real entities so that, taken as a whole, the padded entity contains both real entity and padding. Dummies are entirely fake and independent from their real counterparts.

Still, adding dummies induces padding and padding requires dummies, e.g. we may consider that the addition of dummy characters or keystrokes results in a padded message while padding the communication between two people requires the generation of dummy messages. If we can define a conceptual hierarchy whereby multiple real entities become an independent, higher level entity of its own, dummies induce padding at that higher level in the conceptual hierarchy, e.g. dummy reads and writes in memory induce padding of memory access patterns, dummy packets in network transmissions induce link padding.

2.4.2 Deceive and lure: decoys

Another role dummies play in computer security is that of decoys, as instruments of deception and distraction. Decoys share many similarities with other dummies in that they attempt to replicate adversaries' targets and prevent them from distinguishing reals from decoys. In his work on deception techniques, Cohen describes the idea underlying decoys as *"to fill the search space of the attacker's intelligence effort with [dummies] so that detection and differentiation of real targets becomes difficult or expensive"* [133]. However, decoys go a step beyond the types of dummies we have reviewed so far in that they are intended to serve as *traps* or *bait* that enable the monitoring and observation of adversaries to gather more information about them. In other words, decoys are dummies with a penchant for entrapment.

Honeypots exemplify the use of dummies as decoys. Scottberg et al. define a honeypot as [479]:

> *"a program that takes the appearance of an attractive service, set of services, an entire operating system, or even an entire network, but is in reality a tightly sealed compartment built to lure and contain an attacker (a sandbox where intruders cannot harm production systems or data) —effectively shunting an intruder safely from production systems for covert analysis."*

Scottberg et al.'s definition further highlights the deception strategy underlying decoys: they seek to lure adversaries to attractive yet bogus services, trap them there and learn about them while, at the same time, protect real targets. Bowen et al. echo this same idea defining honeypots as *"deception-based information resources that have no production value other than to attract and detect adversaries"* [91]. Similarly, Stolfo et al. emphasise the dual value of decoys in their work on *"decoy offensive technology"* against data leakages from the cloud: decoys help detecting unauthorised access to a system or resource and confuse adversaries *"with bogus information"* [507].

Honeypots find numerous applications, from integrating defences against external attackers on critical infrastructure [510] or complementing data security in the cloud [507] to being part of the security and privacy toolkit in *opportunistic networks* [354] and defending against data leaks from insiders, e.g. *enticing* malicious users within an organisation to open decoy documents containing bogus credentials that, once opened, trigger a security alert to notify security administrators [91].

Honeypots have also been reconceptualised and adapted to other contexts. Spitzner defines *honeytokens* to describe individual decoys (e.g. files, credentials, records) that comprise a larger honeypot [502], while Yuill et al. refer to *honeyfiles*, namely, decoy documents that seek to detect external adversaries masquerading as legitimate users with access to a file system [462, 569]. The advent of online social networks (OSNs) has prompted work on *social honeypots*, i.e. fake profiles connected to legitimate users that automatically generate content to lure OSNs spammers, gather information about them and so be able to preemptively detect them [345]; Herrera-Joancomartí and Pérez-Solà extend this concept to propose *social honeynets*, i.e. meshes of decoy profiles that seek to trap web crawlers to prevent them from mass harvesting information in OSNs [279].

2.5 Conclusion

We have started this chapter pointing out the panoply of technologies that rely on or provide *some form* of obfuscation to achieve various security and privacy properties, arguing that obfuscation is too vague a word to neatly delimit the conceptual boundaries over the kind of obfuscation tools we study in this thesis.

To better define those boundaries, we have examined the concept of obfuscation across security and privacy research, identifying three subcommunities each with their own understanding of obfuscation. Software engineers' refer to program obfuscation to denote a set of techniques that modify a program's code, increasing the *complexity* of the reverse-engineering process that enables adversaries to learn the program's purpose or internal structure. Cryptographers similarly refer to program obfuscation to denote a set of techniques that modify a program's code to make it *indistinguishable* from another obfuscated program with the same functionality. Privacy engineers and experts however refer to obfuscation to denote a series of techniques that rely on *data inaccuracy and imprecision* to limit the privacy risk of revealing data to an adversary. Our work belongs in the latter community, we focus on privacy engineering through data obfuscation as data inaccuracy and imprecision.

We have proposed an abstract model of data obfuscation and defined the concepts of personal utility, privacy loss and social utility. Depending on how obfuscation deals with trade-offs between personal utility and adversarial gain, we distinguish between utility-degrading obfuscation (UDO) and utility-preserving obfuscation (UPO): UDO minimises privacy loss by degrading personal utility, whereas UPO minimises privacy loss with no impact over personal utility. We have argued that personal utility alone does not impose trade-offs between utility and

privacy that mandate the use of UDO, whereas the provision of social utility, inherently requiring the disclosure of user data to adversarial parties, requires the use of UDO to address the privacy threats that derive from adversaries with arbitrary background knowledge.

We have examined the technical requirements and constraints that motivate the use of obfuscation, in particular as an alternative to cryptographic and anonymity tools. Because data obfuscation is inherently syntax-preserving, users can unilaterally deploy obfuscation tools to protect themselves against adversarial, uncooperative service providers that refuse the adoption of cryptography-based privacy preserving systems. Moreover, obfuscation offers protection in those situations where user anonymity is neither desirable nor possible, in addition to an additional layer of protection to anonymous users.

In this thesis we focus on UPO and, in particular, in the study of UPO through *chaff*. Chaff represents fake, dummy actions on an online service automatically generated on behalf of the user which bear no relation to the utility users expect from the service. Assuming a service where the provider treats each of a user's service requests with independence from each other and provides responses or outputs that do not depend on that user's previous requests and inputs (e.g. as in personalised services), chaff enables to preserve all utility for a user's service requests while polluting the data an adversary gathers about that user. In fact, chaff enables UPO without the need to determine the set of functions $\{f_i\}$ that provide utility to users.

We have provided an overview of the use of chaff in security and privacy research, highlighting the different types of dummy or fake actions and their uses, i.e. dummies for obfuscation and steganography, decoys as traps and lures for adversaries. All these techniques have in common their ability or aim to preserve users' utility while minimising the threats posed by adversaries. Their applicability however depends on the adversary they defend against as well as the particular protection goal or privacy property, e.g. dummies for obfuscation attempt to starve an honest-but-curious adversary of information, whereas dummies for steganography attempt to prevent an adversary from detecting the existence of data or a denial of service attack.

We devote the remainder of this thesis to the study of chaff-based profile obfuscation tools (Protos). In the next chapter we propose an abstract model and analytical framework for the design and evaluation of Protos and examine key elements in Protos design.

Chapter 3

Chaff-based profile obfuscation

It's a very clever machine. Manipulative. Cunning.
The only problem with Leoben isn't that he lies,
that'd be too easy; it's that he mixes lies with truth.
—Commander Adama, *BSG S01E08*

Profiling describes the process of collecting data on users' behavioural patterns and processing them into representations of aggregate data that provide higher-level information, i.e. *profiles*. Profiling gives rise to several privacy problems, from revealing sensitive or personal information about profiled individuals to enabling subsequent decision-making algorithms, impacting individuals in ways that escape their control.

In this chapter, we provide an abstract model of profiling and introduce *Protos*, utility-preserving chaff-based **pr**ofile **o**bfuscation **to**ols that seek to keep users' profiles confidential, undermining the profiling process itself. We introduce an analysis framework consisting of several privacy measures at the designers' disposal and discuss the rationale behind each type of measure and implications of using them for Protos' analysis and design. Furthermore, we provide an overview of key Proto design issues, focusing on a Proto's dummy generation strategy (DGS) and user interaction, highlighting the role of usability in Protos' design. We conclude with a discussion of the assumptions over the adversary that Protos' design and deployment depends on, examining the threats that other adversaries pose to Protos.

This chapter lays out the model and analytical framework that we rely on to study Protos in this thesis. In Chapters 4 and 5 we instantiate the general Protos model in two particular scenarios, web search and online communication services, respectively. Through these two use cases we further examine and discuss Protos' analysis and design issues.

3.1 An abstract model of profiling

We consider an individual or set of individuals that use one or several digital services, e.g. an online newspaper, a search engine, Internet browsing or any application on a smartphone. We make no assumptions about the number of services or their type. For simplicity however we often refer to *one* individual that uses *one* service.Moreover, we also refer to individuals as *users* and to specify particular users we adopt the set of placeholder names *«Alice, Bob, Charlie,...»* commonly used in cryptography.

The provision of the service involves the generation of *protocol messages* that various components or entities of a system exchange. We variously refer to these messages as *actions*, *events* or by the specific name they take in a particular context or system of choice e.g. *queries* in a web search service or *packets* and *frames* in data transmission protocols such as IP and Ethernet, respectively. We denote the protocol messages users generate as $r \in \mathcal{R}$, with $\mathcal{R}$ the universe of possible user-generated protocol messages. Each protocol message r elicits a service response $o \in \mathcal{O}$ that in turn comprises additional protocol messages users receive from the service provider.

Moreover, we consider that the service runs across two trust domains, namely, the *user side* and the *provider side*, according to a classic client-server architecture [73]. Figure 3.1 depicts the system model.

We assume that the user side is free from interference from the service provider, i.e. by default the service provider has neither access to nor visibility over the client side; users' activities only become visible to the provider after the corresponding protocol messages leave the client side. However, as Narayanan points out in his critique of trust models underlying crypto solutions, we acknowledge that this assumption is increasingly becoming less and less realistic in current *"devices or end nodes and the software running on them"*, as *"consumer technology has evolved away from [the client-server] model in the past decade or so. Hardware and software are increasingly vertically integrated and packed together in a way that users can't fully control or modify"* [400]. Any privacy guarantees Protos may be able to offer entirely depend on the security of the software and hardware on which Protos run, i.e. if an adversarial

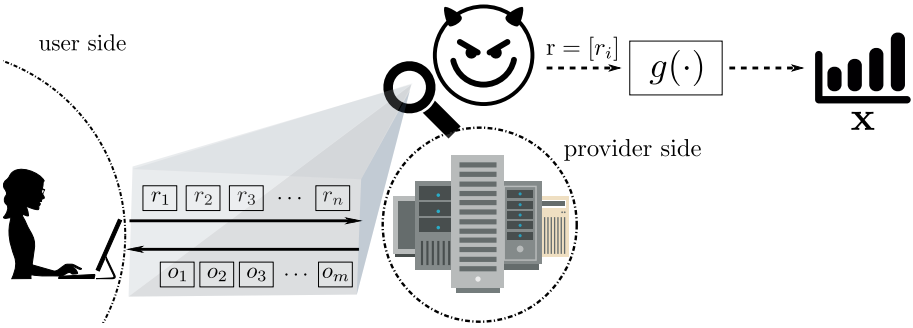


Figure 3.1: Abstract profiling model.

software or software provider is able to compromise the user device, the privacy protections that Protos offer become meaningless. This shift compels us to reconsider the underlying trust model that Protos depend on and opens up additional design and implementation challenges which are outside of the scope of this thesis. We further discuss the implications of the Protos' underlying trust model in Sect. 3.4.1.

3.1.1 Threat model: profiling.

We consider an adversarial entity that monitors the exchange of protocol messages within the system or collection of systems that enable a digital service. We refer to this entity as the *adversary* or *profiler*, indistinctly.

The adversary processes a user's protocol messages $r \in \mathcal{R}$ into a *profile*. We model a profile $\mathbf{x}$ as a multinomial distribution $\mathbf{x} = \{x_i\}$, where each component x_i represents the *probability* that a *profiling strategy* —namely, a function $g(\cdot)$— assigns to a category i the profiler is interested in. The choice of categories i and the meaning or interpretation of probabilities $\{x_i\}$ depend on the *profiling function* g. Hence, the adversary obtains profile $\mathbf{x}$ as:

$$\mathbf{x} = g([r_1, r_2, \ldots, r_n]) = g(\mathrm{r}) \tag{3.1}$$

with $\mathrm{r} = [r_1, r_2, \ldots, r_n]$ the sequence of protocol messages user activity triggers. Figure 3.1 depicts adversarial profiling.

We do not make any assumptions about how g interprets and processes sequences of protocol messages to assign a probability x_i to each category i; we simply consider that the adversary chooses g according to its informational needs, what it wishes to learn.

Building user profiles poses a *privacy threat* to users. Profiles may reveal, among other sensitive data, their usage patterns (day, time, frequency), interests and service predilection (among the options available in the demanded service) as well as users' personal details that even if not explicit in their use of the service the adversary may deduce, i.e. *inferences* from data. In this abstract model we do not limit or make any assumptions about the particular privacy threats users worry about. We simply assume that profiles disclose information that violates users' privacy.

3.1.2 Countering profiling: profile obfuscation tools

A ***pr****ofile* ***o****bfuscation* ***to****ol* (Proto) automatically generates dummy activities on a service on behalf of a user, its *goal* to prevent an adversary from obtaining the user's profile $\mathbf{x}$.

Protos simulate user activity on the service by generating dummy protocol messages d from a universe of protocol messages $\mathcal{D}$. A Proto's *dummy generation strategy* (DGS) governs the selection of dummy messages. The Proto mediates individuals' use of the service by intercepting protocol messages r that derive from real users' activities and delaying or modifying them according to the DGS. We however note that any modifications the Proto performs on users' activities are utility preserving, i.e. modifications may increase *cost* yet never degrade utility.

Protos operate on the user side, which as we have mentioned in Sect. 3.1, we assume to be out of adversarial reach and therefore trustworthy, i.e. the user side is free from adversarial tampering or interference. Similarly, we assume the Proto's implementation to be secure and resistant to adversarial tampering, i.e. we consider Protos' resistance to adversarial tampering to be a *security problem* orthogonal to profiling.

By virtue of generating dummy activity, the Proto pollutes the flow of protocol messages the adversary collects. The adversary no longer sees a sequence of real messages r. Instead, it observes a combined sequence of real and dummy messages $\mathrm{q} = \mathrm{r} * \mathrm{d}$, with $*$ denoting the operation of interleaving both real and dummy messages. Hence, if the adversary processes the sequence q with profiling strategy g, it retrieves an *observed profile* $\mathbf{y} = g(\mathrm{q})$ that should no longer inform the adversary about the original or real profile $\mathbf{x}$. However, we consider a strategic adversary that attempts to undo the obfuscation the Proto injects in an attempt to recover the original profile $\mathbf{x}$, as we detail in the following section.

3.1.3 Adversary model

We consider the following adversary model in the design and evaluation of Protos.

Goals. The adversary's goal is to obtain a user's profile $\mathbf{x}$, with $\mathbf{x}$ as defined in Eq. 3.1.

Capabilities. The adversary is able to eavesdrop on a subset (potentially all) of the service's protocol messages exchanged between the user and server side. Moreover, the adversary is able to detect whether or not a user deploys a Proto and retrieve, test and examine the user's Proto to devise its attack strategies in accordance to the Proto's design. The adversary has an indiscriminate amount of *background* or *auxiliary* information on users. This information may relate to a particular individual, to an "average" user of the service or to the general population at large.

Strategies. As a *strategic* adversary, it does its best to retrieve $\mathbf{x}$, *filtering* as many dummy protocol messages d as possible to recover an approximation $\hat{\mathbf{x}}$ of the real profile with as little noise as possible, i.e. ideally $\hat{\mathbf{x}} = \mathbf{x}$.

Still, we assume that the adversary does not interfere with users' service requests —regardless of its ability to do so, which we make no assumptions about—, i.e. it does not drop, delay or modify any of the protocol messages, either real or dummy. The adversary is, in short, *honest-but-curious*.

As an example of this kind of adversary that we focus on throughout this thesis, we consider an *honest-but-curious adversarial service provider*. A service provider's ability to monitor *all* user activity within the service places it in an unparalleled position to profile its users. We however acknowledge that a provider's incentives may not always align to enable Protos users to continue using the service; in Sect. 3.4.1 we discuss the factors that may encourage a provider to either allow or ban Protos as well as solutions to discourage, prevent or bypass a service provider's Protos ban. Lastly, we note that we use the terms *adversary*, *service provider* and *profiler* interchangeably in the remainder of this thesis.

Table 3.1 summarises the notation we have introduced in this section so far and anticipates further notation we introduce in the remainder of this chapter.

3.2 Analysis

We state in the previous section that the goal of a Proto is to prevent an adversary from obtaining a user's profile $\mathbf{x}$. In this section we provide an

Symbol	Meaning	Symbol	Meaning
r	*Real* user action, event or protocol message	R	R.v. of real actions
r	Ordered sequence of real actions $[r_1, r_2, \ldots, r_m]$	R	R.v. of sequences of real actions
$\mathcal{R}$	Universe of r	$\mathbf{R}$	Universe of sequences r
d	Dummy action	D	R.v. over dummy actions
d	Ordered sequence of dummy actions $[d_1, d_2, \ldots, d_m]$	D	R.v. over sequences d
$\mathcal{D}$	Universe of dummies d	$\mathbf{D}$	Universe of sequences d
q	Action (real or dummy)	Q	R.v. over actions q_i
q	Ordered sequence of actions $[q_1, q_2, \ldots, q_m]$	Q	R.v. over ordered sequences q
$\mathcal{Q}$	Universe of events q	$\mathbf{Q}$	Universe of sequences q
$\mathbf{x}$	User real profile	$\mathbf{y}$	User's observed profile
X	R.v. over real user profiles	Y	R.v. of observed profiles
$\mathcal{X}$	Universe of $\mathbf{x}$	$\mathcal{Y}$	Universe of $\mathbf{y}$
$\hat{\text{r}}$	Sequence the adversary recovers	$\hat{\mathbf{x}}$	Profile the adversary recovers
Ω	Obfuscator	g	Profiling strategy
ℓ	Distance	$\mathbb{E}$	Expected value
H	Shannon's entropy	H_∞	Min-entropy
I	Mutual information	$\mathcal{L}_\infty$	Min-entropy leakage
C	Channel capacity	C_∞	Min-capacity
$(\,)_\beta$	Adversary's belief	$\mathcal{G}$	Adversary's information gain
$\mathbf{E}$	Expected estimation error		

Table 3.1: Summary of notation.

analytical framework to determine the extent to which a Proto meets that goal, assisting in turn the selection of Protos's general design principles that we explore in the next section.

We distinguish between two approaches in the analysis of Protos: *mechanism-centred* (MCA) and *attack-centred* (ACA). Each of these approaches comprises a range of metrics that enable privacy engineers to evaluate Protos according to

various goals. Mechanism-centred approaches analyse Protos with independence of any particular adversary or attack strategy, they focus on the study of how Protos obfuscate, the relation between their inputs and outputs, rather than on the exploits an adversary may deploy to undermine them. Conversely, attack-centred approaches analyse the ability of an adversary to undermine a Proto. They consider a particular attack or family of attacks that they deploy against the Proto, evaluating the success of the adversary at retrieving a user's profile $\mathbf{x}$. Figure 3.2 illustrates how MCA focuses on the relationship between a Proto's input sequence r and its output sequence q, abstracting from further adversarial filtering, whereas ACA takes into account the adversary's attack that produces a filtered sequence $\hat{\mathrm{r}}$.

One conceptual difference between MCA and ACA worth noting is that MCA measures relate to a Proto's *leakage*, this is, how much information Protos leak or disclose, regardless of what an adversary does with that information. ACA measures on the other hand relate to adversarial *retrieval*, this is, the extent to which an adversary takes advantage of the information the Proto leaks and invades users' privacy. In other words, MCA measures capture the effectiveness of the obfuscation mechanism alone, abstracting away from particular adversarial details, whereas ACA measures capture the combined, intertwined effect of obfuscation and adversarial attack, i.e. they capture a Proto's effectiveness through a measure of adversarial success.

Yet another conceptual difference between MCA and ACA relates to their role in assisting Protos' design and evaluation. MCA measures abstract away from particular adversaries and attack strategies; hence, they assist Protos' design as generic constraints or protection goals, i.e. privacy engineers may require that a Proto satisfies a particular measure of privacy without the need to describe an adversary's attack strategy and background knowledge down to the last detail. ACA measures however take into account precisely those details about the adversary, hence, they better assist evaluation of Protos' effectiveness in particular scenarios against particular adversaries. We further discuss the suitability of MCA and ACA measures —over which there is no consensus— in Sect. 3.2.3.

3.2.1 Mechanism-centred analysis

MCA focuses on assessing Protos' effectiveness based on the obfuscation mechanism or dummy generation strategy alone, i.e. they do not consider a particular adversary or attack. Rather, as Fig. 3.2 illustrates, MCA examines the relationship between non-obfuscated inputs and obfuscated outputs to

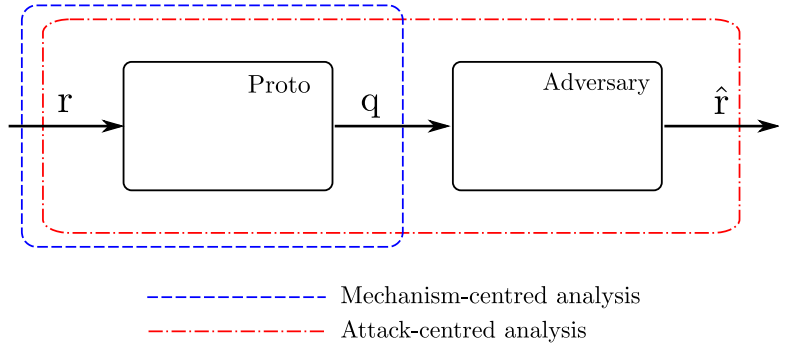


Figure 3.2: Process flow conceptualisation of mechanism-centred and attack-centred analyses.

determine the extent to which an adversary may exploit a Proto's output to retrieve the original input.

We distinguish two families of metrics that fall within mechanism-centred analysis: measures of *indistinguishability* and measures of *leakage*.

Indistinguishability.

Indistinguishability measures a Proto's ability to generate output sequences of protocol messages $\{q_i\}$ so that for any particular output q it is impossible to *distinguish* the sequence of *real* protocol messages r the Proto took as input.

The notion of indistinguishability has its roots in cryptography [107]. To assess the security of cryptosystems, cryptographers rely on various tests of indistinguishability such as resistance to chosen-plaintext attacks [139] or chosen-ciphertext attacks [398, 440]. While each of these tests considers a different setting as to what an adversary is able to obtain from a cryptosystem, the essence of an indistinguishability test is the following. The adversary (e.g. in public-key cryptography, a probabilistic polynomial-time Turing machine) selects two messages m_0 and m_1 and sends them to an *encryption oracle*. The oracle flips a coin to determine the value of a bit $b \in \{0, 1\}$ and outputs c_b, the encryption of m_b. If the adversary cannot determine the value of b with higher probability than $1/2$, namely, random choice, then the cryptosystem is *secure in terms of indistinguishability* [142]. Indistinguishability entails *semantic security* [549], which in turn is the computational analogue to *perfect secrecy* [244], a major goal in the design of secure cryptosystems.

Several works have proposed privacy definitions based on the notion of indistinguishability [278, 562], the most prominent to date being that of *differential privacy* (DP) [189] . Initially defined as ϵ-indistinguishability, DP requires that the outcome o of a function f over a database DB is equally likely (within a multiplicative factor e^{ϵ}) if the input to the function is a database DB' that differs from DB in only one row —supposedly capturing the contribution of an individual [322]—, for all pairs of databases DB and DB' and all possible outcomes o of the function f. In other words, input databases DB and DB' should be ϵ-indistinguishable based on outcome o alone, i.e. an adversary should not be able to distinguish whether it was DB or DB' that produced o.

Similarly, we may rely on notions of indistinguishability in the analysis of profile obfuscation. Let us consider an obfuscator Ω that takes as input a sequence of real messages $\mathrm{r} \in \mathbf{R}$ and outputs a sequence $\mathrm{q} \in \mathbf{Q}$. Ideally, we would like that Ω outputs q with similar probability for any r, so that an adversary upon retrieving q cannot determine which r has engendered it, consequently being unable to retrieve $\mathbf{x} = g(\mathrm{r})$. We therefore aim to measure the ability of Ω to generate indistinguishable outcomes. To do that, we first borrow Duchi et al.'s definition of ϵ-LDP [185] to define *ϵ-profile indistinguishability* (ϵPI).

Definition 3.2.1. ϵ-profile indistinguishability (ϵPI). Let r and r' be sequences of real actions from the universe $\mathbf{R}$ of possible sequences users generate. Let q be a sequence of both real and dummy actions from the universe of possible sequences $\mathbf{Q}$ a Proto generates and S a subset of sequences q. We define *ϵ-profile indistinguishability* of an obfuscation mechanism Ω as:

$$\epsilon = \sup_{S \in \mathbf{Q},\ \mathrm{r},\mathrm{r}' \in \mathbf{R}} \left| \ln \frac{\Omega(S \mid \mathrm{r})}{\Omega(S \mid \mathrm{r}')} \right| \tag{3.2}$$

where $\Omega(S|\mathrm{r})$ represents the conditional probability of the obfuscator Ω generating a subset S of output sequences q given real sequence of actions r. By convention [27], we consider $|\ln \frac{\Omega(\mathrm{q} \mid \mathrm{r})}{\Omega(\mathrm{q} \mid \mathrm{r}')}| = 0$ when $\Omega(\mathrm{q} \mid \mathrm{r}) = \Omega(\mathrm{q} \mid \mathrm{r}') = 0$ and $|\ln \frac{\Omega(\mathrm{q} \mid \mathrm{r})}{\Omega(\mathrm{q} \mid \mathrm{r}')}| = \infty$ when either $\Omega(\mathrm{q} \mid \mathrm{r}) = 0$ or $\Omega(\mathrm{q} \mid \mathrm{r}') = 0$.

An obfuscator that ensures ϵ-indistinguishability for a bounded ϵ is effectively a *locally differentially private* mechanism.

Definition 3.2.2. ϵ-local differential privacy. An obfuscator Ω is ϵ-locally differentially private if:

$$\sup_{S \in \mathbf{Q},\ \mathrm{r},\mathrm{r}' \in \mathbf{R}} \frac{\Omega(S \mid \mathrm{r})}{\Omega(S \mid \mathrm{r}')} \leq e^{\epsilon} \tag{3.3}$$

Parameter ϵ is therefore a *multiplicative measure* of the distance between distributions, denoting how distinguishable input real sequences r and r' are

given output sequence q. A Proto's obfuscation mechanism Ω induces the same output distribution over q given r or r$'$ when $\epsilon = 0$, resulting in *perfect indistinguishability*; conversely, when $\epsilon = \infty$ there is at least one q_i that unequivocally leads back to either r or r$'$, resulting in *perfect distinguishability* between r and r$'$ for that particular q_i.

Definition 3.2.3. Statistical distance. We consider the *statistical distance* (SD) between the probability distributions over sets of output sequences $S \in \mathbf{Q}$ of a profile obfuscation mechanism Ω given input sequences r and r$'$ as:

$$\mathrm{SD}(\Omega(\mathrm{R}), \Omega(\mathrm{R}')) = \sup_{S \in \mathbf{Q},\ \mathrm{r},\mathrm{r}' \in \mathbf{R}} \left| \Omega(S \mid \mathrm{r}) - \Omega(S \mid \mathrm{r}') \right| \tag{3.4}$$

where $\Omega(S \mid \mathrm{r})$ represents the conditional probability of obfuscator Ω generating a subset of output sequences S given input real sequence r.

We note that ϵ-indistinguishability represents a worst-case *multiplicative distance* in that regardless of how statistically close distributions $\Omega(\mathrm{q} \mid \mathrm{r})$ and $\Omega(\mathrm{q} \mid \mathrm{r}')$ are, $\epsilon = \infty$ when one distribution assigns a zero-value in a particular q and the other assigns a non-zero. Conversely, *statistical distance* may be arbitrarily low even when one q_i leads to perfect distinguishability, e.g. if $\Omega(\mathrm{q} \mid \mathrm{r}) > 0$ and $\Omega(\mathrm{q} \mid \mathrm{r}') = 0$ [189].

Choosing between ϵ-indistinguishability or statistical distance hence depends on how stringent a definition of indistinguishability we seek, further having implications in mechanism design; e.g. we may choose to construct an obfuscator that ensures a certain degree of indistinguishability for *every* input sequence r_i or choose to disregard certain sequences r_j, providing no guarantee of indistinguishability for those sequences but still ensuring that the probability of any sequence q is not much greater or smaller for any input r or r$'$. The former requirement imposes a ϵ-DP bound, whereas the latter we can satisfy with an SD bound.

Similarly, as has already been amply discussed in the literature [188, 332, 526] we may relax ϵ-differential privacy into (ϵ, δ)-differential privacy. Dwork et al. introduce the concept of δ-approximate ϵ-indistinguishability to relax the strict requirement that ϵ-indistinguishability imposes when the probability of some input-output combinations *"are not specially likely"* [188], i.e. ensuring ϵ-differential privacy with a leeway additive factor δ. In other words, δ bounds the probability that an obfuscator leaks *"much more information than for ϵ-differential privacy"* [377].

Definition 3.2.4. (ϵ, δ)-local differential privacy. An obfuscator Ω is (ϵ, δ)-locally differentially private [52] if for any pair of input sequences r, r' and any subset of S over the range of output sequences $\mathbf{Q}$:

$$\Omega(S \mid \mathrm{r}) \leq e^{\epsilon} \Omega(S \mid \mathrm{r}') + \delta \tag{3.5}$$

with $\delta = 0$ becoming ϵ-LDP

Hence, (ϵ, δ)-local differential privacy gives an obfuscator a degree of flexibility δ to disclose more information than if purely ϵ-differentially private. Kasivisiwanathan and Smith warn however that (ϵ, δ)-differential privacy is only meaningful for values δ comparatively smaller than ϵ, as otherwise the definition reverts to statistical distance and loses the stringent privacy properties that a multiplicative distance provides [314, 315].

We may introduce an additional relaxation that ties the indistinguishability requirement to a distance or similarity between real sequences of actions $\ell(\mathrm{r}, \mathrm{r}')$, thereby explicitly defining that distant or dissimilar real sequences of actions need not be as indistinguishable from each other as similar sequences [27]. *"Pure"* ϵ-LDP imposes that an obfuscator Ω engenders a similar probability distribution over output sequences q for *any* input sequence r or r'. This means that even if r and r' are extremely dissimilar sequences, Ω must add sufficient chaff to bound the dissimilarity between the probability distributions over the resulting sequences q and q' by parameter ϵ. Conversely, by incorporating a notion of distance ℓ we impose that Ω adds enough chaff to make similar sequences r and r' indistinguishable, yet not so much chaff that *any* two sequences are equally indistinguishable. In other words, the more similar, the more indistinguishable two sequences are. Such parametrisation thus becomes useful in scenarios where indistinguishability is most important between similar sequences, e.g. in the context of private web search, a user may wish to conceal her interests within a particular niche and related topics (e.g. strategy board games, 1920s' modernist literature, new queer cinema) rather than conceal her interests overall. Moreover, applying such a distance-based relaxation enables designers to navigate trade-offs between a limited budget of resources for obfuscation (e.g., bandwidth) and privacy, as we propose in Sect. 3.3.2.

Definition 3.2.5. (ϵ, ℓ, δ)-local differential privacy. An obfuscator Ω is (ϵ, d, δ)-locally differentially private [52] if for any pair of input sequences r, r' and any subset S of output sequences q:

$$\Omega(S \mid \mathrm{r}) \leq e^{\epsilon \cdot \ell(\mathrm{r}, \mathrm{r}')} \Omega(S \mid \mathrm{r}') + \delta \tag{3.6}$$

with $\delta = 0$ becoming (ϵ, ℓ)-LDP.

Hence, (ϵ, ℓ, δ)-LDP introduces vulnerabilities if users require indistinguishability between sequences of actions which lie far away in ℓ-space. (ϵ, ℓ, δ)-LDP relies on assumptions on what it means for users that the adversary cannot distinguish across sequences of actions. To illustrate this, let us consider two GPS users, Alice and Bob, that wish to escape location profiling. Alice wishes to prevent the profiler from determining with precision her location, e.g. she does not mind

that the adversary knows she is in the city she resides, but prefers not to reveal whether she is at home, at work or at any other location within the city. Bob on the other hand is a frequent traveller that wishes to prevent the profiler from determining the city in which he is and is in fact unconcerned about the adversary learning his precise position within the city, for having background information about his job already discloses where he will be (e.g. headquarters of a company, government offices or religious temple). An (ϵ, ℓ, δ)-LDP with ℓ the Euclidean distance satisfies Alice's privacy requirements, for locations within the city are more indistinguishable than two locations in two separate cities. On the other hand, the same Proto with the same ϵ value offers worse privacy guarantees to Bob, as it spends most of its dummies to better conceal locations within small radiuses, rather than generate alternative locations in far-away cities. Hence, by incorporating a distance ℓ into ϵ-LDP, we restrict the privacy guarantees the measure offers with respect to a particular interpretation of privacy.

Moreover, we note that we formalise the definitions above with respect to the sequences of real protocol messages r and obfuscated sequences of messages q instead of the real profile $\mathbf{x} = g(\mathrm{r})$ and observed profile $\mathbf{y} = g(\mathrm{q})$ to avoid making any assumptions about the effect of the profiling strategy on the distribution $\Omega(\mathbf{y} \mid \mathbf{x})$ with respect to $\Omega(\mathrm{q} \mid \mathrm{r})$. Indeed, depending on the profiling strategy g, several sequences $\{\mathrm{q}_i\}$ may map to the same observed profile $\mathbf{y}$ so that the distance $\mathrm{SD}(\Omega(\mathbf{y} \mid \mathbf{x}), \Omega(\mathbf{y} \mid \mathbf{x}')) < \mathrm{SD}(\Omega(\mathrm{q} \mid \mathrm{r}), \Omega(\mathrm{q} \mid \mathrm{r}'))$, yet the adversary retains the ability to attack the Proto with information on $\Omega(\mathrm{q} \mid \mathrm{r})$, in turn undermining the expected level of indistinguishability. We further illustrate the consequences of incorporating assumptions about the distance across sequences and adversarial profiling functions in the case of private web search, which we study in Chapter 4.

Information leakage.

Information theory represents an alternative to measures of statistical distance, conceptualising a Proto as a communication channel through which an individual transmits a sequence of messages r to an adversary, who receives a *noisy* sequence of messages q, as Fig. 3.3 shows. Information theory enables the assessment of a Proto's ability to protect profiles by measuring the amount of information the channel (the Proto) *leaks*, i.e. how much information q carries about r. If the channel is *"perfectly noisy"* sequence q provides no information on the input sequence r, only noise; as a result it is no easier to determine profile $\mathbf{x}$ after obtaining q than before, i.e. with *prior information* about $\mathbf{x}$ alone. Conversely, if the channel introduces no noise, output q univocally reveals channel's input sequence r, from which an adversary trivially computes $\mathbf{x}$.

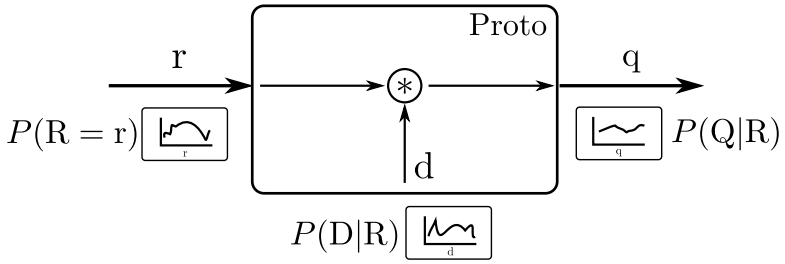


Figure 3.3: A Proto as a noisy channel. Output sequences q result from the tranmission of sequences r through a Proto that adds sequences of dummies d.

As alternative conceptualisations of the same problem, there are equivalences between information theoretical and statistical distance measures [149]. Designers may therefore choose either set of analytical tools in their assessment of Protos according to their needs. In this section we describe the differences in interpretation and meaning of both sets of tools to inform that selection process.

Let us consider a user that relies on a Proto to obfuscate her profile $\mathbf{x}$ and an adversary that captures the observed sequence q and attempts to recover $\mathbf{x}$. We consider that, before acquiring the observed sequence q, the adversary has an *initial uncertainty* on what the user profile $\mathbf{x}$ may be based on *prior information* alone. Subsequently, once the adversary obtains q she uses the information q leaks about r to better determine the value of $\mathbf{x}$.

Smith informally defines [492]:

$$\textit{initial uncertainty} = \textit{information leaked} + \textit{remaining uncertainty}$$

so that

$$\textit{information leaked} = \textit{initial uncertainty} - \textit{remaining uncertainty}$$

proposing to quantify *uncertainty* with *entropy*.

We thus consider a Proto as a *leaky channel* that takes as input a sequence of actions $\mathrm{r} = [r_1, r_2, \ldots, r_{m_r}]$ of length m_r and outputs, according to some obfuscation function Ω, a sequence of actions $\mathrm{q} = [q_1, q_2, \ldots, q_{m_q}]$ of length $m_q \geq m_r$. Random variables R and Q characterise the probability distribution over the input and output sequences $P(\,\mathrm{R} = \mathrm{r}\,)$ and $P(\,\mathrm{Q} = \mathrm{q}\,)$, respectively.

Definition 3.2.6. Shannon's entropy. Let r be a sequence of real protocol messages $[r_1, r_2, \ldots, r_m]$ and R the discrete random variable that determines the probability distribution over the universe $\mathbf{R}$ of possible real sequences. We define the entropy H of random variable R as:

$$\mathrm{H}(\mathrm{R}) = \mathbb{E}(-\log(\,P(\,\mathrm{R}\,)\,)\,) = -\sum_{\mathrm{r}\in\mathrm{R}} P(\,\mathrm{r}\,)\log(P(\,\mathrm{r}\,)) \tag{3.7}$$

with logarithm base 2 giving the entropy in *bits*.

Shannon's entropy is a measure of the *uncertainty* over the value a random variable may take. Measured in bits, it provides a lower bound of *the average number of binary questions an adversary must ask to determine the value of the random variable* [104]. Entropy H(R) thus measures Smith's *initial uncertainty* (or *a priori* uncertainty) about the input sequence r that enters a Proto, whereas H(Q) measures the adversary's a priori uncertainty about the value the Proto outputs. Random variables R and Q are not however independent but linked through the obfuscation function or leaky channel Ω. Once the adversary acquires q, the uncertainty about the input sequence r decreases due to the relationship between both random variables according to the *conditional probability* $P(\,\mathrm{r} \mid \mathrm{q}\,)$.

Definition 3.2.7. Conditional entropy. Let r be a sequence of real actions $[r_1, r_2, \ldots, r_{m_r}]$, q a sequence of both real and dummy actions $q_1, q_2, \ldots, q_{m_q}$ and R and Q the discrete random variables that determine their probability distribution over the universes $\mathbf{R}$ and $\mathbf{Q}$ of possible sequences, respectively. We define the conditional entropy or *equivocation* H of random variable R given random variable Q as:

$$\mathrm{H}(\mathrm{R} \mid \mathrm{Q}) = -\sum_{\mathrm{r}\in\mathrm{R},\mathrm{q}\in\mathrm{Q}} P(\,\mathrm{r} \mid \mathrm{q}\,)\log(P(\,\mathrm{r} \mid \mathrm{q}\,)) \tag{3.8}$$

The conditional entropy hence represents Smith's *remaining uncertainty* about r once the Proto reveals q. Ideally, $\mathrm{H}(\mathrm{R} \mid \mathrm{Q}) = \mathrm{H}(\mathrm{R})$, which means that the Proto leaks no information, i.e. the uncertainty about r is the same regardless of whether the adversary obtains q; the initial and remaining uncertainty are the same. Conversely, the Proto leaks all information about r when $\mathrm{H}(\mathrm{R} \mid \mathrm{Q}) = 0$, i.e. the uncertainty about r is zero, so the adversary can trivially recover $\hat{\mathbf{x}} = g(\mathrm{r}) = \mathbf{x}$.

Given the entropy of a random variable (r.v.) and the conditional entropy given another r.v., we measure the information one provides about the other, i.e. the information q leaks about r —and vice versa— using *mutual information*.

Definition 3.2.8. Mutual information (MI). We define the mutual information I between random variable R and random variable Q as:

$$\begin{aligned} \mathrm{I}(\mathrm{R};\mathrm{Q}) &\equiv \mathrm{H}(\mathrm{R}) - \mathrm{H}(\mathrm{R} \mid \mathrm{Q}) \\ &\equiv \mathrm{H}(\mathrm{R}) + \mathrm{H}(\mathrm{Q}) - \mathrm{H}(\mathrm{R},\mathrm{Q}) \\ &= \sum_{\mathrm{r}\in\mathrm{R}} \sum_{\mathrm{q}\in\mathrm{Q}} P(\,\mathrm{r},\mathrm{q}\,) \log \frac{P(\,\mathrm{r} \mid \mathrm{q}\,)}{P(\,\mathrm{r}\,)P(\,\mathrm{q}\,)} \end{aligned} \tag{3.9}$$

MI hence measures the *information leaked*,[1] the information the channel or obfuscator Ω reveals about r.

Mutual information is a popular measure of information leakage in the security and privacy literature [12, 61, 145, 149, 561, 578]. Shannon himself initiated the study of "secrecy systems" under information theory [483], showing that such systems provide *perfect secrecy* when $\mathrm{H}(\mathrm{R} \mid \mathrm{Q}) = \mathrm{H}(\mathrm{R}) \Leftrightarrow \mathrm{I}(\mathrm{R};\mathrm{Q}) = 0$. Closest to our work on profile obfuscation is Erola et al.'s *profile exposure level* (PEL), in the context of personalised web search [203]. Erola et al. define PEL as the mutual information between the probability distribution over the search queries a user submits (R) and the probability distribution over the search queries the user *appears to submit* (Q), a combination of a subset of her own queries and a subset of queries she receives from other users via a P2P mechanism. In fact PEL is essentially Eq. 3.9.

MI however provides a measure of the channel's information leakage only *for a particular probability distribution of* R [114, 149]. Mutual information says little about the performance of Ω as a whole, i.e. for any probability distribution other than $P(\,\mathrm{R}\,)$. Hence, to determine the channel's or obfuscator's information leakage in any potential scenario, we need to consider the mutual information between inputs and outputs to the channel over *all* possible input probability distributions, namely, the channel's *capacity*.

Definition 3.2.9. Channel capacity. We define the channel capacity C of an obfuscation channel Ω as the maximum information leakage over all possible probability distributions of the input random variable R.

$$C = \sup_{P_{\mathrm{R}}(\mathrm{r})} \mathrm{I}(\mathrm{R};\mathrm{Q}) \tag{3.10}$$

Ideally, a perfect obfuscator has $C = 0$, i.e. it leaks no information on no matter what input probability distribution. Channel capacity characterises a

[1] Note that the first equivalence $\mathrm{I}(\mathrm{R};\mathrm{Q}) \equiv \mathrm{H}(\mathrm{R}) - \mathrm{H}(\mathrm{R} \mid \mathrm{Q})$ corresponds to Smith's informal definition *«information leaked = initial uncertainty − remaining uncertainty»*.

Proto's ability to obfuscate profiles for *any* input probability distribution of real sequences r, providing a tight upper bound on the channel's information leakage. Chatzikokolakis et al. remark that while mutual information depends on the input distribution and therefore on the systems' users, channel capacity depends only on the obfuscator, not on the input distribution which, even if known, may change after time. Yet Chatzikokolakis et al. also concede that whenever we are able to accurately determine the input distribution, mutual information provides a tighter measure of information leakage as we do not need to consider input distributions under which the obfuscator performs worse [114].

Entropy, mutual information and capacity all provide to Protos analysts and designers intuitive measures of information leakage, the decrease in adversarial uncertainty the Proto enables. Shannon's entropy, in Cachin's interpretation, measures *the average number of binary questions an adversary needs to ask about r to determine its value* [104]; yet this interpretation does not necessarily capture the attacks we anticipate from the adversary.

Among other authors, Smith has prominently questioned the research community's overreliance in Shannon's entropy as a one-size-fits-all measure of uncertainty, echoing previous warnings from Shannon himself that mutual information *"is certainly no panacea"* [484, 493]. Smith shows that in the context of adversaries attempting to learn a secret (i.e. confidentiality) Shannon's entropy underestimates the *vulnerability* of a secret against an adversary that recovers the secret after just one guess (i.e. the first guess) by selecting the highest probability candidate [493]. Pliam also shows that there is an unbounded gap between Shannon's entropy and *marginal guesswork*, related to the minimum number of searches a brute-force attacker must perform to attain a certain probability of success [436]. In the context of profile obfuscation this means that the higher the probability $\max(P(\mathrm{R} = \mathrm{r}))$, the lower the uncertainty of such an adversary, as the more likely sequence r is, the more likely the adversary guesses it correctly; yet Shannon's entropy may remain arbitrarily high as long as there are sufficient alternative sequences r' with non-negligible probability. These critiques illustrate that the measures we choose to evaluate profile obfuscation must align with the attack strategies we envision in the adversary model, e.g. min-entropy better suits the measure of information leakage against single-guess adversaries; marginal guesswork may better capture a system's weakness against brute-force attacks. More recently, the notion of g-leakage provides a generalisation of min-entropy that uses *gain functions* to capture a variety of adversarial attack strategies [22].

We have defined the adversary's goal (in Sect. 3.1.3) as *to obtain a user's profile* $\mathbf{x}$ and the adversary's strategy as *to filter as many dummies as possible.* We make

however no assumptions on the number of attempts or guesses an adversary performs to retrieve $\mathbf{x}$. The adversary may resort to an attack strategy that recovers a single profile $\hat{\mathbf{x}}$ with the highest probability of being $\mathbf{x}$. Alternatively, the adversary may rely instead on a strategy that delivers the top k most likely profiles and carry on under the assumption that any of those profiles may be $\mathbf{x}$ —to lessen the probability of focusing on a single $\hat{\mathbf{x}}$ that does not match $\mathbf{x}$.

In this analysis framework we do not attempt to provide a mapping between information-theoretic measures and the adversaries and attack strategies they best attend to, as this mapping depends on particularities that we cannot anticipate for each possible scenario [375].[2] Rather, we seek to illustrate and provide an interpretation of how to analyse a Proto's privacy protection with information-theoretical measures. Analysts and designers must therefore select the particular information-theoretical measure that best meets their needs.

Still, an adversary that succeeds in recovering $\hat{\mathbf{x}} = \mathbf{x}$ by filtering the observed sequence q that maximises $P(\,\mathrm{r} \mid \mathrm{q}\,)$ has been shown to represent an upper bound on information leakage [22]. Hence, we consider *min-entropy* and the notion of *min-capacity* as a measure of *worst-case leakage* and extend the analysis framework with min-entropy measures to account for the worst-case, as a counterpoint to classic Shannon entropy-based measures. We follow on the work of Smith and others on min-entropy as a measure of information flow to provide the definitions below [180, 330, 493, 492].

Definition 3.2.10. Min-entropy. We define the min-entropy H_∞ of a random variable R as:

$$\mathrm{H}_\infty(\mathrm{R}) = -\log(\max_{\mathrm{r}\in\mathrm{R}}(P(\,\mathrm{r}\,))\,) \tag{3.11}$$

We note that min-entropy equals Shannon's entropy when R is uniformly distributed, i.e. $P(\,\mathrm{R}=\mathrm{r}\,) = \frac{1}{|\mathrm{R}|} \Rightarrow \mathrm{H}(\mathrm{R}) = \mathrm{H}_\infty(\mathrm{R}) = \log|\mathrm{R}|$, as this represents a worst-case uncertainty for the adversary, who has no advantage over any first-guess r.

Definition 3.2.11. Conditional min-entropy. We define the conditional min-entropy of a random variable R given random variable Q as:[3]

$$\mathrm{H}_\infty(\mathrm{R} \mid \mathrm{Q}) = -\log\sum_{\mathrm{q}\in\mathrm{Q}} P(\,\mathrm{q}\,)\max_{\mathrm{r}\in\mathrm{R}} P(\,\mathrm{r} \mid \mathrm{q}\,) = -\log\sum_{\mathrm{q}\in\mathrm{Q}}\max_{\mathrm{r}\in\mathrm{R}} P(\,\mathrm{r},\mathrm{q}\,) \tag{3.12}$$

[2] Alvim et al. provide a first step in this direction motivating various *gain functions* and their significance in particular instantiations of g-leakage [22].

[3] Various authors have noted that there is no universal consensus on the definition of conditional min-entropy. Cachin [104] defines it as $\mathrm{H}_\infty(\mathrm{R} \mid \mathrm{Q}) = -\sum_{\mathrm{q}} P(\,\mathrm{q}\,)\mathrm{H}_\infty(\mathrm{R} \mid \mathrm{Q}=\mathrm{q})$, yet observes that such a definition, even if derived from Shannon's conditional entropy, no longer satisfies the property whereby the conditional on Q is smaller than the marginal on R. In this work, similarly to Smith, we adopt Dodis et al.'s definition [180, 493]. Dodis et al. further justify why favour this definition in the analysis of security properties [180].

Conditional min-entropy $\mathrm{H}_\infty(\mathrm{R} \mid \mathrm{Q})$ measures the worst-case uncertainty of the adversary after obtaining q.

Smith further defines *min-entropy leakage* as the difference between the min-entropy of a channel's input random variable probability distribution and the entropy of the input conditioned on the channel's output probability distribution, thus representing the amount of information the channel leaks in terms of min-entropy [493].

Definition 3.2.12. Min-entropy leakage. We define the min-entropy leakage $\mathcal{L}_\infty$ as:

$$\begin{aligned}\mathcal{L}_\infty(\mathrm{R};\mathrm{Q}) &= \mathrm{H}_\infty(\mathrm{R}) - \mathrm{H}_\infty(\mathrm{R} \mid \mathrm{Q}) = \\ &= \log \frac{\sum_{\mathrm{q}} P(\,\mathrm{q}\,) \max_{\mathrm{r}} P(\,\mathrm{r} \mid \mathrm{q}\,)}{\max_{\mathrm{r}} P(\,\mathrm{r}\,)} = \log \frac{\sum_{\mathrm{q}} \max_{\mathrm{r}} P(\,\mathrm{r},\mathrm{q}\,)}{\max_{\mathrm{r}} \sum_{\mathrm{q}} P(\,\mathrm{r},\mathrm{q}\,)}\end{aligned} \tag{3.13}$$

Smith notes that whereas $\mathcal{L}_\infty$ is analogous to mutual information in that it represents the channel's information leakage given certain input and output random variables, unlike mutual information min-entropy leakage is not symmetric, i.e. $\mathrm{I}(\mathrm{R};\mathrm{Q}) = \mathrm{I}(\mathrm{Q};\mathrm{R})$ but $\mathcal{L}_\infty(\mathrm{R};\mathrm{Q}) \neq \mathcal{L}_\infty(\mathrm{Q};\mathrm{R})$ in general.

Lastly, to capture the min-entropy leakage under *any* input probability distribution, we similarly define a channel's *min-capacity*.

Definition 3.2.13. Min-capacity. We define a channel's min-capacity as:

$$C_\infty = \sup_{\mathrm{R}} \mathcal{L}_\infty(\mathrm{R};\mathrm{Q}) = \log \sum_{\mathrm{q}\in\mathrm{Q}} \max_{\mathrm{r}\in\mathrm{R}} P(\,\mathrm{q} \mid \mathrm{r}\,) \tag{3.14}$$

Köpf and Smith provide a proof of the last equality relation [330, 493].

Independence from adversarial knowledge.

Mechanism-centred analysis (MCA) focuses on the study of a Proto's obfuscation mechanism alone, disregarding external factors to the mechanism itself over which we may have little or no control (e.g. sources of information adversaries exploit unknowingly to the Proto's designers), thereby enabling them to abstract away from such factors.

Under certain assumptions, mechanism-centred analysis provides privacy assurances *independently from adversarial knowledge*. Differentially-private mechanism design represents a prominent example of attempting to enforce

such a property. The guarantee differential privacy offers —namely, that the probability of a privacy breach will be more or less the same whether or not a user participates in the system— holds, under certain assumptions about the data-generation process [189, 322], regardless of what the adversary knows. Indistinguishability measures' independence from adversarial knowledge stems from the fact that they make no assumptions about an obfuscation mechanism's input data probability distribution. Hence, they represent a property of the mechanism itself, unrelated to what the adversary does or knows [191].

Similarly, a channel's (min-)capacity measures the maximum amount of leaked information, providing an upper bound on what the adversary may feasibly extract from the system. Information-theoretical measures do however depend on the mechanism's input data probability distribution. Capacity therefore is not independent from the input data probability distribution per se, yet by considering *all possible* probability distributions, it also becomes a property of the mechanism itself and therefore independent from adversary knowledge [145].

Still, ϵ-indistinguishability and min-capacity provide different types of guarantees. Indistinguishability focuses on the multiplicative distance between the probability of any two inputs $\mathrm{r}_i, \mathrm{r}_j$ given an output q, which ensures a minimum level of privacy for each particular user input; capacity on the other hand provides an *average* of the amount of information the mechanism leaks, yet offers no bounds on the distinguishability between any two profiles: the mechanism may expose some users' real profiles $\mathbf{x}$ at the same time that it leaks very little on others. In fact, ϵ-indistinguishability entails bounds on information leakage but the opposite is not generally true, as previous work studying the relationship between both types of measures shows [21, 51, 145].

Hence, ϵ-indistinguishability is best suited to deal with scenarios with stringent privacy requirements, i.e. where privacy guarantees must hold for every Proto's user and against any possible adversary. However, it is too expensive to guarantee ϵ-DP in many settings, in particular those where the universe of sequences exhibits high dimensionality, leading in turn to high *sensitivity*, like in the context of location privacy [82, 421]. Still, as we illustrate in Sect. 3.3, since one of the key Proto design requirements is indistinguishability between real and dummy actions, it is useful to think about how obfuscation maps real to output sequences as the probability of any real sequence leading to the same output sequence; hence, indistinguishability measures remain useful as a conceptual tool to think about DGS design in general.

On the other hand, by providing an average measure of the protection a Proto affords, information theoretical measures are best suited to scenarios where designers face severe trade-offs between privacy and cost (in terms of the amount of dummies a Proto must generate) and it is possible to sacrifice privacy for

individual users, focusing on the average level of protection a Proto provides to the general population of users.

Furthermore, mechanism-centred analyses provide abstract measures of privacy which, being detached from any particular adversary or attack, are often hard to interpret or decide upon [421]. It is unclear which value to assign to ϵ or δ other than $\epsilon = \delta = 0$ when designing a locally differentially private mechanism with no social utility requirements. The privacy guarantee ϵ-indistinguishability provides is *relative*, i.e. it holds regardless of the adversary's background knowledge, yet does not say what an adversary armed with sufficient knowledge actually learns. To determine the *absolute* level of privacy, we must consider a particular adversary with a concrete instance of background knowledge and attack strategy [488]. Information-theoretic measures on the other hand do capture an absolute level of privacy, i.e. the number of bits of information leakage; however, it is still hard to interpret that number of bits as the risk a Proto's user faces. Besides, information-theoretic measures implicitly assume the adversary possesses accurate information on a mechanism's input data probability distribution, i.e. an accurate prior, thus overestimating the knowledge misinformed adversaries gain from the disclosure of obfuscated data [130, 270]. Conversely, by focusing on specific adversaries and attack strategies, attack-centred analysis (ACA) provide an easier-to-interpret set of measures, arguably better capturing the actual level of user privacy towards those adversaries [488, 490].

3.2.2 Attack-centred analysis

We consider two attack-centred analysis (ACA) measures: *information gain* and *expected estimation error*. The former is a variant of information leakage that measures the amount of information the adversary actually gains instead of the amount the mechanism leaks; it facilitates the introduction of additional constraints on what specific adversaries actually know a priori (accurately or not) about the Protos' input data or additional side knowledge they extract from the Proto. The latter further considers the distance between the profiles $\hat{\mathbf{x}}$ the adversary recovers and the actual profiles $\mathbf{x}$ as a measure of how much better it is in terms of users' privacy that the adversary recovers $\hat{\mathbf{x}}$ instead of $\mathbf{x}$.

Information gain. The information-theoretic measures we introduce in Sect. 3.2.1 assess the amount of information Protos leak by comparing the uncertainty about a random variable's value before and after observing the output of a channel. These measures implicitly assume a powerful, knowledgeable adversary, namely, an adversary that initially knows the Proto's

inputs' probability distribution —thus has uncertainty H(R)— and attempts to determine particular channel inputs from the channel output —resulting in uncertainty H(R | Q).

In practice however, as Clarkson et al. illustrate [130, 131], before obtaining a Proto's output an adversary may expect from an incorrect prior a particular sequence r with little uncertainty. Let us use β as a subindex to denote an adversary's prior or *belief*. When the Proto's output q contradicts the adversary's prior and outputs a sequence q so that, e.g. $P(\,\mathrm{q} \mid \mathrm{r}\,) \to 0$ while $P_\beta(\mathrm{r}) \to 1$, the adversary's uncertainty increases, yet the channel still leaks information that the adversary may exploit to correct her erroneous belief. Conversely, when the Proto's output q strongly supports an erroneous prior, the adversary may wrongly recover a profile $\hat{\mathbf{x}} \neq \mathbf{x}$ with high certainty or, by chance, the actual profile $\mathbf{x}$. These later artifacts however tend to disappear with an increasing number of observations, as the Proto leaks more information.

Moreover, Franz et al. warn against adversaries that obtain side knowledge about a Proto's input-output relation, learning information beyond what both outputs and obfuscation mechanism leak [216], even if, as Hamadou et al. indicate, such side knowledge may also be erroneous [271]. As an example, adversaries may possess background knowledge on the interests of a Proto's user v, enabling them to obtain more information about her profile $\mathbf{x}_v$ than what adversaries with only access to the population's prior $P(\,X = \mathbf{x}\,)$ are able to.

To incorporate an adversary's (potentially inaccurate) prior and side information into the evaluation, previous authors have resorted to *information gain*, defined as the *Kullback-Leibler divergence* between an adversary's *belief* —that includes both prior and side knowledge [271]— and the actual Proto's input value r before and after observation.

Definition 3.2.14. Information gain. Let $b(\mathrm{r})$ and $b(\mathrm{r}|\mathrm{q})$ represent adversary beliefs that the Proto obfuscates some real sequence r before and after observing the obfuscated sequence q, i.e. $b(\mathrm{r}) = P_\beta(\mathrm{R} = \mathrm{r})$ and $b(\mathrm{r}|\mathrm{q}) = P_\beta(\mathrm{R} = \mathrm{r} \mid \mathrm{q})$, respectively. Let $\delta(\mathrm{r})$ represent the probability distribution $P(\,\mathrm{r}\,)$ taking values $P(\,\mathrm{r}\,) = 1$ when $\mathrm{r} = \mathrm{r}_\mathbf{x}$ and $P(\,\mathrm{r}\,) = 0$ otherwise, i.e. the degenerate probability distribution that determines the actual input sequence $\mathrm{r}_\mathbf{x}$ to the Proto. We define the information gain $\mathcal{G}$ of an adversary with belief p_β from an observation q as:

$$\mathcal{G} = D_{\mathrm{KL}}(\,\delta(\mathrm{r}) \,||\, b(\mathrm{r})\,) - D_{\mathrm{KL}}(\,\delta(\mathrm{r}) \,||\, b(\mathrm{r}|\mathrm{q})\,) = \log(P_\beta(\mathrm{r}_\mathbf{x}|\mathrm{q})) - \log(P_\beta(\mathrm{r}_\mathbf{x})) \tag{3.15}$$

with logarithm base 2 giving information gain in *bits*.

Clarkson et al. provide an intuitive interpretation of information gain measured in bits, namely, *"k bits of leakage correspond to a k-fold doubling of the*

probability that the attacker ascribes to reality" [131], e.g. if an adversary assigns a prior probability $P_\beta(\mathrm{r}_\mathbf{x}) = 0.1$ to the correct sequence $\mathrm{r}_\mathbf{x}$ and gains 3 bits of information, the posteriori belief becomes $P_\beta(\mathrm{r}_\mathbf{x} \mid \mathrm{q}) = P_\beta(\mathrm{r}_\mathbf{x}) \cdot 2^3 = 0.8$.

Information gain provides a measure that is specific to an adversary with a particular instance of prior information and side knowledge, thus does not capture a Proto's performance against adversaries with more or less knowledge —accurate or otherwise. Since information leakage measures such as mutual information and capacity implicitly assume an adversary with perfect prior and no side information, one may demand that we subject them to the same criticism. Conceptually however there is a difference between measuring what an adversary learns and what the channel or Proto leaks. The former always depends on adversarial beliefs, whereas the latter is a function of the inputs and outputs to the channel alone.

Adversaries with incorrect prior knowledge are in general of lesser concern in the design of Protos, as wrong prior adversarial *beliefs* hinder adversaries' ability to recover users' real profiles. Hamadou et al. [270] show that min-conditional entropy is always smaller than or equal to the min-conditional entropy of an adversary with wrong beliefs, i.e. $\mathrm{H}_\infty(\mathrm{R} \mid \mathrm{Q}) \leq \mathrm{H}^\beta_\infty(\mathrm{R} \mid \mathrm{Q})$, with H^β_∞ representing the uncertainty of an adversary whose prior *belief* $P_\beta(\mathrm{r})$ is incorrect.

On the other hand, adversaries with accurate side knowledge pose a greater threat than those without, yet designers are unlikely to be able to precisely determine the side knowledge adversaries have or may be able to acquire in the future. In general, designing a tool with a particular instance of adversarial beliefs in mind renders Protos vulnerable to adversaries with beliefs other that what designers account for. However, information gain still assists targeted analyses when we wish to ascertain the ability of particular adversaries in breaching users' privacy, e.g. as part of an audit or privacy impact assessment.

Expected estimation error. In the context of their work on privacy-preserving location-based services (LBSs), Shokri et al. argue that [490]:

> *«Neither the uncertainty metric nor the inaccuracy metric, however, quantify the privacy of the users. What matters for a user is whether the attacker finds the correct answer to his attack, or, alternatively, how close the attacker's output is to the correct answer.»*

Shokri et al.'s definitions of *uncertainty* and *inaccuracy* loosely map to our framework's *information leakage* and *information gain*, respectively. Their

claim echoes our previous observations on the difficulty of interpreting measures that are detached from a specific adversary or attack in return for generality: avoiding assumptions about adversarial beliefs and attack strategies prevents us from estimating how successful the adversary is at retrieving a user's real profile $\mathbf{x}$.

Shokri et al. hence propose *expected estimation error* (EEE), a probabilistic measure of how close the adversary gets to the user profile. We note that none of the previous measures we have defined so far consider the profile $\hat{\mathbf{x}}$ the adversary recovers or the actual user profile $\mathbf{x}$; they disregard them because those are specific to the adversary's profiling strategy, which they abstract away from to provide a measure independent from it. Conversely, expected estimation error incorporates both the adversary's beliefs and its profiling and attack strategies to consider what the adversary retrieves and how 'bad' such result is in terms of user privacy. We borrow Shokri et al.'s definitions of *expected estimation error* and *expected distortion privacy* to define a slight variant of expected estimation error [488, 490].

Definition 3.2.15. Expected estimation error. Let $P(\mathbf{x})$ represent the probability distribution of a user's input *profiles* to the Proto —each profile the result of applying the profiling function g to a user input sequences r— i.e. $P(\mathbf{x}) = P(g(\mathrm{r}))$.[4] Let $P(\mathrm{q} \mid \mathbf{x})$ similarly represent the conditional probability that the Proto outputs a sequence q given that the input sequence r maps to profile $\mathbf{x} = g(\mathrm{r})$. Further, let $P(\hat{\mathbf{x}} \mid \mathrm{q})$ represent the probability distribution that results from the attack strategy *and* (estimated) prior knowledge the adversary relies on to recover a filtered profile $\hat{\mathbf{x}}$ from an observed sequence q and,[5] lastly, let $\ell(\hat{\mathbf{x}}, \mathbf{x})$ represent the distance at which the profile $\hat{\mathbf{x}}$ the adversary recovers and the real profile $\mathbf{x}$ are, this distance according to the user's privacy requirements, i.e. the greater $\ell(\hat{\mathbf{x}}, \mathbf{x})$ is, the better for user

[4] Shokri's definition of *expected distortion privacy* uses $\pi(\mathbf{x})$ instead of $P(\mathbf{x})$, with $\pi(\mathbf{x})$ representing $P(\mathbf{x})$ as given by prior, publicly available information [488]. Shokri notes that $\pi(\mathbf{x})$ does not however denote an adversary's auxiliary knowledge, which is generally unknown, i.e. $\pi(\mathbf{x})$ represents our estimation of what the adversary knows based on publicly available information, not what the adversary *really* knows (e.g. through other information channels that neither we know nor control). This is however merely a matter of orthodoxy on the impossibility of absolute disclosure prevention against adversaries with arbitrary knowledge. Using $\pi(\mathbf{x})$ instead of $P(\mathbf{x})$ is consistent with Shokri's goal, namely, to design an obfuscation mechanism that *optimises* the obfuscation strategy based on previous disclosures assuming an informed adversary that uses an optimal attack strategy. Conversely, we consider an adversary with potentially incorrect prior knowledge and suboptimal attack strategy. Hence, we use $P(\mathbf{x})$ to average over the actual probability of profiles $\mathbf{x}$ and incorporate the adversary's prior knowledge (which, as Shokri, we acknowledge is an estimation, nor the actual knowledge) in the term that Shokri reserves for the inference attack alone, i.e. $P(\hat{\mathbf{x}} \mid \mathrm{q})$. These changes enable us to measure the expected estimation error of adversaries with incorrect priors.

[5] See Footnote 4.

privacy. We define expected estimation error $\mathbf{E}$ as:

$$\mathbf{E} = \sum_{\mathbf{x}} P(\mathbf{x}) \sum_{\mathrm{q}} P(\mathrm{q} \mid \mathbf{x}) \sum_{\hat{\mathbf{x}}} P(\hat{\mathbf{x}} \mid \mathrm{q}) \cdot \ell(\hat{\mathbf{x}}, \mathbf{x}) \tag{3.16}$$

We note how EEE incorporates previous measures' assumptions about a Proto's inputs and operation and further integrates the adversary's beliefs and attack strategy. The first two summations $\sum_{\mathbf{x}} P(\mathbf{x}) \sum_{\mathrm{q}} P(\mathrm{q} \mid \mathbf{x})$, characterise the channel as information leakage measures do, albeit assuming a single input probability distribution, thus with the limitations and decreased generality of mutual information as opposed to capacity (which considers all possible probability distributions) and indistinguishability (which is independent from the probability distribution). The term $\sum_{\hat{\mathbf{x}}} P(\hat{\mathbf{x}} \mid \mathrm{q})$ captures the attack strategy of the adversary as the probability that the adversary recovers a filtered profile $\hat{\mathbf{x}}$ after updating its prior knowledge with its observation of obfuscated sequence of actions q.[6] Lastly, the term $\ell(\hat{\mathbf{x}}, \mathbf{x})$ incorporates a definition of how $\hat{\mathbf{x}}$ contributes to user privacy, namely, it measures how *better* it is for users that the adversary recovers $\hat{\mathbf{x}}$ as opposed to $\mathbf{x}$ [488].

EEE indeed represents the last step in a sequence of measures from more abstract and general —less details about the adversary—, to less abstract and general —more details about the adversary. We depict this idea in Fig. 3.4. MCA measures either omit an adversary's side knowledge and attack strategies (indistinguishability) or assume them implicitly (no side information and Bayesian updating in information leakage measures). ACA measures on the other hand incorporate particular details of an adversary. Information leakage considers a particular instance of background knowledge, while EEE further considers the particular profiles $\hat{\mathbf{x}}$ the adversary recovers, thereby implying that not every profile $\hat{\mathbf{x}}$ the adversary recovers is equally *bad* for the user, which in turn depends on a particular adversarial post-processing strategy, namely, the actions the adversary performs upon such a profile and expected user outcomes thereof.

Gervais et al. [235] further propose to compare the effect of obfuscation on profiling relative to an unobfuscated profile, thereby implying that the profiling strategy itself is imperfect and may benefit the user when profiled *in an advantageous way* as opposed to previous measures that abstract away from the particular profiling strategy. In this framework however we do not consider the posterior effects of profiling on users, as we further explain in Sect. 3.3.

Yet EEE as an absolute measure of privacy conflates the effect on users' privacy of the obfuscation mechanism with the adversary's prior and side knowledge

[6] See Footnote 4

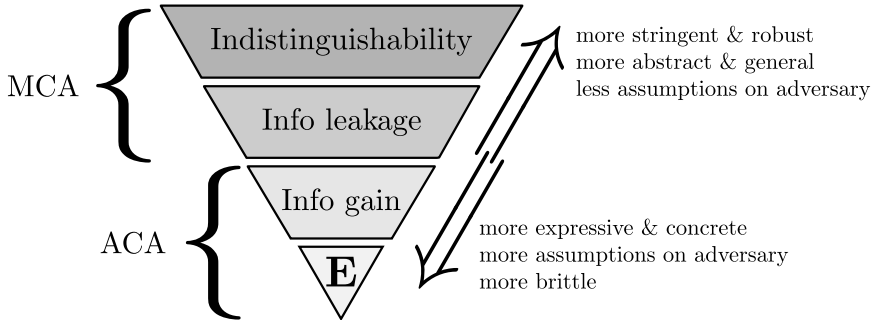


Figure 3.4: Less adversarial assumptions confer measures more generality.

as well as its attack strategy. EEE does not compare the users' privacy before and after the adversary observes the outputs of a Proto, this is, it does not compare the probability that the adversary recovers a profile $\hat{\mathbf{x}}$ *before* seeing any obfuscator's output with the probability that the adversary recovers $\hat{\mathbf{x}}$ *after* observing an obfuscated sequence q. An adversary with an exceptionally good prior may perform equally well without the Proto's output, highlighting the impossibility of absolute disclosure prevention that motivates differential privacy [190]. Hence, even if EEE provides a good measure of privacy against a particular adversary, it does not properly acknowledge the Proto's ability to thwart the adversary.

Lastly, we note that EEE provides an adversary's average error over all possible recovered profiles $\hat{\mathbf{x}}$, as opposed to a worst case that considers an adversary that exclusively selects the filtered profile $\hat{\mathbf{x}}$ that maximises $\max_{\hat{\mathbf{x}}} P(\hat{\mathbf{x}} \mid \mathrm{q})$, as *min-capacity* and *information gain* implicitly do. It is however straightforward to obtain a worst-case measure by replacing the sum $\sum_{\hat{\mathbf{x}}} P(\hat{\mathbf{x}} \mid \mathrm{q})$ with $\max_{\hat{\mathbf{x}}} P(\hat{\mathbf{x}} \mid \mathrm{q})$.

3.2.3 Choosing between MCA or ACA

Disagreements between proponents of different types of measures abound in the literature. Andrés et al. argue that EEE is *"explicitly defined in terms of the attacker's prior knowledge, and is therefore unsuitable for scenarios where the prior is unknown"* [27]; they propose *geo-indistinguishability*, a measure of location privacy based on differential privacy. On the other hand, as a relative measure, the ϵ in ϵ-geo-indistinguishability says very little about what the adversary actually knows about the location of a user, as that requires a specific evaluation that takes an adversary's prior and side knowledge into consideration.

Similarly, Li et al. [350] argue, in the context of website fingerprint defence evaluation, that *"validating [defences] by accuracy alone is flawed [...] when accuracy is low, its corresponding information leakage is far from certain"*, and propose mutual information as a measure of information leakage. Li et al.'s critique echoes a previous point about the difference between mechanism- and attack-centred analyses. ACA measures conflate the performance of the Proto with the attack's prior knowledge and attack strategy. Adversarial success may result from a weak Proto or robust side information, i.e. the adversary may gain little information from the Proto, yet enough to support strong side knowledge and breach users' privacy, thus underestimating the Proto's performance. Conversely, adversarial failure may either result from a robust Proto or flawed side information and a suboptimal attack strategy, with the latter leading to the overestimation of a Proto's performance.

Hence, the choice between MCA and ACA depends on how much generality and stringent a measure of privacy we seek. MCA measures provide stronger, more general privacy guarantees as they do not rest upon specific assumptions on adversarial knowledge, i.e. they are resilient to changes in the attack the adversary deploys and the knowledge it has. The capacity of a Proto does not depend on the side knowledge an adversary has; similarly, differential privacy bounds hold regardless of which adversary we may consider.

However, such generality comes at the expense of expressiveness. MCA focuses on the performance of the mechanism alone, on how much information the mechanism gives away, and does not capture what information the adversary actually recovers, what a particular adversary actually learns. By considering particular adversaries and attacks, i.e. particular *contexts*, ACA provide more expressive measures of the privacy breaches users are subjected to. However, ACA loses the generality that MCA provides, as particular ACA results do not generally hold for other adversaries and contexts, potentially under- or overestimating Protos' performance.

Moreover, MCA and ACA lend themselves to different roles in the design and evaluation process. Indistinguishability constraints such as differential privacy represent strong privacy properties but, in many practical applications, unachievable goals. Moreover, unless differential privacy guarantees are embedded in Proto's design from the beginning, the formulations we provide above, from Eq. 3.2 to Eq. 3.6, do not easily lend themselves to Proto's evaluation. In other words, while we may attempt to design Protos' to guarantee the indistinguishability goals above, it is not trivial to determine the extent to which a Proto that was not designed for such notions of indistinguishability satisfies a particular level of ϵ-, (ϵ, δ) or (ϵ, ℓ, δ)-differential privacy. In this sense, further work that focuses on the quantification of Protos' indistinguishability is required, e.g. along the lines of recent work on anonymous communications

evaluation through game-based indistinguishability definitions [34]. Conversely, ACA measures are best suited to determine the threat that particular adversaries pose to Protos, to evaluate Protos' deployment in particular contexts, taking into account publicly available information that adversaries may exploit or side-channel leakages to which MCA is oblivious.

Lastly, we note that the selection of measures we have provided in this section has allowed us to illustrate the differences between the two types of analyses, *mechanism-centred* and *attack-centred*, yet this set of measures is not comprehensive, with possible variations or combinations of the above also possible, depending upon the particular context or Proto under evaluation.

3.3 Protos' engineering

Proto design requires the formulation of a dummy generation strategy (DGS) that determines which dummies to generate, how many, at which frequency and how to interweave them with the user's real activities. While not the main focus of this thesis, in this section we overview key decision elements in the design and implementation of Protos.

We distinguish three phases in Proto design. First, determining the privacy goals a Proto must fulfil and the adversary the Proto must defend against, which in turn require the selection of a privacy measure to evaluate the subsequent Proto design. Second, dummy generation strategy (DGS) design, which requires determining the amount and type of dummies the Proto must generate as well as the frequency at which it generates them to guarantee the previously defined privacy requirements. Third and last, Proto's implementation, which involves the development of an actual software tool that executes the DGS and users install on their devices to protect their privacy.

3.3.1 Privacy requirements

Earlier in Sect. 3.1.1 we state that we generally consider the threat of *profiling*, i.e. the fact that the adversary collects user data from a service's usage and can extract information from them, without focusing on any particular user concern or privacy goal. In practice however, Protos' design requires the definition of the particular privacy properties we intend to provide. User concerns and adversarial goals inform the privacy properties of choice.

Users' privacy concerns.

User privacy concerns are the ultimate guiding principle in Proto design. Users may have a variety of concerns about profiling, from concerns about what the adversary is able to learn about their interests, their daily routines on the service (when, how often and how long they use the service), how information may leak to other, non-authorised parties or how that information may be misused about them, among many other concerns. Concerns may also range from very broad, e.g. revealing the least possible information to an adversary about their use of the service, to more specific, having little or no interest in concealing certain information while being more conservative about other aspects, placing different levels of sensitivity to the activities they perform on the service, e.g. users of a web search engine may forgo concealing their more mundane interests yet wish to conceal those they consider of higher sensitivity, e.g. that disclose a health condition.

Then, we must determine whether and how Protos can mitigate the privacy concerns that users raise, as Protos may not be able to tackle some of these concerns, e.g. users may raise concerns about the subsequent effects of profiling or targeting based on particular activities, yet Protos may be inadequate to address some of these concerns (see discussion in Sect. 3.4.1). We formalise how Protos tackle users' privacy concerns through privacy properties that capture the technical goal of the Proto that better addresses such concerns, i.e. this involves mapping user concerns to Protos' technical requirements. In Sect. 4.2.1 and 5.2.1 we propose several privacy properties as goals to address a series of underlying privacy concerns in two particular contexts: web search and online communication, respectively.

Adversary model.

The adversary we wish to defend against and the amount of information available we have about it must inform Proto design. Protecting against multiple adversaries about whom we know little requires balancing trade-offs in the level of protection we can afford against several types of attacks and instances of prior and side knowledge. Indeed, in many scenarios we may have little information about the capabilities, attack strategies and motivations of an adversary. Conversely, protecting against a single adversary with explicit adversarial goals and narrowly-defined capabilities allows us to strategically design a Proto to target that adversary in particular.

Furthermore, we may deem it futile to attempt to protect against knowledgeable adversaries that already possess precise information about the user or can

acquire it through alternative channels. We may therefore choose to forgo the protection of that information that we consider the adversary already has or can readily acquire and focus on the obfuscation of additional information items the adversary could acquire from the system where the user requires the Proto.

Operationalisation.

Once we select the privacy properties of interest and the adversary model, we convert the Proto's privacy requirements into a measure that captures the extent to which the Proto fulfils them. To do that, we may instantiate any of the measures we provide in Sect. 3.2, as we illustrate through the two use cases we examine in Chapters 4 and 5. In the remainder of this section we rely on indistinguishability measures to illustrate general aspects of Proto design that are independent from the adversary model. We choose indistinguishability measures because they represent the most stringent privacy guarantees; moreover, indistinguishability intuitively relates to a key requirement of every DGS, namely, that reals and dummies be *indistinguishable*.

We acknowledge that perfect indistinguishability represents an ideal protection goal that is rarely achievable in practice, yet this also enables us to illustrate the stringent requirements of perfect obfuscation.

3.3.2 Dummy generation strategy design

The core component of a Proto is its dummy generation strategy (DGS), which defines which dummies to generate, how many, how often and how to interleave them with the protocol messages that real user activity generates. In this section we review key aspects in DGS design. We note however that the overview we provide in this section does not seek to provide guidance to actual, practical implementation of DGSs. Rather, we seek to highlight the main elements in DGS design and, in the process, show that achieving *ideal* or *perfect* obfuscation is in practice almost always impossible.

To reflect on ideal DGS construction, we borrow the term *'supersequence'* from Wang et al.'s work on website fingerprinting defences [545]. Protos generate *supersequences* q of real sequences r, i.e. a supersequence q combines both the sequence of real actions $\mathrm{r} = [r_i]$ and the additional dummy actions $\{d_i\}$ that a DGS generates.

The DGS is in charge of deciding which dummy actions d_i to generate to obfuscate real sequences of actions r_i through supersequences q_i; the DGS is hence responsible for the mapping $\Omega(\mathrm{Q} \mid \mathrm{R})$ that, as we have seen in Section 3.2,

defines a Proto's effectiveness, i.e. if multiple sequences $\{r_i\}$ have a similar probability $\Omega(q \mid r_i)$, the Proto achieves *indistinguishability* across $\{r_i\}$ and leaks no information about which particular r_i produces q, prompting the adversary to guess based on prior or side information alone.

The pool of dummies.

To create a supersequence, a DGS requires a pool of dummy actions $\mathcal{D} = \{d_i\}$ to generate, viz. all potential actions including not only the type of action, such as a *query* in a web search engine or a *message* in a social networking site, but also details about that particular action that an adversary may exploit to classify or discriminate across actions. These details include, e.g. content or metadata such as length and issuing time. Moreover, to generate viable dummies and enable the service provider to process dummy activity indistinguishably from real activity, the DGS must issue dummies according to protocol description, i.e. the DGS must ensure that dummies follow the same format specification as real protocol messages. Proper dummies' formatting further prevents leaking side-channel information that the adversary exploits to filter dummies out. Besides, determining the proper formatting of dummy messages has implications for resource management, i.e. the bandwidth or processing power requirements that dummy messages impose on both users and system providers.

The ideal *pool* of dummies includes all possible real actions so that $\mathcal{D} = \mathcal{R}$.[7] However, it is not always possible to measure or predict the complete universe of real actions $\mathcal{R}$ and, in practice, the pool of dummy actions may include a subset of real actions $\mathcal{D} \subset \mathcal{R}$ or an overlapping set of actions $\mathcal{D} \cap \mathcal{R} \neq \emptyset \wedge \mathcal{D} \nsubseteq \mathcal{R}$ —e.g. in the context of private web search, it is not possible to determine the universe of real queries a priori, as this depends on infinite combinations of terms as well as the inclusion of new terms;[8] hence we assign an inaccurate prediction $\mathcal{R}_{\text{DGS}}$ of potential queries to the pool.

Averting distinguishing features. In the process of identifying which features about users' actions an adversary may obtain and exploit we must examine which of those distinguishing features are *useful*, i.e. which features contribute to the users' utility and cannot be discarded or modified without utility loss, and which ones are not —we recall from Sect. 2.2 that the notion of utility we consider is limited to that deriving from data disclosure to an adversary— e.g. as we show in Chapter 5, a DGS designer may use content encryption in instant

[7] Generating dummy actions that do not belong to the universe of real actions, $d \notin \mathcal{R}$, represents a waste of resources against informed adversaries who, among other strategies, exploit membership in $\mathcal{R}$ to distinguish reals from dummies.

[8] Mitchell reports that from 2003 to 2012 Google had *"answered 450 billion unique queries"* and that *"16 to 20% of queries that got asked every day [had] never been asked before"* [385].

messaging services to enormously compress the universe of real actions and prevent an adversary from exploiting content to distinguish real from dummy messages, whereas, as we show in Chapter 4, in online web search this is not possible if we consider the search engine provider itself to be the adversary, as it needs to see a query's content to generate the corresponding search results.

More generally, a Proto's DGS may choose to either replace or incorporate real patterns with dummy equivalents or vice versa, i.e. the DGS may attempt to generate dummies that look as if the user generates them [82, 321], or modify the reals so that it looks like it is the DGS who generates them, e.g. replacing scheduled dummies with delayed real actions to fit into a predefined DGS dummy generation pattern [153] or altering the format of real queries to map a predefined universe of equivalent dummy actions —yet the DGS must beware the changes it performs on real actions to prevent utility loss, as otherwise the DGS operates on the realm of utility-degrading obfuscation (q.v. Sect. 4.3.2).

Supersequences.

To achieve indistinguishability between any two real sequences $\mathrm{r}_i, \mathrm{r}_j$, the Proto must output the same q for both r_i and r_j. Hence, the DGS must strive to create supersequences q that contain both r_i and r_j. Figure 3.5 illustrates this process; the DGS ensures indistinguishability between $\mathrm{r}_1, \mathrm{r}_2$ and r_3 when it outputs q with the same probability for any of the r_i it *"contains"*. Hence, to create a supersequence q that multiple real sequences r_i map to, the DGS needs to generate successions d_i of dummy actions that are consistent with the set of actions present in sequences r_i.

The budget of dummies.

The budget of dummies or amount of resources available for obfuscation (e.g. in terms of bandwidth, memory or system capacity) further determines the level of protection a Proto is able to afford and therefore constrains DGS design and supersequence construction. Given an unlimited supply of resources, Protos can provide *perfect security*, (i.e. Protos leak no information from the observation of (obfuscated) user activity; their capacity C is zero) if all possible real actions $r \in \mathcal{R}$ are generated simultaneously at a constant rate independent from real user activity —what we refer to as a *flooding* DGS— resulting in a unique *supersequence* q that any other sequence r_i maps to.

Whereas in settings with a small universe of actions (where $|\mathcal{R}| \to 0$) deploying flooding may be feasible (e.g. in C&W, low-bandwidth communication and

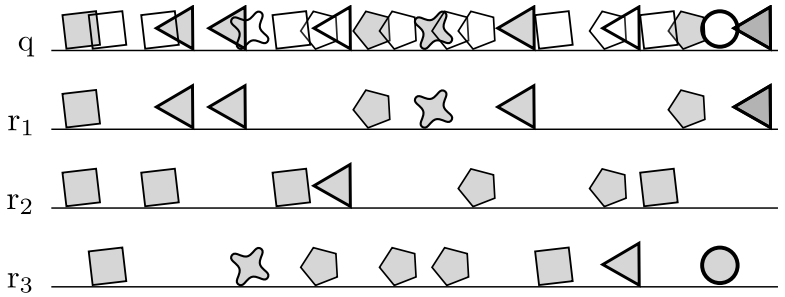


Figure 3.5: Supersequence composition. A Proto outputs supersequence q when input is r_1. Each geometrical form represents an action q, with grey background for real actions r and white background for dummies d. All r_1, r_2 and r_3 are *subsequences* of q, thus indistinguishable.[a]

[a]We note however that the Proto only achieves perfect indistinguishability iff $\Omega(q \mid r_1) = \Omega(q \mid r_2) = \Omega(q \mid r_3)$. An adversary may still exploit the sequences' prior probability $P(r_i)$ to identify the most likely sequence r_i.

restricting the universe of actions to one bit enables flooding by sending the complementary dummy bit to every user bit [453]), flooding becomes unrealisable or its cost prohibitive otherwise, e.g. generating each possible user query in web search (where $|\mathcal{R}| \gg 0$) is both impractical —in terms of bandwidth and (most notably the client's) system capacity— and impossible, as we cannot foresee new user queries in advance.[9] A trade-off between indistiguishability and cost thus arises when a limited budget of resources precludes the possibility to ensure indistinguishability to the level we desire.

Spending fewer resources than needed necessarily translates in diminished indistinguishability, e.g. we may partition the universe of real sequences $\mathbf{R}$ into several *anonymity sets* that map to shorter supersequences [545]. Figure 3.6 illustrates this process. Sequences r_4, r_5 and r_6 map to a supersequence q_2 that does not require the addition of many dummies to each of these sequences; the same occurs with sequences r_7, r_8 and r_9, mapping to supersequence q_3. Hence, we generate less dummies than a unique supersequence requires, however, an adversary that observes q_3 knows that neither r_4, r_5 or r_6 could have led to that supersequence, similarly for q_2 and r_7, r_8 and r_9.

Moreover, a less resource-intensive dummy generation strategy necessarily incorporates a notion of *distance* to the level of indistinguishability, i.e. to save dummies, we map sequences to supersequences *close* in the sequence space

[9]See Footnote 8

with greater probability than to supersequences which are far away. Hence, an adversary can exploit the distance between a supersequence and the subset of real sequences that with higher probability lead to it.

Resource scarcity has further implications for privacy protection allocation across the Proto's user base. To save dummies, Protos must output supersequences q_i that include as many real r_i as possible while keeping the amount of dummies every user has to generate within a limited budget. Hence, Protos under resource scarcity constraints may generate supersequences q_i that include the most probable user sequences $\mathrm{r}_i : P(\mathrm{r}_i) >> P(\mathrm{r}_j)$ and sacrifice rare user sequences $\mathrm{r}_j : P(\mathrm{r}_j) \to 0$ that impose a prohibitive amount of overhead for most other users, thus offering disparate levels of privacy to its user base. Alternatively however, Protos may also require that every user generates an increased amount of dummies that guarantees a minimum level of privacy for all users.

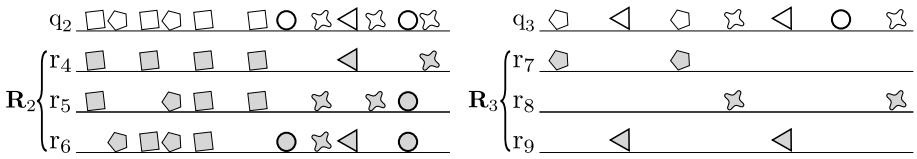


Figure 3.6: Saving dummies. We split the universe of real sequences to create *shorter* common supersequences, resulting in various *anonymity sets* of (similar) real sequences.

Figures 3.7 to 3.10 further illustrate the process whereby Protos produce mappings between input sequences r_i and output supersequences q with decreasing levels of indistinguishability. Figure 3.7 depicts an ideal scenario where ϵPI takes $\epsilon = 0$ and there is no information leakage; all sequences in universe $\mathbf{R}$ map to a unique supersequence q. However, the Protos may not have enough resources available to bring each sequence r_i to q. Figure 3.8 depicts a slightly less-than-ideal scenario where the universe of sequences splits in $\mathbf{R}_1$ and $\mathbf{R}_2$, mapping to either q_1 or q_2, respectively. Thus the Proto creates two *anonymity sets*, i.e. the Proto leaks whether a sequence r_i belongs to $\mathbf{R}_1$ or $\mathbf{R}_2$, but sequences within each of those sets are indistinguishable. Figure 3.9 depicts a more fragmented input universe, with real sequences in various anonymity sets leading to a subset of supersequences with differing probabilities. Lastly, Fig. 3.10 depicts the more general scenario where each input sequence leads to one or several output supersequences with its own probability distribution. Moreover, input sequences $\mathbf{R} \setminus \mathbf{R}_{\mathrm{DGS}}$ that the DGS did not anticipate lead to new supersequences outside of $\mathbf{Q}_{\mathrm{DGS}}$.

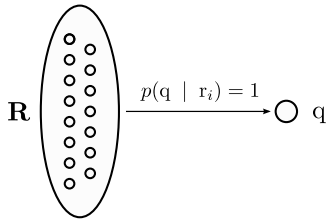


Figure 3.7: Perfect indistinguishability: every real sequence leads to the same supersequence.

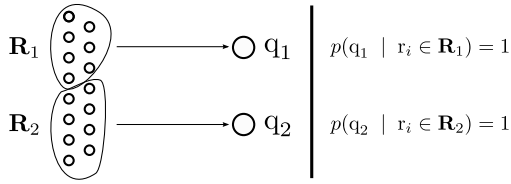


Figure 3.8: Binary split: two supersequences, two anonymity sets.

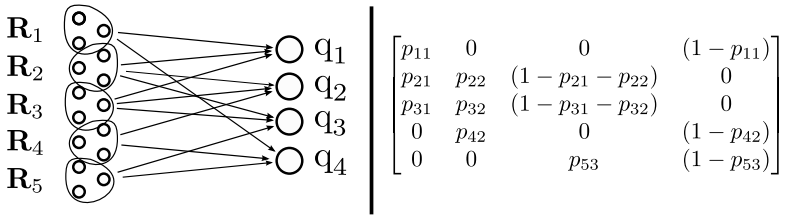
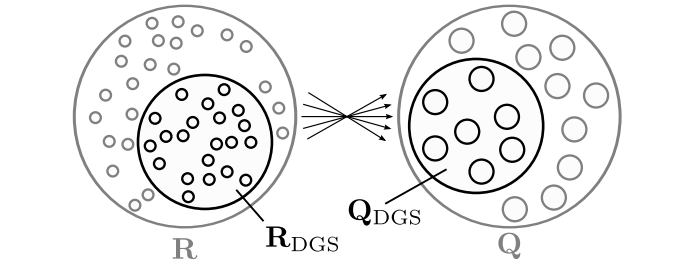


Figure 3.9: Four supersequences with increasingly complex mapping to inputs. The matrix on the right side of the figure represents the conditional probability of each input anonymity set mapping to an output supersequence.



$$\mathbf{R}_{\mathrm{DGS}} \left\{ \overbrace{\begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1n} \\ p_{21} & p_{22} & \cdots & p_{2n} \\ p_{31} & p_{32} & \cdots & p_{3n} \\ \vdots & \vdots & \ddots & \vdots \\ p_{m1} & p_{m2} & \cdots & p_{mn} \\ 0 & 0 & \cdots & p_{(m+1)n} \\ 0 & 0 & \cdots & p_{(m+2)n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & p_{m'n} \end{bmatrix}}^{\mathbf{Q}_{\mathrm{DGS}}} \right. \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \\ p_{(m+1)(n+1)} & p_{(m+1)(n+2)} & \cdots & p_{(m+1)n'} \\ p_{(m+2)(n+1)} & p_{(m+2)(n+2)} & \cdots & p_{(m+2)n'} \\ \vdots & \ddots & \vdots & \vdots \\ p_{m'(n+1)} & p_{m'(n+2)} & \cdots & p_{m'n'} \end{bmatrix}$$

Figure 3.10: A less ideal scenario. Known sequences $\mathrm{r}_i \in \mathbf{R}_{\mathrm{DGS}}, i \in [1, m]$ lead to supersequences in $\mathbf{Q}_{\mathrm{DGS}}$ with differing probabilities. Furthermore, sequences $\mathrm{r}_i, i \in [(m+1), m']$ outside of the universe of known real sequences $\mathbf{R}_{\mathrm{DGS}}$ lead to sequences $\mathrm{q}_j, j \in [(n+1), n']$ outside of the universe of planned supersequences $\mathbf{Q}_{\mathrm{DGS}}$.

We note that the measure of indistinguishability we introduce in Sect. 3.2.1 and its variants capture losses in indistinguishability through various parameters. A DGS that foregoes the protection of rare sequences r_j while protecting the majority of expected sequences r_i may satisfy (ϵ, δ)-local-privacy by ensuring ϵPI for r_i yet with $\delta > 0$ that accounts for rare r_j. Alternatively, a DGS that exploits the similarity of sequences r_i (according to some measure of distance ℓ) to generate supersequences q under resource constraints may choose to satisfy (ϵ, ℓ, δ)-local differential privacy.

The shortest common supersequence problem. Challenges and additional observations in supersequence composition.

Wang et al. examine supersequence composition in the context of designing defences against website fingerprinting and note that optimal solutions require solving two hard problems [545], namely, *anonymity set selection*, which requires choosing which sequences map to which supersequences, and *supersequence*

construction [442], which requires solving the *shortest common supersequence* (SCS) problem, in general NP-hard [305, 545]. Wang et al. thus point to the need to develop appropriate *heuristics* for each particular system and context.

Moreover, additional challenges render strategic supersequence composition unrealisable in practice. As we note above, we are often unable to determine the probability $P(\mathrm{r})$; the universe of real actions $\mathcal{R}$ and the multiple combinations thereof too large to sample or estimate (see e.g. Bindschaedler and Shokri, 2016 [82]). Information leakage is hence unavoidable when a user performs an action r that the DGS has not accounted for. Moreover, users generate actions in real time; the DGS does not know the r that results from user activity until she performs the last action in a sequence, hence the DGS cannot ensure a particular mapping $\mathrm{r} \rightarrow \mathrm{q}$ on the fly as it generates dummies oblivious to the ensuing r. Real sequence batching may however be possible in certain scenarios [423, 546] without forcing unacceptable delays upon users; alternatively, a Proto may include a predictive engine to anticipate real sequences and generate dummies accordingly.[10] However, none of these methods can ensure a predetermined mapping $P(\mathrm{q} \mid \mathrm{r}_i)$ that guarantees a particular level of indistinguishability.

A related point is the set of rules that informs sequence length, i.e. the points at which the DGS considers a sequence starts and ends. In DGS analysis, we may consider the totality of a user's activity —even across periods of inactivity where the user goes offline and comes back online— as one single sequence, then try to determine the probability that such a sequence could have engendered the observed supersequence. In practice however, this is hard to implement for reasons similar to the above. The DGS can only probabilistically predict when a sequence will start or end, let alone when users go offline and come back online to use again the system. Hence, supersequence composition further depends on the criteria we impose on sequence length, which in turn depends on the budget of dummies and the privacy requirements.

As a result, strategic supersequence composition represents an ideal DGS design goal that is hardly ever achievable in practice. Only through *flooding* can one ensure or approximate a predetermined mapping $P(\mathrm{q} \mid \mathrm{r})$. The flooding strategy generates vast amounts of dummy traffic to *"bury"* the real sequence into the resulting supersequence. It relies on the assumption that the resulting SNR is low enough to prevent an adversary to tell apart real from dummy actions. At the extreme, the flooding strategy becomes provably secure through *full padding*, whereby the DGS generates each possible action in the universe of actions $\mathbf{R}$ at a frequency independent from —yet informed by— the user's behaviour, e.g. the heartbeat mechanism in anonymous low-volume

[10] Such a predictive engine would run locally on the client side, without leaking any information about users' behavioural patterns.

communication system *Drac* conceals control communication from adversaries by sending dummies between users at a constant rate and replacing dummies with reals when appropriate [153], while *Loopix* sends dummy messages according to a Poisson process to ensure perfect sender unobservability [435]. Perfect padding ensures perfect indistinguishability through a unique supersequence q, i.e. $\forall i, j : \Pr \mathrm{q} \mid \mathrm{r}_i = P(\mathrm{q} \mid \mathrm{r}_j) = 1$; however, this strategy requires a large budget of dummies when $|\mathbf{R}| >> 0$.

3.3.3 Implementation

In addition to DGS design, a Proto requires an actual implementation as a tool that users install in their devices to run parallel to their own user activity. We briefly review usability and security challenges in Protos implementation.

Usability.

Protos unburden users from the task of obfuscation by automatically generating dummies and relaying real actions on their behalf, yet users need to install and run the Proto in their devices and interact with the Proto to activate and deactivate obfuscation, select the budget of dummies and set any other Proto configuration options available, as well as to determine whether the Proto is providing the protection they seek (e.g. to increase obfuscation resources or discontinue Protos' use altogether; to ensure they are using the tool correctly).

Based on previous work from Shackel, Sasse proposes four usability requirements [467]:

Performance. Designers must ensure that intended users can achieve from the Proto the outcomes they are after.

Learnability. Designers must minimise the amount of learning or practice that users require to use the Proto.

Cost. Designers must minimise user cost (e.g. *physical or mental strain* [467]).

Satisfaction. Designers should promote a pleasurable experience of the system.

Protos' conceptual and technical complexity represents a major usability issue [3]. On the one hand, we wish to free users from the burden of obfuscation, allowing them to afford privacy protection without the need to do anything beyond "using the service (the Proto applies to) as usual". Protos should not degrade UX, e.g. Protos must not require users to manually filter responses to dummy

actions. On the other hand, we wish that users understand the implications of using obfuscation, the limitations of a particular Proto or DGS and the necessary precautions they must take to maximise the benefit they obtain from the tool, e.g. by hiding dummies from users' view, the latter may neither be aware of nor approve of the kinds of dummy actions the Proto generates. Protos must ensure that users understand a DGS' *"side-effects"* and minimise any negative impact on users; they must also communicate the kind of privacy protection they afford, under which conditions, to prevent misleading users into a false sense of security. Protos must also ensure users operate the tool correctly, maximising the benefit they obtain from it. In short, Protos must strike a balance between involving users in the complex obfuscation process or promote a comfortable obliviousness, a common dilemma in security and privacy tools.

Usability is a critical issue in the successful deployment of not only software applications in general, but in security and privacy tools and systems in particular, with security and privacy experts often considering *"people as 'the weakest link' in their efforts to deploy effective security"* [467] as well as acknowledging that security and privacy technologies *"are hard to use"* [240]. Users without the proper security training make mistakes they are unaware of, bypass or disable security features when they prevent them to complete a task or take shortcuts or workarounds to inconvenient security methods [9, 76]. Hence, usability becomes a security requirement in itself; as founders and directors of The Tor Project Roger Dingledine and Nick Matthewson argue, *"if the people who need to use a system can't or won't use it correctly, its ideal security properties are irrelevant"* [175].

Moreover, adoption is a critical factor in services where users interact with each other, as even if users can deploy obfuscation unilaterally with respect to the service provider, they still need to cooperate with each other to achieve indistinguishability (q.v. Chapter 5). Security and privacy scholars have speculated that poor usability underlies the low adoption of security and privacy technology and that if privacy tools were more usable, more people would actually use them [2, 106, 297, 412]. Recent work has however questioned this belief [2, 450], as the main driver for users' adoption of any technology is utility and performance, rather than security [179].

Research on usable security and privacy has advanced in several fronts. A vast body of research has sought to investigate how users understand and conceptualise security and privacy technologies with the aims of both identifying conceptual mismatches that may explain poor security practices and devising communication strategies aligned with users' perceptions [2, 228, 369, 414, 444].

Rader has explored users' perceptions towards data collection by Internet companies such as Facebook or Google, revealing that users who are aware of data aggregation seem to show less concern, while those who know the potential ramifications of data collection and subsequent potential inferences indicate greater concern [441]. Melicher et al. study users' perceptions to online tracking, showing that regardless of their general attitude towards it, users accept tracking in particular websites and contexts depending on factors such as site content or frequency of visits, which current anti-tracking tools are unsuitable to cater to [380]. Yet they also report on users' misconceptions about the information trackers may gather about them and the tracking process, a result that Mathur et al. confirm [369]. Such misconceptions have however little impact on users' attitudes towards tracking, which users perceive as unavoidable, therefore considering any attempts to prevent it futile, explaining in turn why users may mistrust current anti-tracking tools, while other users opine that anti-tracking tools require too much effort [380]. Mathur et al. in fact report that existing anti-tracking tools contribute little to rectify users' misconceptions or dispel their mistrust, a result that Schaub et al. had already previously reported, finding that users believe anti-tracking tools themselves would track them [369, 471].

On the other hand, Abu et al. reveal that despite wide adoption of E2EE, users have misconceptions around its security, thinking e.g. that third parties can still access E2EE messages or that conventional telephony services such as SMS and phone calls are more or as secure as E2EE [1]. De Luca et al. in fact suggest that security and privacy concerns cannot explain the wide adoption of E2EE [162]. Instead, most users choose services with E2EE because of peer influence, thus linking adoption to network effects [25]. Similarly, Rajivan et al.'s study on mobile app privacy risk communication through icons suggests privacy risk has no influence on app selection [443]. Abu et al. thus argue for the need to realign users' perceptions with the actual privacy properties E2EE provides as a way to prevent users' hopelessness and cynicism towards effective protection [1], yet the work of Rajivan finds that priming for privacy has a limited effect on users' final decisions of app selection [443].

Several authors have advocated raising awareness and educating users on security and privacy issues so to enable them to make better informed decisions and develop the necessary skills to protect themselves [223, 222, 292, 323, 469]. Designers thus seek to expose users to and engage them in security and privacy protection processes, encouraging them to learn along the way [10, 215, 458]. Sasse however highlights the difference between teaching users to learn how to use complex security systems so they can use them properly, imposing an unreasonable burden on them, and designing systems *"to make it easy for users*

to do the right thing, with a minimum amount of effort and knowledge" [179, 324, 467, 468, 470].

Research has explored the advantages and disadvantages of exposing users to the technical complexities of E2EE. Ruoti et al. build upon previous research that suggests a strong user preference to avoid dealing with key management, yet find that in their experiments users have greater trust and make less mistakes when using manual encryption as opposed to an integrated solution that performs encryption opaquely, without their involvement [458]. Atwater et al. contest however Ruoti et al.'s result, reporting that users perceive web browser extensions as less secure or trustworthy than standalone applications and that users *"overwhelmingly prefer integrated encryption software* [as opposed to standalone encryption software] *due to the enhanced user experience it provides"* [33].

Designers of security and privacy tools face a dichotomy between visible, transparent implementations that expose users to and involve them in the underlying security and privacy mechanisms, and opaque, integrated implementations that hide the technical complexity from users. Integrating security and privacy in existing systems extends the user base beyond those with enough motivation and awareness to seek a standalone solution to address their security and privacy concerns [162]. Yet even if transparent implementations may raise users' awareness and encourage them to learn, in turn promoting less security mistakes, *"if a supporting task conflicts with a production task, users will attempt to work around it or cut it out altogether"* [467].

Despite the profusion of research on users' perceptions towards and on the usability of security and privacy technologies, to the best of our knowledge no previous work examines users' perceptions towards chaff-based profile obfuscation or Protos's usability; understandably in part as Protos' implementations remain anecdotal. A prominent exception is the work of Howe and Nissenbaum, authors of two Protos, TrackMeNot and AdNauseam, with a considerable user base and therefore available user feedback [294, 295].[11]

Towards Protos' usability. Considering previous research and particularities of Protos, we discuss design decisions towards Protos' usability.

Visibility and filtering of dummy actions. Whereas Protos must ensure that no adversary can filter dummy actions, they must on the other hand filter them from the user to prevent hindering or interfering with her actions on the service, i.e. Protos must be able to capture service provider responses to dummy actions

[11] Whereas downloads do not represent active users, on 6 December 2018 the Mozilla Firefox Add-ons portal reported almost a million downloads of TrackMeNot and almost a quarter of a million downloads of AdNauseam. TrackMeNot averaged a 3/5 rating with 128 reviews and AdNauseam a 4/5 rating with 172 reviews.

and filter them out so that they do not impact UX. Imperceptible filtering however undermines users' awareness of the dummy actions the Proto generates, thus risking nurturing a false sense of security by keeping users oblivious to the effect dummy actions can have on them. Protos may therefore incorporate feedback and awareness mechanisms that provide users with information on dummy activity, e.g. AdNauseam logs dummy clicks on ads in a *vault* that users can consult at any time yet do not have to deal with in any way [295], even if such a logging strategy is limited as it cannot explain the full ramifications of dummy clicks for users' online experience.

Protos can filter dummies with relative ease if the user requires no one but the adversary to observe her real actions, e.g. as in query-response services such as web search or navigation apps; the Proto simply intercepts and filters responses to the dummy queries it sends so the user sees neither dummy queries nor responses. However, whenever users obtain utility from disclosing their real actions to legitimate, non-adversarial parties,[12] e.g. such as in social networking sites, where users publicly post information for other people, Protos must find a mechanism to enable these parties to filter out dummies without exposing additional data to the adversary, essentially requiring these legitimate third parties to share some secret piece of information with the Protos user that enables them to filter the dummies out. In practice, this requires other users to install Protos (or compatible tools that do not obfuscate but can interpret obfuscated data) and devising a signalling scheme that enables legitimate parties to de-obfuscate user data. As we show in Chapter 5, we resort to cryptography to tag dummies for legitimate parties.

Integration. Protos designers may conceive Protos as standalone solutions that users install and use on top of the services on which they require protection or incorporate Protos to an existing system or application. We find prominent examples in Tor and E2EE services of the increased usability integration affords. In the case of Tor, the TBB (a modified version of browser Mozilla Firefox with Tor built in it) represents a clear improvement in usability, as users do not need to install standalone Tor and manually configure it to use their browsers anonymously; instead, they use Tor Browser as they would use any other browser [128, 346, 413].[13] Similarly, in the case of E2EE, as we mention earlier, Atwater et al.'s user study on encryption tools reveals that users prefer encryption integrated on the email client as opposed to having to handle a separate tool. Mathur et al. also recommend the integration of tracking protection in browsers as opposed to requiring users install third party extensions [369].

[12] We note that this differs from the concept of social utility we introduce in Sect. 2.2, which captures the utility users obtain from revealing information to *adversarial* third parties.

[13] *Almost* as they would use any browser. The Tor project warns users of a few habits they need to change when they browse the Internet to use Tor effectively [520].

Integrating Protos as part of services users already use may not only contribute to usability but also adoption. Users who would normally have no interest in a standalone Proto may be willing to use an integrated tool instead if it does not alter their UX. Wide adoption is indeed critical in scenarios that require a collective pool of obfuscating users, as we examine in Chapter 5. Tor is a paradigm of the importance of wide adoption because the greater the number and diversity of Tor users, the more anonymity each user enjoys [175].

Communicating limitations. Previous studies have shown that users often have inaccurate or mistaken mental models of security and privacy risks online; they also misinterpret the protections security and privacy tools afford them, either by assuming that no tool can protect them against certain privacy invasions or the opposite, assuming protection from risks the tool does not even consider. Protos must therefore find strategies to communicate the privacy protection they afford and under which conditions, warning users of any limitations.

Customisation. Ardagna et al. highlight the dichotomy between *usability* and *expressiveness*, the former understood as avoiding complexity and communicating to users in the simplest way, the latter enabling them to tweak the tool and set individual configurations [30].

Dingledine and Matthewson however rail *"against options"*, arguing [175]:

> *"Extra options often delegate security decisions to those least able to understand what they imply[, o]ptions make code harder to audit by increasing the volume of code, by increasing the number of possible configurations* exponentially*, and by guaranteeing that non-default configurations will receive little testing in the field. [D]esigners often end up with a situation where they need to choose between 'insecure' and 'inconvenient' as the default configuration meaning they've already made a mistake in designing their application[, as m]ost users stay with default configurations as long as they work, and only reconfigure their software as necessary to make it usable"*

Dingledine and Matthewson's argument connects with the idea of *sticky defaults* and the philosophy of *privacy by default*, whereby default privacy settings and configurations should offer the most privacy friendly and usable experience to prevent uninformed or misled users to inadvertently degrade the level of privacy protection the Proto affords [553].

Ardagna et al. express a similar view arguing that [30]:

> *"Complex policy specifications, fine-grained configurations and explicit technological details discourage users from fully exploiting the*

provided functionalities. Our goal is then to allow users to express privacy preferences in an intuitive and straightforward way."

Lastly, we must recognise the subversive and playful character of obfuscation. Involving users and exposing them to the obfuscation process, letting them fiddle with the dummies the Proto generates, pretending to generate activity they do not; these activities go against all we know about users' mistaken perception of security and privacy online and may be counterproductive towards securing their protection, yet they can also contribute to raise users' awareness, encourage adoption and improve UX, offering a pleasurable, entertaining incursion into protest and resistance against profiling online [98, 433, 572].

Security against adversarial tampering: preventing side-channel attacks.

Earlier in Sect. 3.1.2 we mention that we assume Protos operate on a secure, trustworthy client in the user's device and consider the security of Protos themselves against adversarial tampering to be an orthogonal problem to profiling.

Moreover, a Protos implementation must ensure that adversaries cannot find a workaround to the DGS by exploiting any vulnerabilities that the DGS does not account for, e.g. differences between the processing time of real and dummy actions or neglected metadata that the DGS is oblivious to.

We acknowledge that regardless of the privacy guarantees a DGS may offer, such guarantees become meaningless if the Protos implementation itself is insecure against tampering that enables an adversary to, e.g. break into the Proto to tag dummy actions and discard them later upon reception.

3.4 Discussion

3.4.1 On Protos' adversary model

In the design and analysis of Protos we assume an honest-but-curious adversary or eavesdropper, i.e. an adversary that, under the condition of indistinguishability between dummies and reals, processes dummies the same way it processes real actions. Moreover, we assume the adversary is strategic, i.e. that it will do anything in its power (even if within the limitations it has as an eavesdropper) to undo obfuscation and break users' privacy protection. Moreover, a strategic adversary knows that the user deploys a Proto (i.e. it

detects the presence of the tool), the Proto's design and can determine or estimate the operation parameters.

We discuss the rationale that motivates this choice and the consequences of using Protos against other types of adversaries.

The naive adversary. A *naive* adversary does not *"attack"* Protos, i.e. it does not attempt to filter dummies, treating all protocol messages, both real and dummy, as real. An adversary may be naive because it is unintentionally unaware of the Proto's deployment or existence, or because it chooses to ignore it. Naive adversaries may have insufficient resources or incentives to detect or attack Protos, e.g. if the cost of attacking a Proto in terms of human capital and computational resources is not worthwhile, or if the Protos' user base is small enough to discard their data at the expense of almost negligible impact on their profiling practices.

Since naive adversaries do not attempt to filter dummies, their *"attack strategy"* involves deterministically producing a filtered profile $\hat{\mathbf{x}} = \mathbf{y}$; there is no notion of uncertainty or probabilistic filtering, naive adversaries take $\mathbf{y}$ as correct, i.e. $\mathbf{y} \equiv \mathbf{x}$. To measure the degree of protection Protos afford against naive adversaries we remove the *"probabilistic component"* from the analysis framework we introduce in Sect. 3.2, thus probabilistic measures become meaningless: it makes no sense to talk of indistinguishability, entropy or information gain. On the other hand, stripping away the probabilistic component from EEE leaves us with $\ell(\mathbf{x}, \hat{\mathbf{x}})$ which, in the case of a naive adversary becomes $\mathbf{E}_{\text{naive}} = \ell(\mathbf{x}, \mathbf{y})$ as $\hat{\mathbf{x}} = \mathbf{y}$. Hence, one measure of protection of *Protos* against naive adversaries is the distance $\ell(\mathbf{x}, \mathbf{y})$, which has further implications for DGS design.

Since naive adversaries do not attempt filtering, a DGS may trivially produce an output q that maximises $\ell(\mathbf{x}, \mathbf{y})$ *with no concern for indistinguishability whatsoever.*[14] However, whereas DGS designs against strategic adversaries implicitly protect from naive adversaries (as maximising $\mathbf{E}$ against a strategic adversary requires generating supersequences r that induce distances $\ell(\hat{\mathbf{x}}, \mathbf{x}) \leq \ell(\mathbf{y}, \mathbf{x})$ as large as possible), the opposite is not true: a DGS that maximises distance $\ell(\mathbf{y}, \mathbf{x})$ with no concern for indistinguishability (i.e. so that $P(\mathbf{x} \mid \mathbf{y}) = 1$) provides no protection against strategic adversaries. Protos' designs that assume naive adversaries thus place users in a vulnerable position, as nothing prevents naive adversaries from becoming strategic and break Protos' security.

[14] In fact, the absence of filtering removes the need for reals and dummies to be indistinguishable in any way.

Moreover, Protos cannot protect individuals who seek to escape *behavioural targeting* against naive adversaries,[15] as this class of adversaries treats every observed action as real, therefore using *all* user activity, both real and dummy, to profile users. Strategic adversaries on the other hand filter obfuscated sequences q to retrieve an approximate profile $\hat{\mathbf{x}} \neq \mathbf{y}$, thereby potentially discarding in the process real actions that users wish to prevent being targeted on. Still, Protos cannot guarantee complete protection against behavioural targeting by strategic adversaries either, as they lack the ability to force adversaries to discard *all* real queries. This lack of control over filtering is in fact a limitation of all chaff-based obfuscation tools, as the obfuscated stream of activity the adversary observes still contains all of a user's real actions.

Adversarial post-profiling decision making. Protos' utility depends on assumptions about adversarial post-profiling decision making, i.e. what we assume adversaries do with partially obfuscated profiles $\hat{\mathbf{x}}$ and its impact on users, highlighting two different but related privacy problems of profiling. Firstly, profiling involves data collection and aggregation through monitoring users' activities and building profiles out of them, revealing information about users' behaviour and identity. Secondly, profiles become instrumental to further adversarial decision making; adversaries build profiles that inform subsequent processes and decisions that may affect the very same users whom the adversary builds profiles on. Protos undermine data collection and aggregation by disrupting the quality and veracity of the data adversaries collect. However, how adversaries ultimately choose to use obfuscated profiles to make decisions is outside the control of Protos.

Our definition of strategic adversary implicitly entails that it attempts to filter from the observed profile $\mathbf{y}$ as much noise as possible, feeding a filtered profile $\hat{\mathbf{x}}$ to whichever processes the real profile $\mathbf{x}$ would be normally fed to. Such processes may or may not have a direct impact on Protos' users, e.g. Protos for web search are likely to have an influence over the adverts users encounter online if such adverts depend on their web search queries, while users may not perceive a direct impact if their queries inform a company's investment strategy. Protos design does not seek to *strategically* correct or tame the algorithmic outcomes that profiling informs, i.e. Protos attempt to thwart or hinder profiling, not manipulate or change its consequences in a specific way.

Other lines of research have examined how to correct or influence algorithmic outcomes such as those that derive from profiling. *Adversarial (machine)*

[15] With *behavioural targeting* we denote practices beyond its meaning in online advertising [378], i.e. any practices informed by an individual's observed behaviour, such as predictive policing or credit scoring [431, 518].

learning studies how adversaries may bias algorithmic outcomes by strategically polluting a machine learning algorithm's input data (i.e. injecting noise), either during training or testing [108, 357]. Adversarial learning considers those running the algorithms to be honest and legitimate and those polluting the inputs to the algorithms adversarial. However, we may reverse the roles to defend users put at a disadvantage by algorithmic decision making by careless or malicious entities running the algorithms [261], which is the idea underlying Gürses et al.'s protective optimisation technologies (POTs), tools that rely on adversarial learning techniques to protect populations and environments from the negative outcomes that *optimisation systems* cause [55, 261]. Protos thus differ from adversarial learning in that the former seek to undermine or prevent profiling, rather than manipulating it; they are oblivious to the impact obfuscated profiles have on users, the underlying assumption that users are either indifferent to the side effects of profiling or willing to bear those effects as a way of protest [100]. Otherwise, Protos do not provide an adequate solution to their privacy problems.

Protos may however obfuscate profiles to such an extent that they become useless to the adversary who, unable to effectively filter them, refrains from using them, abandoning further processing too. This is in fact Protos' ideal and optimal outcome, to thwart profiling and, as a result, any further processing. Protos' ability to prevent profiling thus depends on the adversaries' ability to assume the cost of polluting their databases with obfuscated profiles, i.e. whenever adversaries require very accurate profiling data, Protos have the potential to thwart profiling.

On Protos (un)detectability. Profilers unaware of Protos' adoption among their userbase effectively become naive adversaries, processing obfuscated profiles with potentially unexpected and undesirable side effects for Proto's users. To force naive adversaries to become strategic and (hopefully) discourage profiling, Protos designers have incentives to signal Protos' deployment to profilers, warning them about the presence of dummies in the stream of data they collect. Alerting adversaries of Protos deployment highlights the difference with a steganographic hiding or mimicry-based defence strategy: Protos do not seek to deceive or mislead the adversary into believing a user's profile is $\mathbf{y}$ instead of $\mathbf{x}$, as the consequences for the user of further processing $\mathbf{y}$ may be as detrimental or worse than those of processing $\mathbf{x}$.

Therefore, Protos require *tool detectability* to alert adversaries of the presence of obfuscation and encourage them to discard obfuscated profiles. There are however exceptions to the detectability requirement.

In their analysis of AdNauseam, a Protos against tracking and profiling by online advertisers, Howe and Nissenbaum discuss its potential to provide *social privacy*, i.e. to protect not only AdNauseam users from profiling but *non-users* too [295]. AdNauseam pollutes the data trackers collect on users' clicks on adverts; if trackers combine the ad-clicking behavioural information from both AdNauseam users and non-users, they pollute the behavioural models that govern the optimisation process that determines which adverts to serve to users [304, 369, 541, 568]. Howe and Nissenbaum posit that if such behavioural models are no longer reliable to target advertising to users, they lose their value, which in turn desincentivises profiling, illustrating how AdNauseam users contribute to the discontinuation of profiling for everyone [295].

For AdNauseam to pollute profilers' behavioural models, profilers need to incorporate AdNauseam users' obfuscated profiles to their behavioural models. Profilers may however refrain from doing so if they are unable to filter dummy clicks from AdNauseam users' clicking behaviour data, discarding it instead to keep a behavioural model built on non-obfuscated profiles alone.[16] Thus the potential of AdNauseam or any other Protos for social privacy is lost if profilers detect the presence of AdNauseam and discard user clicking data,[17] even if from the point of view of individual privacy this is an ideal outcome, as trackers no longer profile the AdNauseam user. To pollute the behavioural models by forcing profilers to combine the profiles of those who use Protos and those who do not, Protos must therefore seek *tool undetectability*, preventing trackers to sift AdNauseam users out from their databases.

Tool undetectability brings about a number of dilemmas and trade-offs. Firstly, it restricts the number of dummies Protos are able to generate. A Proto that generates dummies at a rate well beyond a human's increases the chances that the profiler detects it. Such a restriction further impacts a Proto's ability to provide profile confidentiality, as the budget of dummies is limited to plausible rates of action generation which may be insufficient to cover up users' real

[16] In fact, *click fraud* is a massive problem to the online advertising business. The Association of National Advertisers (ANA) reports that in 2017 *"9 per cent of desktop display ad spending and 22 per cent of desktop video ad spending is lost to fraud"*, bringing fraud losses at $6.5bn [127, 251], whereas WPP, a British multinational advertising company, estimates more than $15bn in losses [226, 446]. Hence, advertisers have strong incentives to tackle click fraud and block fake clicks. In particular, besides banning AdNauseam itself, Google has actively sought to mitigate click fraud [295, 347, 408].

[17] *Lost* to the extent that adversaries can discard detectable Protos' users profile data and keep profiling non-users as usual, meaning that non-users do not benefit from the privacy protection that being a potential Proto user confers them, i.e. adversaries know that their profiles have not been obfuscated. Protos however contribute to social privacy in other ways, e.g. by enabling Protos' users to voice their discontent thus pushing for more privacy-friendly practices which then become available to everyone and by refusing to be part of the aggregated models profiles build, undermining their richness and value.

activities. Hence, escaping detectability may involve relinquishing profile confidentiality. Secondly, ensuring tool undetectability becomes analogous to deploying a collective, distributed Proto that instead of obfuscating individual users' profiles, seeks to obfuscate the collective profile of the user population. Hence, undetectability requires indistinguishability between Protos' users and non-users, in addition to indistinguishability between real and dummy actions, resulting in two levels of obfuscation and indistinguishability: at the action level and the user level. Protos obfuscate individual users' profiles; undetectable Protos obfuscate profiles of users' profiles. Lastly, tool undetectability seeks to ensure adversaries cannot distinguish between Protos users and non-users, thus processing the former's observed profiles $\mathbf{y}$ as real and triggering in turn potentially negative side effects deriving from obfuscated profiles. As we have noted above, POTs can more adequately respond to this problem by strategically polluting the inputs to guarantee both tool undetectability or *stealth* and prevent negative side effects for obfuscating users.

The honest-but-curious assumption. Protos assume an HbC adversary, namely, an adversary that does not interfere with the quality of service Protos' users receive. Protos require that this assumption holds for their successful deployment, especially since we consider service providers as reference Protos adversaries, thus conferring them the ability to provide or withhold service at their will.

Contrary to HbC adversaries, *active* adversaries disrupt the protocol that provides utility to users to undermine indistinguishability between reals and dummies. An *active* adversary may strategically discard user actions or provide responses that do not match the action request, forcing users to repeat their commands and slowing them down, eroding QoS. Active adversaries may also subject users to antibot tests that a Proto would be unable to respond to, e.g. forcing users to solve captchas [534]. A Proto would need to adapt to such an adversary, attempting to minimise QoS damage and responding to the adversary's attempts at eroding the indistinguishability between reals and dummies. Worse still, adversaries may simply ban users from the service if they detect the use of obfuscation, e.g. by forbidding obfuscation in their terms of service and legitimising that decision on the additional burden dummies impose on the system. Adversaries may in fact refuse to bear the cost of processing dummies. Protos offer no recourse against banning.

For the honest-but-curious assumption to hold, the cost for the adversary of actively attacking Protos must be greater than permitting them. In terms of human labour and computing power, active adversaries unlikely require more resources than HbC adversaries require for eavesdropping and filtering. Banning

comes at negligible cost, as it only requires Protos detection. A more probable cause for adversaries to be HbC relates to their image and public relations as well as the legal framework that regulates their operations. Banning users or actively sabotaging Protos portrays service providers as privacy-invasive and unwilling to acknowledge or concede their users' discontent at their profiling practices. Banning Protos' users risks further alienating them and encouraging them to change provider altogether, thereby further eroding their trust and losing them to competitors. Moreover, regulatory policy to protect Protos users may dissuade adversaries against active attacks. Justifying the legitimacy of Protos, explaining their role as a mechanism of expression and protest, contributes to forge a favourable public opinion and perception that pressures adversaries to relent and lawmakers to intervene. Brunton and Nissenbaum's work on the ethics of obfuscation provides an analysis of the factors that legitimise the deployment of Protos [98, 100].

Protection, expression and subversion. Howe recognises three main aims of obfuscation tools: protection, expression and subversion [293]. Through the paragraphs above we have implicitly laid out the conditions and assumptions that intervene in the attainment of each of those goals. We have focused our analysis of *protection* in terms of both *profile confidentiality* and the negative effects of subsequent decision making. A Proto provides profile confidentiality if it generates enough *indistinguishable* dummy traffic to prevent a *strategic* adversary from retrieving the user's profile $\mathbf{x}$. Protos' ability to protect against subsequent processing is largely out of the designer's control: it is up to the adversary to decide whether or not to feed polluted profiles $\hat{\mathbf{x}}$ to the algorithmic machinery on which subsequent outcomes depend.

We have conceptualised *expression* as *tool detectability*. Undetectable tools preclude user expression by remaining invisible to the adversary. Detectable tools on the other hand make themselves visible to the adversary, thereby prompting an attack or response.

Lastly, we analyse the trade-off between subversion and the two previous goals. Simply by obfuscating profiles, Protos disrupt the data collection process. The extent to which they do so depends however on a set of assumptions. If a Proto is undetectable, a profiler incorporates a user's obfuscated profile $\mathbf{y}$ as is and succeeds at polluting the profiler's database. However, we recall that undetectability comes at the expense of expression and possibly requires a limit on the amount of dummies a Proto can generate to avoid detectability, thereby undermining protection. Moreover, undetectability means that a profiler processes a user's obfuscated profile $\mathbf{y}$ as is, therefore a user must be ready to bear the burden of further processing on $\mathbf{y}$. Hence, under the assumption of

undetectability, Protos realise subversion at the expense of diminished protection and expression.

On the other hand, if a Proto is detectable, a profiler may choose to either attempt to filter and process a user's obfuscated profile $\mathbf{y}$, discard it or ban the user altogether. Let us assume that a Proto's subversive impact is greater when it incorporates a user's obfuscated profile $\mathbf{y}$ than if it discards the user's data (with banning the user altogether having an equivalent impact to discarding data).

Detectability imposes no upper limits on the amount of dummies a Proto can generate; therefore, omitting any other constraints on the budget of dummies, such as cost, a detectable Proto can attempt to maximise a user's profile confidentiality. Let us further assume that the probability that an adversary incorporates an obfuscated profile to its database instead of discarding it depends on its ability to filter it, i.e. a profiler that can obtain an accurate approximation of the original profile is more likely to incorporate it than if it is unable to remove any noise. Under this assumption, increasing levels of profile confidentiality offer more privacy protection but prompt adversaries to discard obfuscated profiles, whereas low levels of profile confidentiality offer less privacy protection but encourage adversaries to incorporate them to their profiling database. Hence, we observe an unproductive relationship between protection and subversion. As we increase protection, we limit a Proto's potential for subversion by encouraging profilers to discard user data; however, decreasing protection further undermines a Proto's potential for subversion by limiting the amount of noise an obfuscated profile introduces in the profiler's database.

The profiling function. The general profiling model we provide in Sect. 3.1 implicitly assumes that the profiling function g takes as input an individual user's activity data r, rather the data of all users of the service. In practice, profilers cluster and separate users in categories informed by the data of the whole user population. Moreover, changes in input data and adversarial goals over time further imply *dynamic* profiling, i.e. whereas we assume the profiling function r to be stable, profiling changes are bound to change over time, even *optimising* outcomes in real-time. In this thesis we abstract away from such complexity, yet acknowledge that, similarly to ongoing work on POTs, a more realistic, practical model of profiling should account for the dynamic nature of profiling [261]

Trust. Security and privacy engineering mandates designers should minimise the trust assumptions they place on components and entities systems require to work, as failing those assumptions the system faces additional vulnerabilities designers did not account for.

The Protos' model relies on the assumption that the environment on which Protos run is trustworthy (i.e. free from adversarial interference and tampering) by virtue of deploying adequate security measures which are out of the scope of this thesis (q.v. Sect. 3.1.2 and 3.3.3).

Recent trends have shown tech giants such as Apple and Google moving towards ever greater vertical integration, producing everything their customers need to use their services and products, from the devices and hardware within to the operating system, software applications and platforms that provide online services. Narayanan has pointed that this results in hardware and software *"packed together in a way that users can't fully control or modify. [...] Combined with the fact that today's software typically updates automatically, not trusting vendors isn't an option anymore"* [400]. When users rely on the very adversary that profiles them to supply them with the software and hardware on which Protos run, the assumption that the *client side* is trustworthy becomes meaningless. Instead, we need to rely on the HbC assumption (namely, that the adversary permits the deployment of Protos because of market or regulatory forces) while deploying mechanisms to detect adversarial tampering and interference.

Lastly, users may not trust Protos developers themselves, lacking the means or expertise to validate Protos' privacy claims. Indeed previous research has shown that users perceive third-party extensions and applications to be untrustworthy [33, 471].

The security and privacy research community has often promoted open-source designs as a way to increase security and reliability, owing to, as Fuggetta notes, two main factors [221]. First, anyone can examine and evaluate open-source designs, enabling public scrutiny and validation that in turn enables finding design flaws or malicious code; whereas it is unlikely and unreasonable to expect non-experts users to examine Protos's code [467], other privacy engineers or researchers may be willing to do [90, 535]. Secondly, open-source also enables developers to fix the design flaws they or others encounter. However, Schryen refutes these arguments echoing Levy's *"Sure, the source code is available. But is anyone reading it?"* and noting that *"in the Open-BSD source, foundational vulnerabilities have a median lifetime of at least 2.6 years"*; he concludes that there is no difference in terms of *"vulnerability disclosure and vendors' patching behavior"* between open and closed-source software [476].

Whereas we advocate for open-source Protos, given non-expert users' lack of knowledge in computer security and privacy, open-source scrutiny is unlikely to have an effect on their perception. Therefore, designers must devise alternative communication methods to tackle users' concerns or, as we discuss earlier in Sect. 3.3.3, opt for smooth, transparent integration into users' workflow.

3.4.2 On Protos' cost

In addition to the role of service providers as gatekeepers for Protos' users, Protos' viability depends on the cost of generating dummies. For Protos to be viable, the cost of generating dummies must be negligible, as if the cost is too high, any level of privacy that requires generating a significant amount of dummies becomes too expensive, e.g. whereas users may obfuscate searches for products on Amazon by using a Proto that automatically generates dummy searches, obfuscating their *purchasing profile* becomes prohibitively expensive, requiring users to make dummy purchases of goods —not to mention the handling of deliveries.

3.5 Conclusion

In this chapter we have proposed an abstract model for chaff-based profile obfuscation tools (Protos). We have formalised profiling as a privacy threat for users of online services, introduced Protos as an abstract solution that relies on chaff to thwart profiling and assumed an honest-but-curious service provider as the reference adversary that Protos protect against.

To assist Protos' design and evaluation, we have further introduced a set of measures for the analysis and evaluation of Protos, distinguishing between mechanism-centred analysis (MCA) measures, such as indistinguishability and mutual information, and attack-centred analysis (ACA) measures, such as information gain and expected estimation error.

MCA measures focus on the analysis of the relationship between a Proto's inputs and outputs alone; they capture the amount of information Protos leak, regardless of whether and how an adversary leverages that information. They also represent general privacy measures that do not focus on particular adversaries or scenarios, abstracting and, under certain conditions, being independent from adversary knowledge, which makes them specially suitable for DGS design. ACA measures on the other hand focus on particular adversaries and attack strategies; they capture the amount of information an adversary obtains, how close an adversary gets to its goal —with its corresponding effect on user privacy. ACA are thus particularly useful to assess the impact that particular adversaries may have on users' privacy, e.g. as part of an audit or privacy impact assessment. Hence, both types of measures play a role in Protos' design and evaluation.

Furthermore, we have examined key aspects in Protos' design, distinguishing three phases in the design process: first, privacy requirements elicitation,

adversary modelling and privacy properties operationalisation. We illustrate how to address this phase of the process through the use cases we study in chapters 4 and 5. Secondly, DGS design. We have highlighted the challenge of determining the *pool of dummies*, namely, the type and format of the dummy actions a Proto generates, in particular when the universe of real actions is unknown or ever-expanding. We have conceptualised the sequence that results from combining real and dummy actions as a *supersequence* and illustrated the importance of ensuring that several real sequences map to the same supersequence to ensure *indistinguishability*. Moreover, we have examined the challenges in supersequence composition when a limited *budget of dummies* limits the number of real sequences that map to the same supersequence. Thirdly, Protos' implementation. We have argued that usability represents a paramount design issue, as on it depends that users can reap the benefits a Proto offers by using the tool correctly and at no detriment to their user experience. We have reviewed previous work on security and privacy tools' usability, extrapolating their findings to Protos design. Protos should require as less user effort and technical expertise as possible; previous studies indicate a strong user preference towards integrated privacy solutions that do not interfere with the way users are accustomed to using a service. Among other consequences, this entails that Protos should minimise dummies' impact on user experience and develop effective communication strategies.

Lastly, we have discussed the implications that assuming an honest-but-curious (HbC) adversary have on Protos' deployment when in practice adversaries may behave otherwise, focusing on two main types of adversaries: naive, who do not attack Protos, and active, who threaten denial of service. Protos do not protect against the effects that derive from profiling based on dummy actions; likewise, Protos cannot protect against an active adversaries that disrupt quality of service for Protos' users.

In this chapter we have introduced an abstract Protos and provided general analysis and design principles. To illustrate how the general Protos model instantiates in practice as well as how the analysis and design principles apply in concrete scenarios, we devote the next two chapters, Chapter 4 and Chapter 5, to examine the analysis and design of Protos in two use cases: web search and online communication services.

Two use cases.

In Chapter 4 we instantiate the general Protos analysis framework to evaluate chaff-based private web search (CBPWS) tools, this is, Protos for web search that seek to conceal users' search interests from the search engine provider. CBPWS

illustrates how users may unilaterally deploy Protos against uncooperative service providers to avert web search profiling. CBPWS further highlights the complexity of achieving indistinguishability between real actions and dummy actions when the size of the universe of real actions' tends to infinity and the random variable over real actions is hard to model.

In Chapter 5 we instantiate the general Protos analysis framework to evaluate communication profile confidentiality (CPC) tools, namely, Protos that seek to conceal users' communication patterns from the provider of an online communication service. CPC tools illustrate how even if users can unilaterally deploy Protos against adversarial, uncooperative service providers, in certain contexts they require cooperation from other users to achieve any meaningful level of protection. CPC tools further illustrate how encryption enables content indistinguishability, thereby significantly easing the task of generating indistinguishable dummy actions. At the same time, enabling content indistinguishability exposes and highlights the importance of metadata in Protos' design. We resort to CPC tools to examine the metadata selection process, namely, which metadata a DGS must consider to ensure indistinguishability between real and dummy actions. We show how to leverage mutual information as an information leakage measure to assist the metadata selection process and, in particular, how to simplify mutual information computation to speed up preliminary design assessments. We further examine, in the context of online communication, the role that stakeholders other than designers and privacy engineers may play in the successful deployment of Protos. We analyse the role service providers may play in the deployment of encryption in social networking sites (SNSs) —that ensures content indistinguishability for CPC tools— and the role users' perceptions towards third-party E2EE tools (TPETs) may play in the uptake of CPC tools.

Chapter 4

Private Web Search

Once we searched Google, but now Google searches us.
—Shoshana Zuboff.

To think strategically one has to imagine oneself in the enemy's place. [...] Misinterpreting an enemy can lead, in the long run, to defeat —one's own. This is how sometimes empires fall.
—John Berger, *Hold everything dear.*

Web search has become an indispensable online service, the *"primary means by which individuals access Internet content"* [200], allowing Internet users to find the information, services and resources they seek online. In web search, users compose *search queries* comprised of one or several *keywords* that capture what users are looking for. Users send their queries to a *search engine* or *search provider*, which in turn searches on its databases and returns a list of potentially relevant candidate sites to the user.

In currently dominant web search service architectures like that of Google, Bing or Baidu, search providers collect all users' search queries [200]. Search query collection enables user profiling to better target users with advertising, thereby supporting service subsidisation, i.e. requiring no payment fee from users; it also enables service improvement and personalisation. However, web search profiling poses several privacy risks, enabling providers to determine, among other sensitive or personal traits, users' geographical location, education level, occupation, sexual orientation and health status [302, 307, 514]. The infamous

AOL scandal has highlighted the privacy risks of collecting search data, showing not only how easy it is for third parties to re-identify web search users from supposedly *"anonymised"* data, but also how sensitive web search data is [48]. Moreover, beyond the privacy risks of inferring and disclosing users' personal details, web search profiling raises additional questions relating to the algorithmic processes that feed on users' profiles, posing risks of algorithmic isolation, discrimination and manipulation [201, 383, 407].

In light of these threats, users have several options to protect their privacy online. Online privacy and consumer protection advocates propose a series of actions that users can undertake to protect themselves, such as avoiding signing up to personal web search accounts, regularly deleting their cookies, distributing search queries across various search engines or simply *watching what they search for* [178, 486]. These actions however increase the burden on web search users and disrupt their user experience, even if there are tool designs to automate and assist users in some of these tasks, e.g. by partitioning user queries across interest categories and sending those belonging to different categories to different search engines, using different cookies [310]. Still, such rudimentary solutions have limited efficacy, e.g. search providers have the means to cookieless tracking [409].

Alternatively, users can switch to more privacy-friendly web search engines that promise not to log their queries, such as Startpage[1] or DuckDuckGo.[2] However, users may still mistrust these search providers or be unwilling to disclose their search queries, as nothing other than good faith prevents providers from profiling. Besides, malicious insiders or eavesdroppers can still gain access to users' queries, the latter potentially able to do so even when users send their queries over a TLS-encrypted connection [415].

Users may also connect to the search engine through an anonymous web browsing system such as Tor, that makes them appear as someone different in each session [74, 176, 449, 461]. Anonymous web browsing hinders the creation of search profiles through session unlinkability, as search engines cannot ideally link users' search queries across different sessions. Using anonymous web browsing however comes at the cost of a slower user experience that users may be unwilling to tolerate [227, 228]. Moreover, eavesdroppers may still rely on query fingerprinting to re-identify some user queries [415], while one-hop anonymisers represent a single point of failure that requires users trust the anonymising proxy, offering little or no advantage to a good-faith, privacy-friendly search engine [79].

Other researchers have proposed the use of collaborative relaying systems, whereby a pool of users relays queries between each other before sending them

[1] Previously known as *Ixquick*, available at `https://www.startpage.com/`.
[2] Available at `https://duckduckgo.com/`

to the search engine. Hence, the latter cannot ideally determine who among the pool of users is the originator of each query [109, 203, 313, 532], yet these proposals open additional privacy problems as web search users gain visibility over other users' queries, e.g. Erola et al. and Viejo et al. propose to use an online social network's friends as relay nodes, exposing users' queries to their friends, even if relaying operations seek to hide the originator [203, 532]. Some proposals rely on cryptography to broadcast queries' search results to the whole group so that each user retrieves its search query result *anonymously*, yet do not consider the threat of de-anonymisation from exposed search results [109, 313], an issue that Lindell and Waisbard's proposal addresses, albeit still leaking search results to the entity that coordinates the users' pool [355].

Private information retrieval (PIR) on the other hand offers provably-secure solutions to conceal search queries from the search engine, enabling a user to retrieve records from a database without the database owner determining which records she accesses [123, 337]. While these cryptography-based solutions provide strong privacy guarantees, their complexity is linear on the database size and therefore remains prohibitively expensive for certain applications that involve very large databases, such as web search [64, 15]. Proposals that weaken the security guarantees in exchange for greater efficiency do exist [170, 519], however, search engines have little incentive to implement costly protocols they cannot profit from.

Obfuscation tools on the other hand enable users to unilaterally protect their privacy, without the need to rely on other, potentially malicious users, slow anonymous communication systems, or the cooperation of web search providers. On the one hand, utility-degrading solutions such as Masood et al.'s propose to replace user queries with alternative queries that pose a lower privacy threat at the cost of some utility penalty [368]. On the other hand, utility-preserving solutions rely on chaff or *dummy queries* to obfuscate the search profile the search engine retrieves, enabling the concealment of a user's actual search queries and interests. Toledo et al. argue that chaff-based private web search (CBPWS) tools represent a relaxation of PIR; instead of retrieving all possible records in the database with each user request, they retrieve a subset of records, thereby leaking which records the user does not access. Still, one major advantage of CBPWS tools over PIR is that users do not require the cooperation of the search engine to deploy them.

Besides protecting individual users against profiling, CBPWS diminishes the utility of search profiles to search engines and, assuming that a sufficiently large user base adopts CBPWS tools, further reduce the economic incentives to perform profiling. CBPWS further provides an advantage over anonymity systems in that both types of solutions prevent individual profiling, yet only

the former pollutes the logs of queries across the userbase, therefore exhibiting greater subversion potential.

Due to the advantages of CBPWS over other solutions, several researchers have proposed designs and implementations of CBPWS [182, 198, 294, 397, 448]. In this chapter we instantiate the general model and analysis framework of chaff-based profile obfuscation (CPO) we have introduced in Chapter 3 to evaluate CBPWS tools. Then, we evaluate several CBPWS proposals, uncovering systematic vulnerabilities in their designs and flaws in their original evaluation.

4.1 Modelling chaff-based private web search

4.1.1 System model

We consider an online web search provider, i.e. an entity that provides an online web search service. The search provider *indexes* websites by *keywords* related to their content. Web search users compose *queries* comprised of one or more keywords related to the topics they are interested in and send them to the search provider. The provider compiles a list of web pages indexed by the query keywords (according to some selection and ranking algorithm that we abstract away from) and returns it to the user.

Figure 4.1 depicts the web search system model. We denote individual queries as r, taken from a universe of queries $\mathcal{R}$. For each query, the search engine returns a set of web search results o.

We assume user devices are secure, namely, free from malware that may monitor and leak information about their searches, i.e. user devices are *trustworthy*.

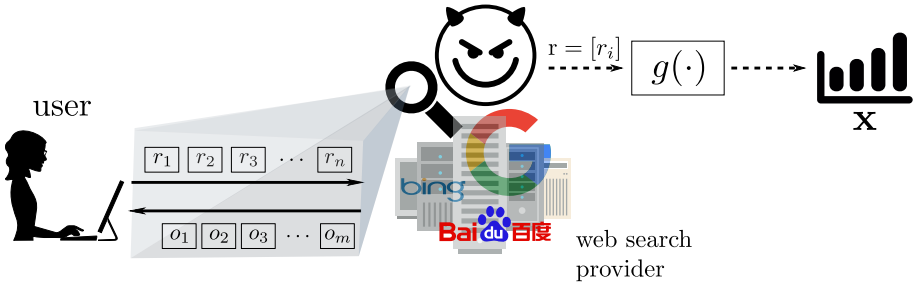


Figure 4.1: Web search system and threat models.

4.1.2 Threat model

The search provider collects and processes each user's sequence of search queries $\mathrm{r} = [r_1, r_2, \ldots, r_n]$ into a profile, $\mathbf{x}$. We model a search profile as a multinomial distribution $\mathbf{x} = \{x_i\}$, where each element x_i represents a *probability* the providers' profiling function assigns to a category i. The choice of categories i and the meaning or interpretation of probability x_i depend on the profiling function $g(\mathrm{r})$. We do not make any assumptions about $g(\cdot)$, i.e. it may map queries to topics according to the keywords the queries contain and other contextual information such as previous queries or search results Alice clicks on. Still, we consider that g relies on two subfunctions or components. First, a *semantic classification algorithm* SCA that maps queries to categories [59, 303]. Second, a *modulator* MOD that weighs the contribution of each query or sequence of queries in the profile according to the profiling strategy chosen by the adversary.

As an example, an SCA may attribute the query {*red wine*} to categories [*alcoholic beverages*] and [*health*], assigning a different weight to each category. Categories may range from very broad (e.g. health, sports, music) to very specific (e.g. each keyword meaning representing its own category). The modulator MOD on the other hand assigns e.g. greater weight to recent queries (in the last month), discounting the weight of older queries, or adjusts the weight of queries as a response to surges in topic popularity.

The profiling function g may more generally consider Alice's search engine usage patterns, e.g. the time of the day Alice uses the search engine or the volume of queries at different times of the day. However, for simplicity and the sake of illustration, we consider in the remainder of this chapter that the profiling function g constructs each profile $\mathbf{x}$ by normalising the weights that an SCA assigns to each query, namely:

$$\begin{aligned} x_i &= \sum_j \mathrm{SCA}(r_j)[i] \\ \mathbf{x} &= \left[\frac{x_i}{\sum_i x_i}\right] \end{aligned} \tag{4.1}$$

One interpretation of this profiling strategy is that x_i represents the magnitude of interest in topic i the provider assumes a user has.

4.1.3 Chaff-based private web search tools

A chaff-based private web search (CBPWS) tool is a Proto that generates dummy search queries and clicks on search results on behalf of the user to prevent the adversary from retrieving the user's search profile $\mathbf{x}$.

We denote dummy search activity as d and, unless we state otherwise, we focus on the generation of dummy search queries, abstracting away from the analysis of clicks on results and any further activity on web search, which we discuss in Sect. 4.4.4.

We conceptualise CBPWS tools through the following elements: the *privacy property*, the *privacy measure* and the *dummy generation strategy* (*DGS*).

Privacy property. CBPWS tools generally aim to prevent the retrieval of users' search profiles $\mathbf{x}$. However, such a goal is open to interpretation, as it does not specify the information a profile contains. The privacy property must therefore point to a more specific, narrower definition of profile and level of protection.

Privacy measure. The privacy measure quantifies the privacy property the tool is after. Whereas privacy properties often refer to abstract notions of protection open to interpretation, privacy measures unambiguously formalise and *quantify* privacy. Hence, privacy measures must capture and accord with the privacy property the tool intends to provide. Moreover, the privacy measure is the yardstick by which designers choose to evaluate the CBPWS tool and set the desirable privacy level or bounds at which the tool should operate to be effective. We have provided examples of privacy measures that CBPWS designers may rely on in Sect. 3.2; we show how to instantiate and adapt these measures for private web search in Sect. 4.2.

Dummy generation strategy (DGS). The DGS governs how the CBPWS tool generates dummy activity, i.e. the number of dummy queries to generate, their content and semantics, their distribution amongst categories, their sending time and any other metadata and relevant features, as well as visits to search results.

Similarly to the profiling function g the adversary uses, the CBPWS tool requires a profiling function $g_{\mathcal{T}}$ for the DGS to generate dummy activity. The tool's $g_{\mathcal{T}}$ may or may not match the adversary's g, depending on e.g. whether information about the adversary's profiling strategy is available or whether the tool chooses to defend against one specific profiling algorithm. Still, the tool's $g_{\mathcal{T}}$ relies on internal modules analogous to the

adversarial g, like modulators and semantic classification algorithms. For simplicity, we generally consider throughout this chapter that $g_{\mathcal{T}} = g$. We however discuss the implications of relying on a different $g_{\mathcal{T}} \neq g$ in Sect. 4.4.3.

Furthermore, we specifically consider the following dummy generation strategy (DGS) parameters:

The dummy rate. We define the budget of dummies as a dummy rate ρ which represents the proportion of dummy actions to total actions:

$$\rho = \frac{|\mathrm{d}|}{|\mathrm{d}| + |\mathrm{r}|}$$

where $|\mathrm{d}|$ and $|\mathrm{r}|$ represent the cardinality of the sequence of dummy and real actions, respectively.

We may define different rates for dummy queries ρ_{query} and dummy clicks on search results ρ_{click}. We recall however that in this chapter we focus on the generation of dummy queries so unless we state otherwise ρ refers to the ratio of dummy queries to total queries alone.

The dummy rate can alternatively be defined in terms of data volume (e.g. MB) to better account for bandwidth constraints.

Dummy, target and observed profiles. The *dummy profile* $\mathbf{w}$ is the outcome of applying $g_{\mathcal{T}}$ to the sequence of dummy queries d the DGS generates. The *observed profile* $\mathbf{y}$ is the result of applying $g_{\mathcal{T}}$ to the combined sequence of real and dummy queries $\mathrm{q} = \mathrm{r} * \mathrm{d}$. The *target profile* $\mathbf{y}^t$ represents the profile the DGS aims to build out of both real and dummy activity, so that the observed profile $\mathbf{y} \to \mathbf{y}^t$. Figure 4.2 represents the processing of real and dummy activity into profiles $\mathbf{x}$, $\mathbf{w}$ and $\mathbf{y}$.

Lastly, as any CPO tool, CBPWS tools must include a filter component to prevent dummy activity from impacting user experience, as well as a user interface to communicate with the user. Fig. 4.3 provides a modular depiction of a CBPWS tool.

4.1.4 Adversary model

Once users deploy CBPWS tools, the adversary can no longer profile them as usual, as processing both real and dummy search actions results in an *observed profile* $\mathbf{y}$ that may bear no relation to the *real profile* $\mathbf{x}$.

We instantiate the generic adversary model in Sect. 3.1.3 to model how the adversarial profiler responds to CBPWS deployment.

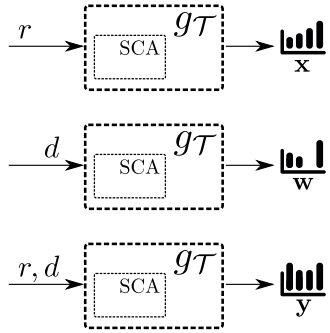


Figure 4.2: A CBPWS tool's $g_\mathcal{T}$ processes real activity into real profile $\mathbf{x}$ and dummy activity into dummy profile $\mathbf{w}$. Processing both types of activity results in observed profile $\mathbf{y}$.

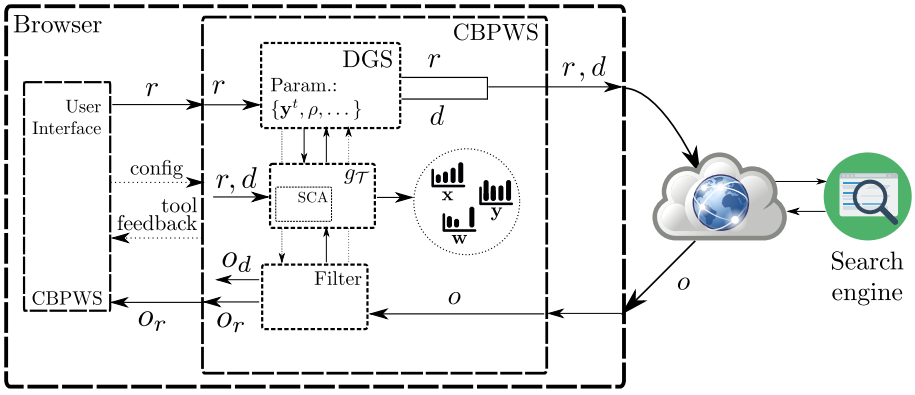


Figure 4.3: Abstract modular depiction of a CBPWS tool.

Goals. The adversary aims to recover the real search profile, $\mathbf{x}$, with $\mathbf{x}$ the output of the adversarial profiling algorithm, $\mathbf{x} = g(\mathrm{r})$, that takes as input a user's sequence of search activities r.

Capabilities. The adversary is able to observe and log all user search activity, both queries and clicks on search results. In this sense, the adversary might be the search engine itself or any other entity able to intercept users' queries to and responses from the search engine. However, since the use of a secure channel between web search users and the provider limits the attack capabilities of external observers, we focus on adversarial search providers. Moreover, because we assume user devices to be trustworthy, it follows that the adversary cannot

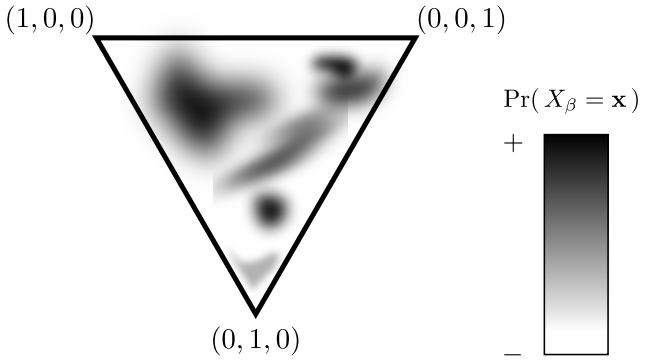


Figure 4.4: $P(\, X = \mathbf{x}\,)$ over profile space.

break into and manipulate them to reveal which queries are real and which are dummy.

The search engine provider further possesses *background knowledge* on users' profiles (e.g. a history of previous search queries) and other *auxiliary information* such as trending topics on the Internet or datasets from other search engines. We note that this implicitly assumes that there is a sufficiently large sample of non-obfuscated search profiles available, i.e. many individuals do not attempt to conceal their search interests.

We denote adversarial prior knowledge through the random variable X_β, with $P(\, X_\beta = \mathbf{x}\,)$ describing the adversary's *a priori* belief on the probability that a user has a particular profile $\mathbf{x}$. We further denote knowledge on a particular individual, say Alice, as $b(\mathrm{A})$. Hence, $P(\, \mathbf{x} \mid X_\beta, b(\mathrm{A})\,)$ represents the probability of Alice's real profile being $\mathbf{x}$ considering the adversary's prior knowledge X_β and $b(\mathrm{A})$.

Figure 4.4 shows an example of the probability density $P(\, X_\beta = \mathbf{x}\,)$, simplified to three dimensions, i.e. profiles $\mathbf{x} = \{x_1, x_2, x_3\}$ that have three components $0 \leq x_i \leq 1$ such that $\sum_i x_1 = 1$. Darker areas represent highly likely profiles, while lighter areas represent rarer profiles.

Strategy. The adversary is honest-but-curious (HbC) and does its best to retrieve profile $\mathbf{x}$. For analysis purposes, we classify adversaries' attack strategies in two categories:

Profile-based. Profile-based attacks exploit information about a tool's DGS at the profile level, i.e. they model the impact the DGS has on the users'

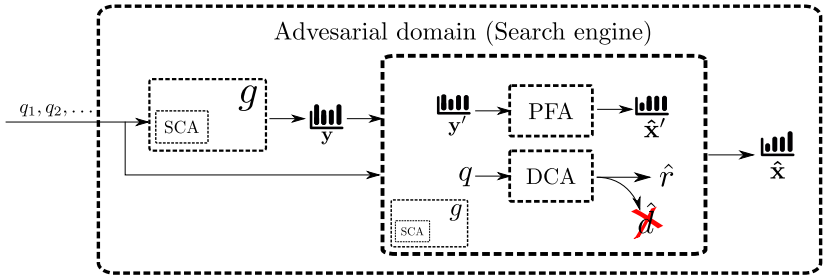


Figure 4.5: The adversary processes both real and dummy activities into an *observed* profile $\mathbf{y}$ which then post-processes into a *filtered* profile $\hat{\mathbf{x}}$.

profiles DGS and attempt to reverse or mitigate it. De-obfuscation attacks exploit, among other DGS features, the target profile $\mathbf{y}^t$ a DGS seeks to project.

Profile-based attacks rely on profile filtering algorithms (PFAs) that apply a set of *filtering rules* on the observed profile $\mathbf{y}$ to retrieve a filtered profile $\hat{\mathbf{x}}$. A profile-based attack fully succeeds when the PFA recovers $\hat{\mathbf{x}}$ without noise, i.e. when $\hat{\mathbf{x}} = \mathbf{x}$.

Query-based. Query-based attacks exploit information about the tool's DGS mechanisms that seek to ensure indistinguishability between real and dummy search actions. Query-based attacks exploit differences between real and dummy actions' features such as query semantics and grammar, timing or metadata, to tell dummies apart and discard them.

Query-based attacks rely on dummy classification algorithms (DCAs) that apply a set of *classifying rules* to tag the search activity as either real ($\hat{r}$) or dummy ($\hat{d}$). A query-based attack fully succeeds when all queries r and d are correctly classified as $\hat{r}$ and $\hat{d}$, respectively.

We note that classifying attacks as profile-based and query-based is ambiguous and open to interpretation. Information about a tool's obfuscation strategy that PFAs exploit also enable DCAs to better distinguish between real and dummy activity; a DCA's classification output results in a filtered profile $\hat{\mathbf{x}}$ that is closer to $\mathbf{x}$. This classification however assists our analysis of CBPWS tools, as we show in Sect. 4.3.

Figure 4.5 depicts the interaction between PFAs and DCA in the adversary's attack strategy to recover a filtered profile $\hat{\mathbf{x}}$ from the observed profile $\mathbf{y}$.

Model summary.

CBPWS users generate *real queries* r that an adversary processes with a profiling function g into a profile $\mathbf{x}$. A CBPWS tool receives as input *real queries* r and automatically generates *dummy queries* d according to a dummy generation strategy (DGS). The CBPWS tool sends both real and dummy queries to the adversarial web search provider, we refer to either type of query as q, to denote that the query may be real or dummy. The adversary constructs an observed profile $\mathbf{y}$ from the sequence of queries q. However, the adversary relies on DCAs and PFAs to exploit DGS's vulnerabilities and obtain a *filtered profile* $\hat{\mathbf{x}}$. Table 4.1 provides an overview of the notation we have introduced so far in this chapter.

4.2 Analysis framework

In this section we provide an analysis framework for CBPWS tools. To this end, we first define a set of privacy properties for private web search. Then, we instantiate the measures we introduce in Sect. 3.2 to capture such properties. Table 4.2 offers a summary of additional notation we use throughout this section.

4.2.1 Privacy properties

There are a number of privacy properties that a CBPWS tool may attempt to provide. The privacy concerns of any particular user relate to her own situation and needs, thus making it impossible for a privacy engineer to account for all possible privacy concerns in the design of a CBPWS tool.

Given the impossibility of providing an exhaustive list of the users' privacy concerns, we consider and redefine three general privacy properties that have been implicitly or explicitly considered in previous CBPWS tool designs, namely, *profile confidentiality* [198, 448], *query deniability* [396, 397] and *query undetectability* [182, 294]. In Sect. 4.3 we illustrate how to leverage this analysis framework to evaluate the very CBPWS tool designs that inspire it.

Profile confidentiality guarantees that the adversary cannot determine a user's search profile $\mathbf{x}$.

Query deniability guarantees that users are able to *deny* having issued a certain query, namely, if a user is accused of having searched for something,

Symbol	Meaning	Symbol	Meaning
r	Real query	d	Dummy query
q	Query (real or dummy) the adversary observes	o	Search result
$\hat{r}$	Query adversary classifies as real	$\hat{d}$	Query adversary classifies as dummy
r	Sequence of real queries	d	Seq. of dummy queries
q	Sequence of queries (both real and dummy)		
$\mathcal{R}$	Universe of real queries	$\mathcal{Q}$	Universe of queries (both real and dummy)
$\mathbf{x} = \{x_i\}$	Real profile	$\mathbf{y} = \{y_i\}$	Observed profile
$\hat{\mathbf{x}} = \{\hat{x}_i\}$	Filtered profile	$\mathbf{y}^t = \{y_i^t\}$	DGS' target profile
$\mathbf{w} = \{w_i\}$	Dummy profile		
g	Adversarial profiling function	$g_{\mathcal{T}}$	CBPWS tool profiling function
ρ	Dummy rate	X	Random variable over real profiles $\mathbf{x}$
Y	Random variable over observed profiles	$\hat{X}$	Random variable over filtered profiles $\hat{\mathbf{x}}$
SCA	Semantic classification algorithm	MOD	Profile components modulator
DCA	Dummy classification algorithm	PFA	Profile filtering algorithm
X_β	Adversarial prior knowledge on X	b(A)	Adversarial prior knowledge on Alice's profile $\mathbf{x}_\mathrm{A}$

Table 4.1: Overview of CBPWS model notation.

she should be able to *plausibly* claim that it was the CBPWS tool instead, i.e. that her query is an automatically generated dummy.

Query undetectability guarantees that the adversary classifies users' search queries as dummies, enabling users to evade targeting.

4.2.2 Privacy measures

Profile confidentiality.

MCA: As information leakage. We measure profile confidentiality using *mutual information* (see Sect. 3.2.1). We choose mutual information to evaluate CBPWS tools with independence of the adversary knowledge and particular attack strategy. We favour mutual information over min-entropy leakage to avoid making any assumptions about the number of guesses or candidate profiles $\hat{\mathbf{x}}$ the adversary considers for each individual. Moreover, we focus on mutual information instead of capacity because we do not aim to account for all possible probability distributions of web search profiles $P(\, X = \mathbf{x}\,)$.

Symbol	Meaning	Symbol	Meaning
$\hat{\mathrm{f}}$	Query sequence adversary classifies as real	$\hat{\mathrm{d}}$	Query sequence adversary classifies as dummy
R	R.v. over sequences of real queries	Q	R.v. over sequences of observed queries
$\mathbf{R}$	Universe of real query sequences	$\mathbf{R}_{\mathrm{q}}$	Multiset of q subsequences
r_d	Query sequence to deny	$\mathbf{R}_d$	Multiset of q subsequences containing r_d
$\mathbf{R}_d^c$	Multiset of q subsequences complementary to $\mathbf{R}_d$	ℓ	Distance
I	Mutual information	$\mathbf{E}$	Expected estimation error
$\mathrm{N}_{\mathrm{IND}}$	*A priori* deniability	$\mathrm{U}_{\mathrm{IND}}$	*A priori* undetectability
N	Deniability	U	Undetectability

Table 4.2: Overview of CBPWS' analysis framework notation.

We thus compute mutual information between the input (real) search activity sequences random variable R and output (obfuscated) search activity sequences random variable Q as:

$$\mathrm{I}(\mathrm{R};\mathrm{Q}) = \mathrm{H}(\mathrm{R}) - \mathrm{H}(\mathrm{R} \mid \mathrm{Q}) \tag{4.2}$$

We say that a CBPWS tool provides *perfect profile confidentiality* if the tool leaks *zero* bits of information about the input real sequences X from the output profiles r_i; i.e. $\mathrm{I}(\mathrm{R};\mathrm{Q}) = 0$. Conversely, the tool provides no profile confidentiality at all when the adversary gains $\mathrm{H}(\mathrm{R})$ bits of information from Q, namely, $\mathrm{H}(\mathrm{R} \mid \mathrm{Q}) = 0 \Rightarrow \mathrm{I}(\mathrm{R};\mathrm{Q}) = \mathrm{H}(\mathrm{R})$. In this case, the tool leaks enough information for adversaries to perfectly reconstruct real profiles $\mathbf{x}$ from observed profiles $\mathbf{y}$.

Lastly, we recall that mutual information is an *average* measure of leakage, i.e. it does not bound information leakage on particular individuals. Assessing the information a CBPWS tool leaks about specific users requires restricting the output random variable Q to the sequences the DGS generates on a particular individual's input, namely, compute $\mathrm{I}(\mathrm{R};\mathrm{Q}_\mathrm{A})$ with $\mathrm{Q}_\mathrm{A} = \Omega(\mathrm{r}_\mathrm{A})$ for a user's, say Alice's, search actions sequence r_A. Besides, because we choose to abstract away from adversarial knowledge, mutual information does not measure the privacy protection against particular adversaries with arbitrary background knowledge. We recall that accounting for specific instances of background knowledge requires an attack-centred analysis (ACA), as enabled by expected estimation error (EEE).

ACA: As expected estimation error. To evaluate the performance of a CBPWS tool against a particular adversary or attack, we resort to expected estimation error as given by Eq. 3.16. To recall,

$$\mathbf{E} = \sum_{\mathbf{x}} P(\,\mathbf{x}\,) \sum_{\mathrm{q}} P(\,\mathrm{q} \mid \mathbf{x}\,) \sum_{\hat{\mathbf{x}}} P(\,\hat{\mathbf{x}} \mid \mathrm{q}\,) \cdot \ell(\hat{\mathbf{x}}, \mathbf{x})$$

where $\mathbf{x} = g(\mathrm{r})$ represents a user's profile, with r the user's sequence of search activities and g the adversary's profiling function; q the obfuscated sequence of search activities the adversary observes; $\hat{\mathbf{x}}$ the adversary's estimation of the user profile after processing and filtering q and $\ell(\mathbf{x}, \hat{\mathbf{x}})$ a distance that represents the privacy improvement that results from the adversary obtaining $\hat{\mathbf{x}}$ instead of $\mathbf{x}$. We do not specify the profiling function g or distance function ℓ, as these depend on the particular adversary against which we evaluate CBPWS tools.

Deniability and undetectability.

We provide two alternative formulations of deniability and undetectability. On the one hand, we provide an indistinguishability-based definition that enables a mechanism-centred analysis (MCA). On the other hand, we provide an expected estimation error definition that enables an attack-centred analysis (ACA).

As indistinguishability.

Deniability. Deniability requires that users are able to plausibly claim that a particular query or sequence of queries is the outcome of the DGS as opposed to their own search activity. Hence, we measure deniability as the probability that, given an output sequence q, the CBPWS tool generates a query or sequence of queries r_d that we wish to deny. In other words, we measure the probability that the DGS generates r_d, as opposed to the user. Hence, we formalise *a priori deniability* $\mathrm{N}_{\mathrm{IND}}$ for two sequences r_d and q as:

$$\mathrm{N}_{\mathrm{IND}} = P(\,\mathbf{R}_d^c \mid \mathrm{q}\,) \tag{4.3}$$

where q denotes the output sequence of the CBPWS, $\mathbf{R}_d$ denotes the multiset of subsequences of q that contain the query or sequence r_d users wish to deny and $\mathbf{R}_d^c = \mathbf{R}_{\mathrm{q}} \setminus \mathbf{R}_d$ denotes the complementary set of $\mathbf{R}_d$, i.e. the set of subsequences of q that do *not* contain r_d (see notation table in page 125). Expanding and applying Bayes,

$$\mathrm{N}_{\mathrm{IND}} = \sum_{\mathrm{r}_j \in \mathbf{R}_d^c} P(\,\mathrm{r}_j \mid \mathrm{q}\,) = \frac{\sum_{\mathrm{r}_j \in \mathbf{R}_d^c} P(\,\mathrm{q} \mid \mathrm{r}_j\,) \cdot P(\,\mathrm{r}_j\,)}{\sum_{\mathrm{r}_i \in \mathbf{R}_{\mathrm{q}}} P(\,\mathrm{q} \mid \mathrm{r}_i\,) \cdot P(\,\mathrm{r}_i\,)} \tag{4.4}$$

where the term $\frac{P(\,\mathrm{q} \mid \mathrm{r}_j\,)}{P(\,\mathrm{q} \mid \mathrm{r}_i\,)}$ measures CBPWS indistinguishability and $\frac{P(\,\mathrm{r}_j\,)}{P(\,\mathrm{r}_i\,)}$ the ratio between the prior probability of the real subsequences in q that do not contain r_d and all the subsequences $\mathrm{r}_i \in \mathbf{R}_{\mathrm{q}}$.

A priori deniability takes its minimum value at $\mathrm{N}_{\mathrm{IND}} = 0$, when $P(\,\mathrm{q} \mid \mathbf{R}_d^c\,) = 0$ (namely, the probability that the CBPWS tool generates sequence q from a real sequence that does not contain r_d is zero) or the prior $P(\,\mathrm{r}_j\,)$ is zero. Its maximum value depends on the interplay between the level of indistinguishability and the priors. In particular, when all q subsequences are indistinguishable, i.e. when $P(\,\mathrm{q} \mid \mathrm{r}_j\,) = P(\,\mathrm{q} \mid \mathrm{r}_i\,) \quad \forall \mathrm{r}_i, \mathrm{r}_j \in \mathbf{R}_{\mathrm{q}}$ deniability depends on the cardinality of the multiset of possible sequences and their prior probability alone, i.e.

$$\mathrm{N_{IND}} = \frac{\sum_{\mathrm{r}_j \in \mathbf{R}_d^c} P(\,\mathrm{r}_j\,)}{\sum_{\mathrm{r}_i \in \mathbf{R}_\mathrm{q}} P(\,\mathrm{r}_i\,)} \tag{4.5}$$

the rationale being that the CBPWS tool does not disclose any additional information, so the adversary must guess based on prior probabilities and background knowledge alone. If we further consider that all prior sequences have the same probability $P(\,\mathrm{r}_i\,) = P(\,\mathrm{r}_j\,)$ then $\mathrm{N_{IND}} = \frac{|\mathbf{R}_d^c|}{|\mathbf{R}_\mathrm{q}|}$.

Dependence on the cardinality of $\mathbf{R}_\mathrm{q}$ and prior probabilities further highlights the interplay between indistinguishability and deniability. On the one hand, the more indistinguishable subsequences in q, the higher the chance that their combined prior probability offsets that of subsequences $\mathbf{R}_d$, increasing deniability. On the other hand, for a given indistinguishability level, the higher the prior probability of subsequence r_d, the harder it is for users to deny.

Undetectability. Undetectability requires that adversaries classify a particular query or sequence of queries as dummy rather than real. Hence, we measure undetectability as the probability that, given an output sequence q, the CBPWS has generated it. We formalise *a priori undetectability* $\mathrm{U_{IND}}$ for two particular sequences r_d and q as:

$$\mathrm{U_{IND}} = P(\,\mathbf{R}_d^c \mid \mathrm{q}\,) = \sum_{\mathrm{r}_j \in \mathbf{R}_d^c} P(\,\mathrm{r}_j \mid \mathrm{q}\,) = \frac{\sum_{\mathrm{r}_j \in \mathbf{R}_d^c} P(\,\mathrm{q} \mid \mathrm{r}_j\,) \cdot P(\,\mathrm{r}_j\,)}{\sum_{\mathrm{r}_i \in \mathbf{R}_\mathrm{q}} P(\,\mathrm{q} \mid \mathrm{r}_i\,) \cdot P(\,\mathrm{r}_i\,)} \tag{4.6}$$

We note that Eq. 4.6 matches Eqs. 4.3 and 4.4 above. A priori undetectability equals a priori deniability, i.e. $\mathrm{U_{IND}} = \mathrm{N_{IND}}$, because they estimate classification errors that have not yet taken place by computing the probability that a sequence of queries is made of dummies. Both deniability and undetectability depend on the probability that a query is a dummy, the former to enable a user to deny having issued a query that the adversary correctly classifies as real, i.e. to claim that her real query *should* have been classified as a dummy, the latter to actually have the adversary classify real queries as dummies. Moreover, because we formalise both properties in terms of indistinguishability, we abstract away from the particular attack the adversary deploys and the errors therein, relying on prior probabilities and the CBPWS operations alone. Hence, *a priori*, deniability and undetectability are the same. A posteriori, however, the adversary may deploy a suboptimal attack or attempt to minimise false positives (dummy queries classified as real) at the expense of false negatives (real queries classified

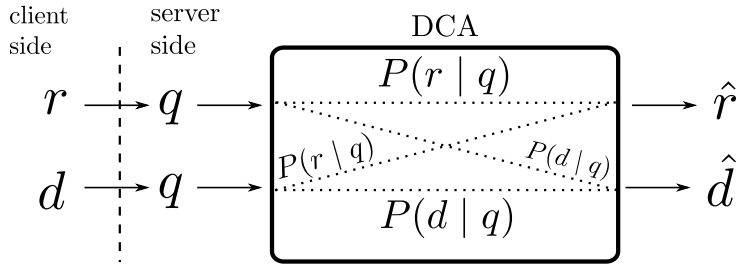


Figure 4.6: Dummy classification algorithm.

as dummy), thus making deniability and undetectability diverge. That is why we refer to these indistinguishability-based measures as *a priori*.

As expected estimation error.

We consider the adversary relies on a DCA to classify queries as either dummy or real, obtaining a sequence $\hat{\mathrm{r}}$. Figure 4.6 depicts an abstract representation of a DCA and the classification process. To capture the actual level of deniability and undetectability users have against a particular adversary, we formulate them as expected estimation error (EEE). From Eq. 3.16, we substitute profiles with sequences to obtain:

$$\mathbf{E} = \sum_{\mathrm{r}} P(\,\mathrm{r}\,) \sum_{\mathrm{r}} P(\,\mathrm{q} \mid \mathrm{r}\,) \sum_{\hat{\mathrm{r}}} P(\,\hat{\mathrm{r}} \mid \mathrm{q}\,) \cdot \ell(\hat{\mathrm{r}}, \mathrm{r})$$

where $\ell(\hat{\mathrm{r}}, \mathrm{r})$ denotes either of the two distance functions we use to capture deniability and undetectability, as we show below.

Deniability We measure the level of *deniability* (N) as the proportion of queries a DCA classifies as real that are dummies. We define

$$\ell_{\mathrm{N}} \equiv \frac{|\hat{\mathrm{r}} \setminus \mathrm{r}|}{|\hat{\mathrm{r}}|}$$

where $|\hat{\mathrm{r}} \setminus \mathrm{r}|$ represents the cardinality of the relative complement of r in $\hat{\mathrm{r}}$, namely, the number of dummy queries the adversary assigns to the filtered sequence r. The distance ℓ_{N} represents the ratio of dummy queries in the sequence of queries the adversary classifies as real. Hence, we obtain deniability N as:

$$\mathrm{N} = \mathbf{E}_{\mathrm{N}} \equiv \sum_{\mathrm{r}} P(\,\mathrm{r}\,) \sum_{\mathrm{q}} P(\,\mathrm{q} \mid \mathrm{r}\,) \sum_{\hat{\mathrm{r}}} P(\,\hat{\mathrm{r}} \mid \mathrm{q}\,) \cdot \frac{|\hat{\mathrm{r}} \setminus \mathrm{r}|}{|\hat{\mathrm{r}}|} \tag{4.7}$$

Undetectability We measure the level of *undetectability* (U) as the proportion of real queries a DCA classifies as dummy. We define

$$\ell_{\mathrm{U}} \equiv \frac{|\mathrm{r} \setminus \hat{\mathrm{r}}|}{|\mathrm{r}|}$$

where $|\mathrm{r} \setminus \hat{\mathrm{r}}|$ represents the cardinality of the relative complement of $\hat{\mathrm{r}}$ in r, namely, the number of real queries the adversary classifies as dummies. The distance ℓ_{U} represents the proportion of real queries the adversary classifies as dummy with respect to the total number of real queries. Hence, we obtain undetectability U as:

$$\mathrm{U} = \mathbf{E}_{\mathrm{U}} \equiv \sum_{\mathrm{r}} P(\,\mathrm{r}\,) \sum_{\mathrm{q}} P(\,\mathrm{q} \mid \mathrm{r}\,) \sum_{\hat{\mathrm{r}}} P(\,\hat{\mathrm{r}} \mid \mathrm{q}\,) \cdot \frac{|\mathrm{r} \setminus \hat{\mathrm{r}}|}{|\mathrm{r}|} \tag{4.8}$$

A CBPWS provides zero deniability and undetectability if $\hat{\mathrm{r}} = \mathrm{r}$, i.e. either the DGS reveals which queries are real and which ones are dummy or the adversary has enough background information to accurately tell all real and dummy queries apart. A CBPWS provides maximum deniability and undetectability when the adversary's best guess is to classify queries as dummies based on the proportion of dummies issued by the obfuscation tool and the prior probability of any query to be real.

On topic agnosticism.

We remark in Sect. 3.2 that EEE is unique among the analysis measures we propose in that it incorporates the adversarial profiling function g that maps each sequence of queries r to a profile $\mathbf{x} = g(\mathrm{r})$. Moreover, (ϵ, ℓ, δ)-local differential privacy (LDP) incorporates a notion of distance ℓ between sequences r and r′ that allows CBPWS designers to guarantee varying levels of indistinguishability ϵ according to distance $\ell(\mathrm{r}, \mathrm{r}')$.

In private web search, measures that do not account for the profiling function g and its semantic classification algorithm (SCA) are oblivious to the mapping between queries and topics, which means that any dummy query that differs from a real query *obfuscates "equally well"*. To illustrate, let us consider three CBPWS tools, T1, T2 and T3, that process user query "Karl Marx". Tool T1 generates dummy queries "Friedrich Engels" and "Rosa Luxemburg". Tool T2 generates dummy queries "Charles Darwin" and "Leonardo da Vinci". Tool T3 generates dummy queries "homemade Nutella" and "Bohemian Rhapsody". Let us further consider that all three T1, T2 and T3 provide the same level of deniability and undetectability, namely, $\mathrm{N} = \mathrm{U} = \frac{2}{3}$. The level of deniability

T1 provides is worthless if we assume the user wishes to hide she is interested in "communism". However, if the user is interested in "famous bearded guys" instead, T1 provides a plausible cover, while T2 does not. To believe that T3 provides better cover simply because no apparent topic relationship exists between real and dummy queries relies on a set of assumptions about the mapping between queries and topics outside the control of a CBPWS tool, i.e. the combination "*Karl Marx, homemade Nutella, Bohemian Rhapsody*" may reinforce a user's real profile components x_i in ways that CBPWS designers are not aware of.

Topic-agnostic measures are oblivious to query-topic mappings, thus potentially capture indistinguishability between real and dummy queries that from the user's or adversary's point of view are equivalent. *Topic-aware* measures on the other hand capture a set of query-topic mappings to prevent paradoxes like in the example above, yet at the risk of overestimating protection under alternative query-topic mappings. As Fig. 3.4 illustrates, the more assumptions underlie a measure, the less generality, yet more expressivity with respect to a particular adversary and attack. The evaluation of CBPWS tools in Sect. 4.3 sheds further light on the implications of using either type of measure.

4.3 Evaluation of chaff-based private web search tools

In this section we provide an overview of CBPWS tools that have been proposed in the literature. We bring these designs under the model and analytical framework we have introduced above to demonstrate our model's ability to describe diverse tools and the suitability of our analytical framework to evaluate CBPWS.

Our evaluation reveals pervasive flaws and misconceptions in the design and analysis of CBPWS tools. We analyse and deconstruct why existing tools fail to adequately address the challenges that CBPWS involves, identifying common pitfalls and solutions to address them.

Lastly, we note that bringing these tools under our framework involves replacing the original notation and concepts they define with ours.

4.3.1 GooPIR: $h(k)$-Private Information Retrieval[3]

GooPIR is a web search program available for Windows and Unix[4] that adds dummy keywords to users' queries before forwarding them to Google, then retrieves and returns to users only the search results that relate to the real keywords [182]. Instead of generating and sending additional, separate dummy queries as we model CBPWS in Sect. 4.1.3, GooPIR selects $k-1$ dummy queries and *'ORs'* them to the real query r. In fact, GooPIR's authors say to rely on Google not only because of its popularity, but also because it offers the possibility to OR query terms in a single query. A regular Google search 'ANDs' the keywords in each query, i.e. each search result is relevant to *all* the keywords in a query. Conversely, in an OR-ed query, Google returns results for each OR-ed keyword separately. GooPIR retrieves the results Google provides, then returns to the user only those that relate to the real query, filtering away dummy search results. GooPIR does not modify the real query's keywords nor intends to retrieve different results to what real queries alone produce. Hence, in practice, OR-ing $k-1$ dummy keywords to the real query is analogous to sending $k-1$ dummy queries simultaneously with the real query. Hence, to better accommodate this design choice in our model, we denote each set of real keywords as real query r, each set of dummy keywords as dummy query d, and the set that GooPIR sends simultaneously as query $q = \{r \vee d_1 \vee \ldots \vee d_{k-1}\}$.

Privacy property and measure. GooPIR aims to offer what authors Domingo-Ferrer et al. denominate $h(k)$-*private information retrieval* ($h(k)$-PIR). A CBPWS satisfies $h(k)$-PIR if the uncertainty of any adversary about the real query r corresponds to $\mathrm{H}(\hat{R}) \geq h(k)$, with $\hat{R}$ the random variable that models the *'value'* the adversary estimates the real query r takes, for some function h and a non-negative integer k so that $h(k) \geq 0$.

Dummy generation strategy. GooPIR's DGS relies on the following tactics:

Simultaneous submission of real and dummy queries as part of the same query set q, to prevent the adversary from exploiting dummy queries' timing and metadata to filter them away.

[3] Whereas throughout the evaluation of existing CBPWS tools we perform in this section we adapt each paper's notation to match the one we introduce in this thesis, in this particular case we keep the author's notation $h(k)$ because it is included in the paper's title. We however warn that it conflicts with the notation of h as a social utility function (q.v. Sect. 2.2) and that GooPIR's $h(k)$ is entirely unrelated to the concept of social utility.

[4] Available for download `http://unescoprivacychair.urv.cat/goopir.php`

Equally popular accompanying dummies. GooPIR's designers acknowledge that the adversary may be able to use a DCA that exploits the *popularity* of queries to identify and remove dummies, i.e. with more popular queries having a higher probability of being the real query. To overcome this threat, GooPIR computes the popularity of the real query then selects $k-1$ dummy queries with a similar popularity. GooPIR assumes query popularity to be proportional to the frequency of its keywords' appearance in the Web and that a public dictionary labelled with such frequencies is available.

Fixed sets of accompanying dummies. To prevent disclosure attacks [13, 151] GooPIR *always* expands each real query r with *the same set* of $k-1$ dummy queries. Thus GooPIR prevents recurrent real queries from appearing more frequently than dummies, thwarting adversaries' attempts to perform frequency analyses to identify them.

Evaluation

Authors Domingo-Ferrer et al. provide an evaluation of GooPIR, concluding that it provides $h(k)$-PIR [182]. Their analysis however underestimates the adversary; it disregards adversaries' background knowledge and neglects the fact that adversaries can exploit topic correlations across sequences of queries r.

GooPIR's privacy measure, $h(k)$-PIR. GooPIR's $h(k)$-PIR is an attack-centred analysis (ACA) measure of the adversary's uncertainty about users' real queries, namely, the entropy of the random variable the adversary estimates to describe real query values, $\mathrm{H}(\hat{R})$. GooPIR's $h(k)$-PIR differs from information gain (q.v. Sect. 3.2.2) in that it does not consider an adversary's prior belief. Whereas $h(k)$-PIR measures the privacy threat an adversary poses, it does not capture to what extent GooPIR (or any other CBPWS tool) is responsible for it, i.e. $h(k)$-PIR is oblivious to the relationship between the prior information of the adversary about $P_\beta(R=r)$ and the estimation $P_\beta(R=r \mid Q=q) = P(\hat{R})$.

Hence, as an ACA measure that focuses on adversarial uncertainty, $h(k)$-PIR is inadequate as a general CBPWS design constraint. No CBPWS can bound the uncertainty of the adversary; if the adversary has extremely accurate prior knowledge —so that $\mathrm{H}(\hat{R}) < h(k)$ *before* observing a CBPWS tool's output— the CBPWS can at most, *on average*,[5] prevent the adversary from gaining additional

[5] On particular attack scenarios and observations the entropy of the adversary may increase, e.g. if by chance the user's behaviour contradicts the (accurate) background knowledge the adversary has about her [130, 173]. This is why *information gain* is a more suitable measure to evaluate particular attacks and scenarios (q.v. Sect. 3.2.2).

knowledge and further reducing $\mathrm{H}(\hat{R})$, rather than increasing its uncertainty to $\mathrm{H}(\hat{R}) \geq h(k)$. Conversely, mechanism-centred analysis (MCA) measures such as indistinguishability and information leakage (q.v. Sect. 3.2.1), abstract away from adversaries' knowledge and focus on the obfuscation mechanism alone, offering more general guarantees that apply to a wider range of adversaries, prior beliefs $P_\beta(R = r)$ and prior probability distributions $P(\, R = r\,)$; that is why we favour them to impose design constraints.

GooPIR's DGS. Since no CBPWS can control the prior knowledge of an adversary, it follows that GooPIR cannot generally provide $h(k)$-PIR.

Let us however assume a weaker adversary, namely, one that does not have information about each particular user —say Alice, with $P_\mathrm{A}(R_\mathrm{A} = r)$— yet has perfect information on the probability distribution of the general user population, i.e. $P_\beta(R = r) = P(\, R = r\,)$. Let us further assume that, for each real query r, GooPIR generates $k-1$ dummy queries d_j with exactly the same prior probability $P(\, r\,) = P(\, d_j\,), \forall j \in \{1, k-1\}$ —that GooPIR assumes to match its online popularity, as we examine below. Under these conditions, according to GooPIR's designers, the adversary cannot distinguish real queries from dummies and, as a result, GooPIR would provide $h(k)$-PIR with $h(k) = \mathrm{H}(R) = \log(k)$.

However, unless we assume that *there is no logical sequence, no dependence between successive user queries*, such guarantees do not hold, as GooPIR disregards the semantic relationship between real keywords in successive user queries. Because GooPIR chooses dummies for each query independently, it does not hide correlations between real terms relating to a particular topic or family of topics. As an example, let us consider $k = 3$ and the following three sets of queries: $q_1 = \{$"lion", "airport", "vacancy"$\}$, $q_2 = \{$"song", "shower", "leopard"$\}$, $q_3 = \{$"stock", "tiger", "ribbon"$\}$. Figure 4.7 depicts the web search provider receiving these three queries. A DCA uses a SCA to learn that topic "big cats" appears more often than others, hence it is more likely that the user sent queries $\{$"lion", "leopard", "tiger"$\}$ than any other combination, even if the frequency of the keywords in each query is roughly the same and each real real keyword is always accompanied by the same dummies.

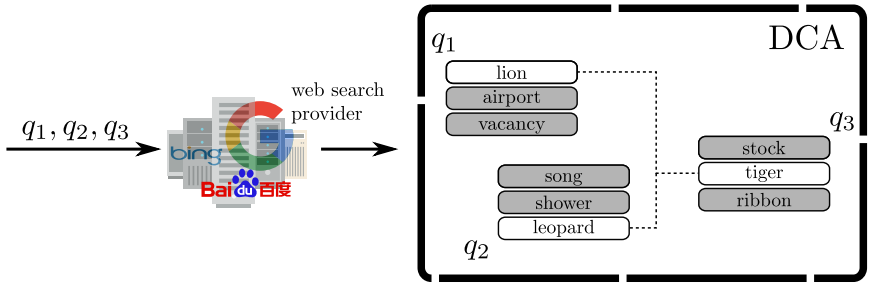


Figure 4.7: SCA attack on GooPIR, exploiting topic similarity across successive queries. Connected keywords over white background most likely to be real. Greyed-out keywords discarded as dummies.

If $h(k) = \mathrm{H}(\hat{R}) = \log(k)$, it follows that in terms of deniability:[6]

$$\mathrm{N}_{[h(k)=\log(k)]} = \sum_{r} P(\,r\,) \sum_{q} P(\,q \mid r\,) \sum_{\hat{r}} P(\,\hat{r} \mid q\,) \cdot \frac{|\hat{r} \setminus r|}{|\hat{r}|} \tag{4.9a}$$

$$= \sum_{r} P(\,r\,) \sum_{q} \cancel{P(\,q \mid r\,)}^{1} \sum_{1}^{k-1} \frac{1}{k} \tag{4.9b}$$

$$= \frac{k-1}{k} \sum_{r} P(\,r\,) = \frac{k-1}{k} = \rho \tag{4.9c}$$

i.e. deniability matches the dummy rate, as that is precisely the ratio of dummies the adversary classifies as real. However, Eq. 4.9a requires that $\forall \mathrm{r}, P(\,\mathrm{r}\,) = \prod_{r \in \mathrm{r}} P(\,r\,)$. Under this assumption, $P(\,\hat{\mathrm{r}} \mid \mathrm{q}\,) \cdot |\hat{\mathrm{r}} \setminus \mathrm{r}|$ follows a binomial distribution with parameters $n = |\hat{\mathrm{r}}|$ and $p = (k-1)/k$, with p the marginal probability of any query $\hat{r}$ being a dummy, as the adversary's optimal strategy involves randomly selecting one query from each query set of k queries in a sequence of $|\mathrm{q}| = |\hat{\mathrm{r}}|$ querysets, with a success probability (for the user) of selecting a dummy within each queryset of $\frac{k-1}{k}$. Hence,

$$\sum_{\hat{\mathrm{r}}} P(\,\hat{\mathrm{r}} \mid \mathrm{q}\,) \cdot \frac{|\hat{\mathrm{r}} \setminus \mathrm{r}|}{|\hat{\mathrm{r}}|} = \frac{k-1}{k}$$

However, this is not generally true if $P(\,\mathrm{r}\,) \neq \prod_{r \in \mathrm{r}} P(\,r\,)$, i.e. if users do not generate each query independently of other queries.

[6] For brevity we omit results for undetectability, as assuming $h(k) = \log(k)$ implies $\mathrm{U} = \mathrm{N}$.

Despite the fact that GooPIR disregards topic correlation across query sequences, thereby underestimating the level of indistinguishability between reals and dummies, we further examine the role of two of GooPIR's DGS elements in achieving indistinguishability.

We recall that a priori deniability

$$\mathrm{N_{IND}} = \frac{\sum_{\mathrm{r}_j \in \mathbf{R}_d^c} P(\,\mathrm{q} \mid \mathrm{r}_j\,) \cdot P(\,\mathrm{r}_j\,)}{\sum_{\mathrm{r}_i \in \mathbf{R}_\mathrm{q}} P(\,\mathrm{q} \mid \mathrm{r}_i\,) \cdot P(\,\mathrm{r}_i\,)}$$

where the term $\frac{P(\,\mathrm{q} \mid \mathrm{r}_j\,)}{P(\,\mathrm{q} \mid \mathrm{r}_i\,)}$ represents a CBPWS tool's indistinguishability and the term $\frac{P(\,\mathrm{r}_j\,)}{P(\,\mathrm{r}_i\,)}$ the relationship between priors.

GooPIR's *fixed set of accompanying dummies* ensures that $\forall r_i, r_j \in q, P(\,q \mid r_i\,) = P(\,q \mid r_j\,) = 1$, even if it fails to ensure indistinguishability because it does not account for sequences of queries r so that $\forall \mathrm{r}, \mathrm{r}' \in \mathrm{q}, P(\,\mathrm{q} \mid \mathrm{r}\,) = P(\,\mathrm{q} \mid \mathrm{r}'\,) = 1$. If we assumed independence between queries however, GooPIR would indeed provide indistinguishability between real and dummy queries so that, *for each query* r_i in query set q:

$$\mathrm{N_{IND}}(r_i) = \frac{\sum_{r_j \in q \mid r_j \neq r_i} P(\,r_j\,)}{\sum_{r_j \in q} P(\,r_j\,)} \tag{4.10}$$

showing that, despite indistinguishability between $\{r_i\} \in q$, the adversary can still exploit knowledge about a priori probabilities to undermine deniability, e.g. if the CBPWS tool accompanies a popular query r_d with extremely uncommon queries r_j so that $\sum_{r_j \in q \mid r_j \neq r_d} P(\,r_j\,) \ll \sum_{r_i \in q} P(\,r_i\,)$ then $\mathrm{N_{IND}}(r_d) \to 0$.

To prevent that $\sum_{r_j \in q \mid r_j \neq r_d} P(\,r_j\,) \ll \sum_{r_i \in q} P(\,r_i\,)$, GooPIR resorts to *equally popular accompanying dummies*, in an attempt to ensure that $P(\,r_i\,) \simeq P(\,r_j\,)\ \forall r_i, r_j \in q$. This equalises deniability for every query so that

$$\mathrm{N_{IND}}(r_d) = \frac{\sum_{r_j \in q \mid r_j \neq r_d} P(\,r_j\,)}{\sum_{r_i \in q} P(\,r_i\,)} = \frac{k-1}{k}$$

These two GooPIR DGS's elements illustrate the type of guarantees indistinguishability, as a measure, provides and limitations thereof. On the one hand, indistinguishability holds regardless of what the adversary knows or the prior probability distributions $P(\,R = r\,)$ and user Alice's $P_\mathrm{A}(R = r)$; i.e. with fixed sets of accompanying dummies, no amount of adversarial knowledge or changes

in $P(\, R = r \,)$ can alter the fact that $P(\, q \mid r_j \,) = P(\, q \mid r_i \,)$, $\forall r_i, r_j \in q$. On the other hand, because indistinguishability is oblivious to external information sources a Proto cannot control, adversaries can still exploit those sources of knowledge the CBPWS does not account for to undermine its protection. With fixed sets of accompanying dummies alone, an uninformed adversary cannot distinguish which of the k queries in q is the real query, all of them are equally likely with $P(\, r_i \in q \,) = \frac{1}{k}$. An informed adversary can however determine that one of the queries r_i has a higher probability of being real, thus defeating the CBPWS tool's protection. Hence, a CBPWS tool can account for the prior probability $P(\, R = r \,)$ and strategically select accompanying sets of equally likely queries, so that the informed adversary cannot exploit that knowledge in its attack. Still, whereas *fixed sets of accompanying dummies* guarantees indistinguishability regardless of priors and adversary knowledge, *equally popular queries* depends on an assumption about what the adversary knows and succumbs to changes, alterations and misestimations. GooPIR assumes probability distribution $P(\, R = r \,)$ equals queries' online popularity according to some universal reference dictionary or corpus, thus does not account for and is vulnerable to attacks based on the evolution, popularity shifts and trends of web search, or specific information about individual users, e.g. the adversary may detect a sudden increase in query frequency related to trends and viral content of interest to the user, as the probability of choosing those terms would not be explained by the dictionary alone —unless such a dictionary is made aware of short-lived surges of query term popularity.

Moreover, by equalising deniability and undetectability to $\frac{k-1}{k}$ the *equally popular queries* strategy forestalls the ability of users to enjoy higher levels of deniability or undetectability if the prior probability or adversarial knowledge is in their favour, e.g. in the example we provide earlier, where $\sum_{r_j \in q \mid r_j \neq r_d} P(\, r_j \,) \ll \sum_{r_i \in q} P(\, r_i \,)$, if the user's query is actually one of the uncommon queries r_j, a priori r_j becomes undetectable, as $\mathrm{N}_{\mathrm{IND}}(r_j) \to 1$.

Conclusion. GooPIR's DGS relies on *simultaneous submissions* and *fixed sets of accompanying dummies* seeking to ensure indistinguishability between reals and dummies. Whereas the rationale for both strategies is well founded, it disregards topic correlations across sequences of queries that enable a DCA to tell reals and dummies apart. Moreover, its *equally popular dummies* strategy seeks to prevent an adversary from exploiting information about the popularity of queries, yet assumes that popularity to be static and true, disregarding mismatches between online popularity and the actual prior probability distribution $P(\, R = r \,)$, as well any other additional source of information.

4.3.2 Plausibly deniable search

Murugesan and Clifton propose *Plausibly deniable search* (PDS), a CBPWS tool that aims to enable a user to *plausibly deny* her search queries [396, 397]. Similarly to GooPIR, PDS accompanies each real query r with $k-1$ dummy queries to compose a set of k queries S_q and defeat timing attacks; it also relies on assumptions about adversarial knowledge and attack strategies that undermine the privacy guarantees it purports to offer.

Privacy property and measure. Muruguesan and Clifton seek to provide *Plausible Deniable Privacy* (PD-Privacy), which they define as a property of queryset $S_q = \{q_1, \ldots, q_k\}$, where one query q_i is the real user query q_v and the remaining $k-1$ queries are dummies. Set S_q satisfies k-PD-Privacy if:

i) any query $q_j \in S_q$ leads to S_q with equal probability.

ii) all $q_j \in S_q$ relate to different topics.

iii) all $q_j \in S_q$ are equally plausible, i.e. the probability that the CBPWS generates any $q_j \in S_q$ is similar to the probability that users generate q_j themselves.

Dummy generation strategy. To guarantee PD-Privacy, Murugesan and Clifton propose the following tactics:

Simultaneous submission. Analogously to GooPIR, each real query is accompanied by $k-1$ dummy queries to prevent an adversary from exploiting query timing and metadata.

Canonicalisation. PDS substitutes user queries with *canonical queries* [396, 397]. PDS builds a dictionary of *canonical terms* that represent the universe of documents that users are interested in, then generates a set of *canonical queries* by selecting the canonical terms that are closest in the semantic space, i.e. PDS assigns semantically similar terms —assuming they retrieve the same or similar documents— to a single, canonical query. Thus PDS assigns to each real query q_v the canonical query q_i that is closest in the distance space, so to obtain equal or similar as possible results to the ones the real user query q_v would retrieve.

As canonicalisation entails a potential drop in user utility, PDS does not satisfy the fundamental utility-preserving principle underlying Protos.

Instead, PDS better represents a hybrid approach that combines utility-preserving and utility-degrading obfuscation. In our analysis of PDS, we do not examine the canonicalisation strategy's impact on utility, implicitly assuming that the universe of canonical queries is rich enough and there is a sound replacement strategy to preserve user utility. At the same time, we acknowledge that realising such a canonicalisation strategy is far from trivial.

PD-Querysets. In addition to replacing the real user query q_v with a canonical query q_i, PDS's DGS generates PD-Querysets, this is, sets S_q of k canonical queries that PDS submits simultaneously. PDS generates PD-Querysets according to the following principles:

Coverage. PDS ensures that there is a query $q_i \in Q$ that is equal or similar enough to q_v to retrieve similar results from the search engine to the ones q_v obtains.

Deniability. PDS ensures that for every $q_j \in Q$ there is at least one user query q'_v that maps to canonical query q_j. We note that PDS's definition of deniability is independent and differs from our definition of deniability in Sect. 4.2.2.

Diversity. PDS groups k canonical queries that are as far as possible from each other in the semantic space, seeking to satisfy PD-Privacy's second requirement.

Plausibility. PDS groups k canonical queries according to the number of real queries that map to them. Canonical queries to which a large number of real queries map to are included in the same PD-Queryset. Similarly, PDS groups in the same PD-Queryset canonical queries to which a small number of real queries map to, queries PDS considers *implausible*.

Murugesan and Clifton acknowledge that, similarly to GooPIR, PDS does not defend against attacks that exploit sequences of *edited queries*, this is, sequences of queries that users send to narrow down their search or further explore a topic. They argue that PDS may not protect against this threat, as it is possible that dummy queries' topics in a sequence of PD-Querysets do not match, i.e. that dummy queries in the first queryset relate to different topics than the ones in the second query set and so on. As each query set Q is generated independently from each other, if sequential real queries map to different canonical queries, PDS does not guarantee that the corresponding dummy queries in their respective query sets are related to each other the way that real queries are.

Evaluation.

PD-Privacy. We review the three conditions Murugesan and Clifton set for a CBPWS tool to satisfy k-PD-Privacy.

The first condition enforces query indistinguishability, requiring that a CBPWS tool generates a query set S_q with the same probability for any $q_j \in S_q$, i.e. requiring that $P(\, S_q \mid q_j \,) = P(\, S_q \mid q_i \,)\ \forall q_i, q_j \in S_q$. As in our evaluation of GooPIR (q.v. Sect. 4.3.1), indistinguishability is oblivious to the prior and the adversary's knowledge and postprocessing activities, which means that it holds regardless of them, yet does not capture the actual privacy threat such unaccounted-for variables pose.

The second condition enforces a notion of privacy specific to a definition of, on the one hand, a query's *topic* and, on the other hand, a notion of distance between *topics* that represents how *"different"* topics are, i.e. implicitly requiring a certain distance $\ell(\mathbf{x}, \mathbf{y})$, with $\mathbf{x} = g(q_v)$. Such a constraint however does not necessarily translate into a guaranteed distance $\ell(\mathbf{x}, \hat{\mathbf{x}})$ that leads to profile confidentiality, as it disregards adversarial filtering and post-processing. In other words, enforcing a minimum distance $\ell(\mathbf{x}, \mathbf{y})$ does not necessarily lead to a minimum distance $\ell(\mathbf{x}, \hat{\mathbf{x}})$.

The third condition enforces a deniability bound, requiring that the probability that the user generates any query in query set S_q is the same, i.e. $P(\, q_j \,) = P(\, q_i \,)\ \forall q_i, q_j \in S_q$, so that, similarly to GooPIR, $\mathrm{N} = \frac{|S_q|-1}{|S_q|}$.

Similarly to GooPIR, the deniability condition implies that a CBPWS tool cannot generally enforce k-PD-Privacy. As we have examined in the case of GooPIR, deniability depends on assumptions about the prior probability of users' search activity sequences, so adversaries can exploit flawed assumptions about $P(\, \mathrm{R} = \mathrm{r} \,)$ that a CBPWS tool incorporates.

Lastly, the level of k in k-PD-Privacy conflates the interplay between the three conditions above, e.g. let us consider two CBPWS tools that satisfy k-PD-Privacy, one with high topic diversity, the other with low topic diversity; k does not account for the difference in topic diversity.

PDS's DGS. Similarly to GooPIR, since PDS can neither control the prior probabilities $P(\, \mathrm{R} = \mathrm{r} \,)$ nor the prior $P(\, \mathrm{R}_\beta = \mathrm{r} \,)$ adversaries relies on, it follows that it cannot generally satisfy k-PD-Privacy.

PDS's simultaneous submission and canonicalisation prevents an adversary from exploiting divergences between real queries' and dummy queries' timing and syntax, respectively, to distinguish them. Whereas simultaneous submission

forces dummy queries into real timing patterns, canonicalisation forces real queries into *"dummy querying"* syntax. Both represent sound piecemeal strategies to hamper an adversary's efforts to tell real and dummy queries apart.

PDS's assumptions in its PD-Queryset construction about what makes queries *diverse* (in terms of topic similarity) and *plausible* introduce however vulnerabilities an adversary can exploit and PDS disregards.

PDS defines topic similarity as the distance between two queries in a semantic space, assuming a function $g_{\mathcal{T}}$ that maps queries to the semantic space and a particular measure of semantic distance, namely, cosine similarity. Hence, queries are diverse if they are far from each other on the semantic space. Moreover, PDS ties plausibility to the semantic space by using the *relative density* of real queries in the semantic space around the neighbourhood of the canonical query they map to, assuming a large corpus of queries S_r is available to estimate such density. Thus queries that map to canonical queries with a high relative density (many real queries map to that canonical query) end up in a query set with other canonical queries whose neighbourhood in the semantic space is high density. We note that this notion of plausibility is a spin on GooPIR's understanding of plausibility by focusing not on individual queries' popularity, but groups of queries that, taken together, lead to a popular topic. This divergence across interpretations further highlights that designers adopt non-universal definitions of what makes a query *plausible*.

In fact, because the notion of plausibility PDS adopts is only one among many, the ability of PDS to ensure plausibility relies on the assumption that the adversary uses the same g as PDS, i.e. that $g = g_{\mathcal{T}}$. If the adversary uses a different SCA, the semantic relationships that PDS seeks to enforce may no longer hold, undermining the plausibility of dummy queries.

To illustrate this, consider the following example using a PDS system with $k = 2$, i.e. PDS generates one dummy query to accompany each real query. Let us say that a user issues the queries {*Justin Bieber*}, {*Toy Story*}, {*Disneyland*}, and that according to SCA_{PDS} the dominant topics of these queries are "music", "cartoons", and "amusement parks", respectively. Let us further say that, also according to SCA_{PDS}, PDS masks these categories with dummy queries about "history", "physics", and "cars", respectively. Now consider that the adversary implements a different SCA_{Adv} that classifies all three real queries above as being related to "entertainment for children", rather than individually associated to "music", "cartoons", and "amusement parks". Given SCA_{Adv}, it is apparent to the adversary that kids-related topics appear more often than others, hence that kids-related queries are probably the user's real queries.

Conclusion. PDS relies on *equally plausible*, *simultaneously sent*, *fixed sets* of dummy queries, yet by incorporating assumptions about which topics a query belongs to and what plausibility means, it underestimates the ability of an adversary to exploit those very assumptions to its advantage. Thus PDS highlights the lack of control designers have over g and the risks of relying on assumptions a CBPWS tool has no effective control over.

4.3.3 TrackMeNot 2.0

TrackMeNot (TMN) is a CBPWS tool designed by Howe and Nissembaum [294]. Implemented as a browser plugin for both Firefox and Chrome,[7] the first design of TMN [294] suffered from critical flaws that undermined its ability to meet its own privacy goals. Several authors, amongst whom ourselves, pointed out to or demonstrated an adversary's ability to filter out dummies from a user's TMN-obfuscated web search activity [17, 42, 430, 473]. As a response to such criticisms, Toubiana et al. released an improved version [523] —what we refer to as TMN 2.0— that we evaluate in the remainder of this section.

Privacy property and measure. The new version of TMN sets itself two objectives: query indistinguishability and side channel leakage prevention. Toubiana et al. say that a a CBPWS tool provides indistinguishability if:

$$\forall q \in \mathrm{q},\ P(\, q \notin \hat{\mathrm{r}} \mid q \in \mathrm{r}\,) = P(\, q \notin \hat{\mathrm{r}} \mid q \notin \mathrm{r}\,) = \rho \tag{4.11}$$

simplifying,

$$\forall q \in \mathrm{q},\ P(\, q \notin \hat{\mathrm{r}}\,) = \rho \tag{4.12}$$

Toubiana et al. define two variants of indistinguishability. On the one hand, *topic-exposed indistinguishability*, whereby a user's estimated profile $\hat{\mathbf{x}}$ has the same non-zero components as her real profile $\mathbf{x}$, formally:

$$\forall x_i \in \mathbf{x}, x_i = 0 \Rightarrow \hat{x}_i = 0 \tag{4.13}$$

On the other hand, *topic-obfuscated indistinguishability*, whereby a user's estimated profile $\hat{\mathbf{x}}$ has $n > m$ non-zero components, with m the number of non-zero components originally in $\mathbf{x}$, namely,

[7] The extension can be obtained at `https://cs.nyu.edu/trackmenot/`

$$\|\hat{\mathbf{x}}\|_0 - \|\mathbf{x}\|_0 = n - m \tag{4.14}$$

where $\|\cdot\|_0$ represents the L^0 "norm".

TMN's authors provide a set of guidelines towards side channel leakage prevention such as indistinguishability of query metadata and browser behaviour, as well as clicks on web search results. However, they do not formalise how to measure a CBPWS tool's ability to prevent side channel leakage.

Dummy generation strategy.

TMN assumes the search engine publishes a universe of topics T that it uses to classify user queries. According to this universe of topics T, each of a user's profile coordinates x_i represents the weight of a topic $\tau_i \in T$ in the user's profile. To generate dummy queries, TMN relies on the following elements:

Frequency profile. TMN computes a *frequency profile* of a user's search activity that consists of query frequency across topics, keyword frequency within topic and relative popularity of n-grams; TMN generates dummy queries that follow a similar frequency profile.

Timing profile. TMN analyses user search behaviour timing patterns, maintaining weekly and daily profiles to generate dummy queries that approximately replicate the timing and the inter-arrival times of users' genuine web search activity. Moreover, TMN only generates queries when the browser is active, yet does not weave dummies into a sequence of user queries in a topic. Instead, it generates a temporally correlated sequence of dummy queries in a target topic τ_i.

To provide topic-exposed indistinguishability, TMN uses public RSS feeds to generate queries that relate to topics $\tau_i \mid x_i > 0$.

To provide topic-obfuscated indistinguishability, TMN expands the frequency profile to identify long-term and time-varying interest topics, so that for each frequency range TMN generates dummy queries at a similar frequency pattern on at least one additional topic $\tau_i \mid x_i = 0$.

Evaluation.

Toubiana et al. argue that TMN does not attempt to provide robust privacy guarantees, that rather it seeks to protect against general bot detection tools [523,

566]. TMN's authors evaluate the ability of generic search bot detection tool to detect TMN queries, showing in their experiments that it fails. However, the bot specification they use in the experiments does not exploit any of TMN's design particularities, which means that TMN assumes a *naive* adversary that does not exploit all the information available about the tool and therefore does not even detect it.

Assuming a naive adversary does not justify the complexity of TMN's DGS, as a naive adversary may require TMN to implement little to no measures to achieve indistinguishability. Moreover, it offers little guarantees against future general bot detection tools that incorporate TMN's dummy generation patterns. Hence, we examine both TMN's DGS and the measures it implements towards tool undetectability (q.v. Sections 3.4.1 and 2.4.1).

TMN's indistinguishability. TMN adopts an attack-centred analysis (ACA) by focusing on adversarial classification performance. TMN's query indistinguishability requires that an adversary classifies real and dummy queries based on the dummy rate alone, i.e. so that the posterior probability that the adversary classifies any query as a dummy coincides with the a priori probability that any query is dummy. As in our previous evaluation of GooPIR and Plausibly deniable search (PDS), such a privacy goal requires a CBPWS to assume a particular distribution over the space of real query sequences $\mathbf{R}$ or a particular instance of adversarial knowledge.

TMN's two flavours of indistinguishability, topic-exposed and topic-obfuscated, further require mapping queries to topics. Hence, whereas Eq. 4.11 imposes a privacy requirement through a topic-agnostic measure, its realisations as topic-exposed and topic-obfuscated mandate additional assumptions about the profiling function g, thus representing a largely unrealisable requirement if we expect a CBPWS tool to provide resistance against arbitrary profiling functions g_i.

TMN's DGS. Toubiana et al. evaluate TMN against generic bot detection tools and argue that *"[TMN] can not be detected and no features let a search engine distinguish artificial and user queries"* [523]. Their evaluation however largely underestimates the ability of an adversary to detect TMN, as generic bot detection tools do not target the features that make TMN potentially detectable, e.g. they do not test TMN's detectability by training a classifier with both TMN and non-TMN web search activity samples. Hence, TMN's undetectability largely rests on the assumption that the adversary does not strive to detect it.

Moreover, Toubiana et al. do not evaluate TMN's ability to provide query indistinguishability, be it topic-exposed or topic-obfuscated. Rather, they test whether a search engine that publishes the user search profiles it builds incorporates TMN's dummy queries, concluding that since it does, they must be indistinguishable [523].

In light of such divergence between TMN's stated privacy goals and its evaluation, we briefly examine to what extent TMN's DGS can meet its stated design goals.

TMN suffers from the same vulnerability as PDS in that it assumes a mapping between queries and topics, yet does not consider the ability of an adversary to exploit alternative mappings that reveal relationships between a users' real queries TMN's SCA disregards.

Moreover, its DGS to provide topic-obfuscated indistinguishability fails to provide ρ indistinguishability *by design* because it partitions the set of real and dummy profile components in two: by replicating the frequency profile of a user's queries related to one particular topic $\tau_i \mid x_i > 0$, in another topic $\tau_j \mid x_j = 0$, TMN exposes that one of them is entirely dummy and the adversary entirely filters the dummy component x_j with probability $1/2$, regardless of ρ's value.

Lastly, as we discuss in Sect. 3.4.1, tool undetectability and profile confidentiality impose conflicting constraints on a CBPWS's DGS. On the one hand, tool undetectability requires that CBPWS's users exhibit similar activity patterns to those that do not use a CBPWS tool, thus severely limiting the rate ρ at which the tool can generate dummies, e.g. a user that generates many dummies may be detected as a bot, as no human can generate queries at such rate on so many different topics. On the other hand, a small rate ρ has an impact on the maximum indistinguishability level a CBPWS can attain, thus undermining profile confidentiality. Toubiana et al's evaluation of TMN disregards the trade-offs between tool undetectability and profile confidentiality.

Conclusion. TMN illustrates the consequences of discrepancies between a CBPWS tool's stated goals, its operationalisation in privacy measures and its DGS. Moreover, by assuming a naive adversary that does not strive to attack the tool, TMN's authors foreclose a thorough evaluation of TMN's resistance against attacks.

4.3.4 PRAW - A PRivAcy model for the Web.

Privacy model for the web (PRAW) is a CBPWS tool that has undergone several rounds of analysis and refinements [195, 196, 197, 198, 335, 485]. PRAW differs

from the CBPWS tools we have examined so far in that it considers as adversary the Internet service provider as opposed to the search engine. Moreover, PRAW considers that the adversary builds profiles out of the keywords on the sites that users' clicked query search results link to, as opposed to the keywords on users' search queries. Whereas we may argue that dummy queries are an integral part of a CBPWS tool's design, we also note that the other CBPWS tools we examine in this chapter disregard the generation of dummy clicks on search results, thus rendering dummy queries completely distinguishable from real queries. Hence, in our analysis and evaluation of PRAW we abstract away from dummy search query generation and focus on its DGS of clicks on search results, even if we acknowledge that a sound CBPWS tool must account for both dummy queries and clicks on search results (q.v. Sect. 4.4.4).

Privacy definition. PRAW defines privacy in web search as the *distance* between a user's real profile $\mathbf{x}$ and the obfuscated, observed profile $\mathbf{y}$.

Privacy measure. PRAW uses the *cosine similarity* between profiles as a measure of privacy; so that the less similar $\mathbf{x}$ and $\mathbf{y}$ are (greater distance), the smaller $S_C(\mathbf{x}, \mathbf{y})$ is and the less information $\mathbf{y}$ gives away about $\mathbf{x}$.

PRAW's authors refer to the similarity $S_C(\mathbf{x}, \mathbf{y})$ as the level of privacy *"from PRAW's point of view"* and suggest to use $S_C(\mathbf{x}, \hat{\mathbf{x}})$ as an *"external privacy measure, from an attacker's point of view, [...] assuming an attack on the system"* [198].

Dummy generation strategy. Concordant with PRAW's privacy definition, its DGS seeks to generate dummies that decrease the similarity $S_C(\mathbf{x}, \mathbf{y})$ to maximise the level of privacy.

PRAW relies on an SCA to build profiles $\mathbf{x}$ and $\mathbf{y}$ from the terms in each visited web page. PRAW measures the level of privacy protection as the distance between $\mathbf{x}$ and $\mathbf{y}$ and communicates this number to the user, so that she can choose to generate more dummy traffic to increase her level of privacy protection.

PRAW's DGS performs (on average) k dummy visits for each real visit, so that $\rho_{\text{visits}} \simeq {}^{k}/_{(1+k)}$, and generates dummy queries using *"a mix of terms, originating in* [the real user profile $\mathbf{x}$]*, along with random terms originating from an internal database of terms that is a glossary of terms related to the general domain of the user's interests"* [198]. The goal of this strategy is to obtain search results that relate to topics that are not too different from those the user is interested in to prevent an adversary from deploying clustering attacks that

distinguish real and dummy visits based on their topic [198]. PRAW's authors acknowledge that such a strategy may reveal users' broader interests, but argue that it is necessary to generate plausible dummy visits and that preventing the adversary from inferring specific topics of interest offers sufficient privacy protection, e.g. the adversary may learn about users' interest in computer security, but not about what exactly they are interested in within that domain, be it cryptography or malware detection.

Among the search results returned to each dummy query, PRAW clicks on a random subselection of them. Moreover, because users tend to follow hyperlinks within the pages they visit, PRAW similarly clicks on links to a random depth on dummy visits to boost indistinguishability between real and dummy visits. To undermine an adversary's ability to exploit time patterns and tell dummy visits apart, PRAW monitors user clicking patterns and simulates similar clicking behaviour in terms of both time between clicks and time spent browsing.

Evaluation.

Privacy as similarity. PRAW defines privacy *"from PRAW's point of view"* as the similarity $S_C(\mathbf{x}, \mathbf{y})$ and privacy *"from an attacker's point of view"* as the similarity $S_C(\mathbf{x}, \hat{\mathbf{x}})$. According to the terminology we introduce in Sect. 3.2, PRAW's authors implicitly refer to two types of measures or analyses: MCA and ACA, respectively. However, similarity or distance measures are unsuitable for MCA analysis because they implicitly assume a naive adversary that does not attack the CBPWS tool, namely, an adversary that observes $\mathbf{y}$ and does not attempt to filter it.

To illustrate why distances $\ell(\mathbf{x}, \mathbf{y})$ are inadequate measures of privacy, let us assume that PRAW relies on a DGS that chooses an observed profile $\mathbf{y}_i$ for every real profile $\mathbf{x}_i$ according to an optimisation function $f(S_C(\mathbf{x}, \mathbf{y}), \rho_{\text{visits}})$ that maximises the distance $\ell(\mathbf{x}, \mathbf{y})$ for a budget of dummy visits ρ_{visits}. Let us further assume that for each $\mathbf{x}_i$ there is a unique $\mathbf{y}_i$ that satisfies the condition above. This DGS fails to provide any privacy protection because even if similarity $S_C(\mathbf{x}_i, \mathbf{y}_i)$ is minimum (and therefore distance is maximum), a strategic adversary can revert the optimisation function $S_C(\mathbf{x}_i, \mathbf{y}_i)$ to retrieve the $\mathbf{x}_i$ that triggers each $\mathbf{y}_i$, as $\forall j \neq i,\ P(\,\mathbf{x}_j \mid \mathbf{y}_i\,) = 0 \iff P(\,\mathbf{x}_i \mid \mathbf{y}_i\,) = 1,\ j = i$. Therefore, this DGS does not offer *any* protection and the similarity $S_C(\mathbf{x}, \mathbf{y})$, even if minimum, is meaningless.

Conversely, as we explain in Sect. 3.2, distances $\ell(\mathbf{x}, \hat{\mathbf{x}})$ are appropriate measures of privacy *for particular adversaries and attacks*. If an adversary deploys a suboptimal attack or relies on imprecise or limited background knowledge, the distance $\ell(\mathbf{x}, \hat{\mathbf{x}})$ that measures how close to $\mathbf{x}$ the profiles $\hat{\mathbf{x}}$ the adversary

estimates are underestimates the ability of a more knowledgeable adversary to get closer to $\mathbf{x}$, as $\ell(\mathbf{x}, \hat{\mathbf{x}})$ does not capture all the information the CBPWS tool leaks. Moreover, the use of a distance $\ell(\mathbf{x}, \hat{\mathbf{x}})$ assumes a particular profiling function g and metric space that ℓ induces that do not necessarily match the adversary's, thus further overestimating the privacy level, as we have earlier discussed in the analyses of both PDS and TMN (q.v. Sections 4.3.2 and 4.3.3, respectively).

PRAW's authors argue that $S_C(\mathbf{x}, \hat{\mathbf{x}}) = 0$ represents zero similarity and therefore maximum privacy, whereas $S_C(\mathbf{x}, \hat{\mathbf{x}}) = 1$ represents maximum similarity and therefore minimum privacy. Yet the lower bound $S_C(\mathbf{x}, \hat{\mathbf{x}}) = 0$ fails to acknowledge that the expected distance between profiles $\mathbf{x}$ and $\hat{\mathbf{x}}$ bounds the maximum average level of privacy any CBPWS tool can attain.

To illustrate why $S_C(\mathbf{x}, \hat{\mathbf{x}}) = 0$ represents an unattainable goal, let us assume that PRAW relies on a DGS that, for each page the user visits, it generates a dummy visit to each and every other page on the internet.[8] Since all possible webpages are visited with every new user visit, this DGS provides perfect privacy: of all the pages the user could have visited, the DGS visits them all. As a result, the best attack strategy is no better than choosing user profiles based on prior knowledge alone, which means that the resulting distance $S_C(\mathbf{x}, \hat{\mathbf{x}})$ cannot be larger than the expected value of the distance between any two $\mathbf{x}$, namely, $\mathbf{E}[S_C(X, X)]$.

PRAW's DGS. PRAW's authors have evaluated PRAW's ability to withstand clustering attacks [198], finding that dummy queries are hard to filter based on their topic with reasonably low similarity $S_C(\mathbf{x}, \hat{\mathbf{x}})$. They thus conclude that PRAW provides an adequate level of privacy protection. Their evaluation however considers an adversary that uses clustering blindly, that does not strategically exploit PRAW's weaknesses, thereby underestimating a strategic adversary's ability to compromise PRAW's DGS.

Despite the fact that PRAW defines privacy as the distance between $\mathbf{x}$ and $\mathbf{y}$, its DGS does not strategically generate dummy visits to maximise that distance. In fact, PRAW's designers argue that observed profiles $\mathbf{y}$ should gravitate around the real profiles' general interests to prevent easy filtering of dummy visits, which introduces a contradictory constraint whose impact on privacy they do not assess. PRAW's DGS does not strategically attempt to maximise the distance between $\mathbf{x}$ and $\mathbf{y}$ within the general topics to which $\mathbf{x}$ relates either, which means that PRAW's DGS design principles are in contradiction with PRAW's privacy definition.

[8] More accurately, every other page on the Internet retrievable through queries to the search engine or linked by visited webpages.

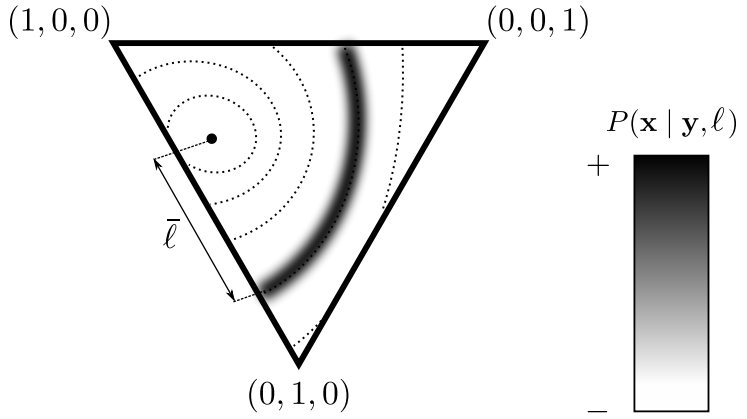


Figure 4.8: $P(\mathbf{x} \mid \mathbf{y}, \ell)$ assuming that $P(X = \mathbf{x})$ is uniform (e.g. due unavailable information).

Regardless, we note that PRAW's stated goal to attempt to generate observed profiles $\mathbf{y}$ as dissimilar as possible from real profiles $\mathbf{x}$ is a counterproductive strategy, since a strategic adversary can exploit information about the induced distances to better filter $\mathbf{y}$ into $\hat{\mathbf{x}}$.

Elovici et al. report that PRAW's strategy works in such a way that the similarity $S_C(\mathbf{x}, \mathbf{y})$ is a function of the dummy generation rate ρ_{visits}, e.g. generating 2 to 10 dummy visits per real visit results in similarities between 0.17 and 0.07, respectively [195]. Since it is possible to infer the dummy rate from the total number of visits [430], a PFA can exploit the correlation between $S_C(\mathbf{x}, \mathbf{y})$ and the dummy generation rate ρ_{visits} to significantly reduce uncertainty on $\mathbf{x}$.

Figure 4.8 shows the space $\mathcal{X}$ of possible profiles $\mathbf{x}$ when considering three categories or topics (vectors $\mathbf{x} = \{x_1, x_2, x_3\}$ are such that $\sum_i x_i = 1$). Let us consider that PRAW produces profile $\mathbf{y}$, which in Fig. 4.8 corresponds to the point marked as $\bullet$. Given PRAW's DGS, the real profile $\mathbf{x}$ that triggers observation $\mathbf{y}$ lies with high probability at distance $\bar{\ell}$ from $\mathbf{y}$, where $\bar{\ell}$ is the expected distance between profiles given ρ. In Fig. 4.8, we depict higher probability densities $P(\mathbf{x} \mid \mathbf{y}, \ell)$ in a darker shade, assuming that, a priori, any profile $\mathbf{x}_i$ is equally likely, i.e. $P(\mathbf{x}_i) = P(\mathbf{x}_j)$, $\forall \mathbf{x}_i, \mathbf{x}_j$. The set of candidate profiles $\mathbf{x}$ form a circle of radius $\bar{\ell}$ centred around $\mathbf{y}$ whose width is given by the variance of ℓ. PRAW's DGS leaks that the user's real profile most likely lies in these dark areas —thus significantly leaking information about $\mathbf{x}$, i.e. $\mathrm{H}(X \mid Y) < \mathrm{H}(X)$.

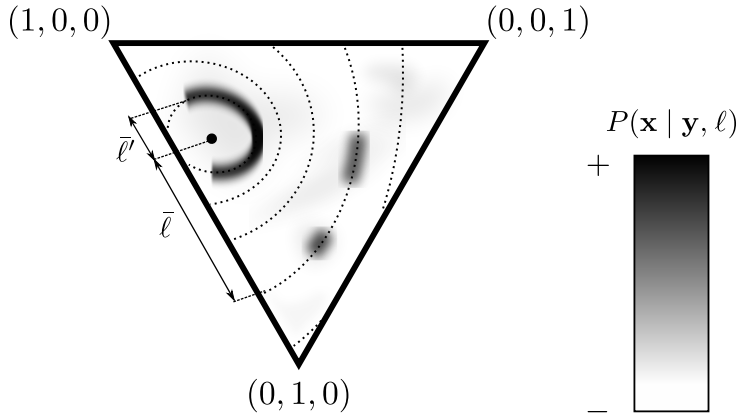


Figure 4.9: $P(\mathbf{x} \mid \mathbf{y}, \bar{\ell})$ and $P(\mathbf{x}' \mid \mathbf{y}, \bar{\ell}')$ assuming that $P(X = \mathbf{x})$ is as depicted in Fig. 4.4 (on page 121) and available to the adversary.

An adversary that knows the prior probability $P(X = \mathbf{x})$ can exploit PRAW's information leakage to improve its estimation $\hat{\mathbf{x}}$. Let us consider that Figure 4.4 (on page 121) depicts the prior probability $P(X = \mathbf{x})$. Using Bayes' theorem, the adversary computes the posterior probability $P(X \mid Y, \bar{\ell})$, with $\bar{\ell}$ the expected distance $\ell(\mathbf{x}, \mathbf{y})$ given ρ. This attack enables the adversary to narrow down the set of highly likely profiles to those $\mathbf{x}$ that are both reasonably common in the population *and* that lie at a distance $\bar{\ell}$ from the observed profile $\mathbf{y}$. Fig. 4.9 depicts the outcome of combining information about PRAW's DGS and background information on X, given two possible estimated distances $\bar{\ell}$ and $\bar{\ell}'$.

PRAW considers that privacy is proportional to distance (inversely proportional to cosine similarity), so that if $\bar{\ell}' < \bar{\ell}$ a DGS that enforces distance $\bar{\ell}$ provides a higher level of privacy than a DGS' enforcing $\bar{\ell}'$. We note that in the scenario Fig. 4.9 depicts, considering background information may result in $\bar{\ell}'$ corresponding to a higher level of uncertainty on $\mathbf{x}$ (larger dark surface) than $\bar{\ell}$; i.e. $\mathrm{H}(\mathbf{x} \mid \mathbf{y}, \bar{\ell}')$ higher than $\mathrm{H}(\hat{X} \mid \mathbf{x})$ although $\bar{\ell}' < \bar{\ell}$. This further illustrates that distance $\ell(\mathbf{x}, \mathbf{y})$ is not necessarily proportional to privacy and that a DGS that maximises a particular geometric distance leaks information about $\mathbf{x}$ that a strategic adversary exploits to produce a better estimation $\hat{\mathbf{x}}$.

Conclusion. PRAW measures the privacy it provides as the distance between the user's real profile $\mathbf{x}$ and the obfuscated profile $\mathbf{y}$. We demonstrate the inadequacy of equating such a distance with the level of privacy a CBPWS tool provides and show that a strategic adversary can precisely exploit induced distances to obtain estimated profiles $\hat{\mathbf{x}}$ so that $\ell(\mathbf{x}, \hat{\mathbf{x}}) < \ell(\mathbf{x}, \mathbf{y})$.

4.3.5 Optimized query forgery for private information retrieval

Rebollo-Monedero and Forné propose *"Optimized query forgery for private information retrieval"* (OQF-PIR) [448], a CBPWS DGS design that seeks to optimise profile confidentiality given a limited budget of dummy queries.

Similarly to PRAW, OQF-PIR seeks to hide user profiles rather than individual queries. However, OQF-PIR's underlying privacy definition differs from PRAW's in that it is a function of the distance between a user's observed profile and the population's average profile, rather than the distance between the observed profile and the original profile. Moreover, OQF-PIR seeks to optimally use the budget of dummy queries to maximise privacy protection, whereas PRAW's DGS does not.

Privacy definition. OQF-PIR's authors claim that *"whenever the user's [profile] differs from the population's, a privacy attacker will have actually gained some information about the user, in contrast to the statistics of the general population"* [448]. They define the *population's profile* $\mathbf{x}^p$ as the expected value of the search engine users' real profiles and assume that the number of OQF-PIR users is small enough for their impact on profile $\mathbf{x}^p$ to be negligible, i.e. $\mathbf{x}^p = \mathbb{E}[X]$.

Privacy measure. OQF-PIR's authors propose to measure the amount of information an adversary gains as the Kullback-Leibler divergence (KLD) between a user's observed profile $\mathbf{y}$ and the population's profile $\mathbf{x}^p$, i.e. $D_{\mathrm{KL}}(\,\mathbf{y}\,||\,\mathbf{x}^p\,)$ [140]. They interpret $D_{\mathrm{KL}}(\,\mathbf{y}\,||\,\mathbf{x}^p\,)$ as a measure of dissimilarity between the observed and population profiles or, more precisely, the adversary's *information gain* about the user's profile from observing $\mathbf{y}$ instead of the population profile $\mathbf{x}^p$ (cf. Sect. 3.2.2). Hence they consider that to attain perfect privacy the adversary must learn *nothing* about the user profile, namely, according to their privacy measure, that $D_{\mathrm{KL}}(\,\mathbf{y}\,||\,\mathbf{x}^p\,) = 0 \iff \mathbf{y} = \mathbf{x}^p$.

Dummy generation strategy. OQF-PIR's goal is to optimally minimise $D_{\mathrm{KL}}(\,\mathbf{y}\,||\,\mathbf{x}^p\,)$. OQF-PIR models the observed profile $\mathbf{y}$ as a weighted function of the real profile $\mathbf{x}$ and a dummy profile $\mathbf{w}$:

$$\mathbf{y} = (1-\rho)\mathbf{x} + \rho\mathbf{w} \tag{4.15}$$

where $\mathbf{w}$ is a multinomial distribution whose elements w_i represent the fraction of dummy queries in category i the DGS must generate. OQF-PIR implicitly

assumes an $\mathrm{SCA}_{\mathrm{OQF}}$ that identifies query topics to construct the profiles $\mathbf{x}$, $\mathbf{w}$ and $\mathbf{y}$. The weighting factor ρ represents the dummy rate so that for a given real profile $\mathbf{x}$, the optimal dummy profile $\mathbf{w}$ minimises $D_{\mathrm{KL}}(\,\mathbf{y}\,||\,\mathbf{y}^t\,)$.

OQF-PIR's optimisation algorithm first orders the profile categories such that

$$\frac{x_1}{y_1^t} \leq \cdots \leq \frac{x_i}{y_i^t} \leq \cdots \leq \frac{x_n}{y_n^t}\,, \tag{4.16}$$

where the last component $\frac{x_n}{y_n^t}$ leads to *critical redundancy* $\rho_{\mathrm{crit}} = 1 - \frac{y_n^t}{x_n}$, namely, the minimum budget of dummy queries OQF-PIR requires to provide perfect profile confidentiality.

OQF-PIR allocates the budget of dummy queries according to a *water-filling* algorithm [225], i.e. it adds dummies to the *'deepest'* components $\mathbf{w}_i$ first until the budget is exhausted.

To illustrate how the water-filling strategy works, let us consider the following user and population's profiles (whose components are already ordered according to Eq. 4.16):

$$\mathbf{x} = (0.15,\, 0.25,\, 0.1,\, 0.3,\, 0.2)$$

$$\mathbf{x}^p = (0.3,\, 0.3,\, 0.1,\, 0.2,\, 0.1)$$

It follows that $\rho_{\mathrm{crit}} = 1 - \frac{y_n^t}{x_n} = 1 - \frac{0.1}{0.2} = 0.5$, so the optimal $\mathbf{w}$:

$$\mathbf{w}_{\mathrm{crit}} = \frac{\mathbf{x}^p - \mathbf{x}}{\rho_{\mathrm{crit}}} + \mathbf{x} = (0.45,\, 0.35,\, 0.1,\, 0.1,\, 0)$$

Figure 4.10 depicts how the water-filling algorithm underlying OQF-PIR's design allocates the budget of dummy queries in this particular example. For budgets of dummy queries below critical redundancy, the water-filling algorithm fills the *'deepest'* components first. Hence, in this particular scenario, OQF-PIR assigns the first dummy queries to the first category until it is *"filled"* to the level of the second. Then OQF-PIR assigns dummy queries both to the first and second categories until both are at the level of the third and fourth, filling each level until it exhausts the budget of dummy queries.

Evaluation

Privacy as deviation from the average. OQF-PIR's definition of privacy implicitly assumes an adversary that before it observes any search activity

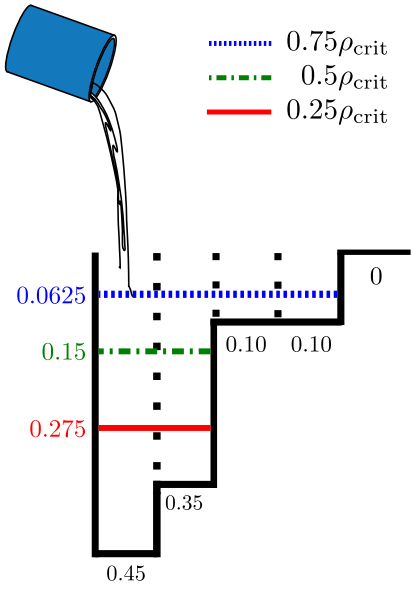


Figure 4.10: OQF-PIR's DGS allocates budgets of dummy queries below critical redundancy ρ_{crit} according to a water-filling strategy.

from a user, it assigns to her a candidate profile based on prior information alone, namely, $\mathbf{x}^p$. OQF-PIR's authors consider that if after observing a user's (obfuscated) sequence of search activity q the observation contradicts the adversary's prior, the CBPWS tool enables the adversary to gain some knowledge; if on the contrary the observation matches the adversary's prior, the CBPWS tool leaks no information to the adversary. The measure of *information gain* we define in Sect. 3.2.2 operationalises this definition of privacy protection.

Moreover, OQF-PIR assumes the adversary knows the prior $P(\mathbf{R} = \mathrm{r})$ so that $P_\beta(\mathbf{R} = \mathrm{r}) = P(\mathbf{R} = \mathrm{r})$. Since OQF-PIR's authors consider the search engine provider as adversary, they reason that the information the search engine provider has from all users enables it to derive the prior probability distribution $P(\mathbf{R} = \mathrm{r})$ and therefore $P(X = \mathbf{x})$.

OQF-PIR's authors assume the adversary selects $\mathbf{x}^p = \mathbb{E}[X]$ as *a priori* guess, yet $\mathbb{E}[X]$ represents an average profile that no user may have (e.g. the expected value rolling a six-sided die is 3.5, yet no side of the die has that particular value). Moreover, whereas we may argue that by choosing $\mathbf{x}^p = \mathbb{E}[X]$ the adversary minimises, *a priori*, its expected estimation error, it still represents a particular attack strategy, thereby undermining the generality of OQF-PIR's privacy

definition, as it does not capture the privacy threat an adversary with more or less knowledge, or an alternative *"guessing strategy"*, poses to a CBPWS tool.

Furthermore, we note that according to this metric a user whose profile coincides with the population's average (i.e. $\mathbf{x} = \mathbf{x}^p$) enjoys perfect privacy protection even if she does not use a CBPWS tool, implying that only users who *deviate* from the average need privacy protection.

OQF-PIR's measure of privacy. We argue that $D_{\mathrm{KL}}(\,\mathbf{y}\,||\,\mathbf{x}^p\,)$ does not appropriately operationalise OQF-PIR's privacy definition. Beyond the limitations of assuming the adversary's a priori guess is the average population's profile $\mathbf{x}^p = \mathbb{E}[X]$, the measure $D_{\mathrm{KL}}(\,\mathbf{y}\,||\,\mathbf{x}^p\,)$ implicitly assumes a naive adversary that does not attack OQF-PIR, that does not filter $\mathbf{y}$ and takes it as the best approximation of $\mathbf{x}$. In other words, OQF-PIR assumes that $\hat{\mathbf{x}} = \mathbf{y}$ and proposes $D_{\mathrm{KL}}(\,\mathbf{y}\,||\,\mathbf{x}^p\,)$ as a measure instead of $D_{\mathrm{KL}}(\,\hat{\mathbf{x}}\,||\,\mathbf{x}^p\,)$, which actually captures the information gain in retrieving $\hat{\mathbf{x}}$ as opposed to $\mathbf{x}^p$.[9]

OQF-PIR's DGS. OQF-PIR assumes a naive adversary, thus implicitly conflating $\hat{\mathbf{x}} = \mathbf{y}$. However, a strategic adversary can exploit OQF-PIR's *deterministic* water-filling algorithm and target profile $\mathbf{y}^t = \mathbf{x}^p$ to obtain a estimated $\hat{\mathbf{x}}$ closer to $\mathbf{x}$ than $\mathbf{y}$ is. We show how an adversarial DCA can identify (some of the) real queries and a PFA yield a better estimate $\hat{\mathbf{x}}$.

DCA-based attack. Let us consider an observed profile $\mathbf{y} = g(\mathrm{q})$ such that its k last components y_i have bigger values than $\mathbf{x}^p$ (i.e. $x_i^p < y_i$, for $n-k < i \leq n$), and let T denote the set of categories $T = \{\tau_i\}_{n-k<i\leq n}$. The water-filling mechanism OQF-PIR implements does not generate any queries on those k categories, as they would take $\mathbf{y}$ further from, rather than closer to, the target profile $\mathbf{y}^t$. A DCA can exploit this to identify queries q_T that according to $\mathrm{SCA}_{\mathrm{OQF}}$ relate to topics in set T and classify them as real. Thus, these queries enjoy neither undetectability nor deniability, as $P(\,\mathrm{q}|q_T \in \mathbf{R}_d^c\,) = 0$.

PFA-based attack. OQF-PIR assumes that the dummy rate ρ is a secret parameter; however, it is possible to estimate a rate $\hat{\rho} \simeq \rho$ from the overall number of queries and OQF-PIR's default configuration parameters. To

[9] We note that $D_{\mathrm{KL}}(\,\hat{\mathbf{x}}\,||\,\mathbf{x}^p\,)$ implicitly assumes a deterministic adversary that retrieves $\hat{\mathbf{x}}$ with probability $P_\beta(\hat{\mathbf{x}}) = 1$. To account for a probabilistic adversary that considers a range of possible $\hat{\mathbf{x}}_i$, we may instantiate Eq. 3.15 to measure the amount of information $\mathcal{G}$ an adversary gains about $\mathbf{x}$ from observing $\mathbf{y}$ compared to guessing $\mathbf{x}$ based on the prior alone. Formally,

$$\mathcal{G} = \log(P_\beta(\mathbf{x}|\mathbf{y})) - \log(P_\beta(\mathbf{x})) \tag{4.17}$$

with $P_\beta(\mathbf{x}) = P(X = \mathbf{x})$ assuming the adversary has perfect information about the prior. Moreover, Eq. 4.17 does not impose $\mathbf{x}^p$ as the adversary's a priori candidate profile.

illustrate how an adversary exploits an estimated $\hat{\rho}$, let us consider a three-dimensional profile space formed by categories or topics (τ_1, τ_2, τ_3), and a population profile that lies at the centre of the space, i.e. at point $\mathbf{y}^t = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$. Figure 4.11 represents this scenario.

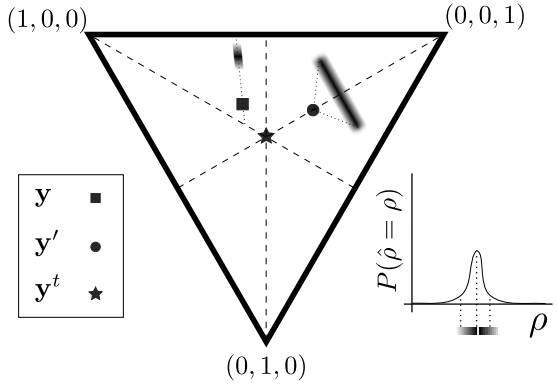


Figure 4.11: Exploiting distance ρ to attack OQF-PIR. Observed profiles, target profile and estimated real profiles as a function of ρ in the profile space.

Through its water-filling algorithm, OQF-PIR generates dummies in a deterministic way. Let us say an adversary observes profile $\mathbf{y}$, that we represent as a square dot in Fig. 4.11. The components of $\mathbf{y}$ are such that $y_{\tau_2} < y_{\tau_3} < y_{\tau_1}$; the gap between the two smallest components (y_{τ_2} and y_{τ_3}) indicates that ρ is too small to *fill* the smallest component (y_{τ_2}). The DGS must thus have generated dummies with a vector $\mathbf{w} = (w_{\tau_1}, w_{\tau_2}, w_{\tau_3}) = (0, 1, 0)$ and it is possible to estimate profile $\mathbf{x}$ as:

$$\hat{\mathbf{x}} = \left(\frac{y_{\tau_1}}{1-\hat{\rho}}, \frac{y_{\tau_2}-\hat{\rho}}{1-\hat{\rho}}, \frac{y_{\tau_3}}{1-\hat{\rho}}\right).$$

We note that $\hat{\rho} \to \rho \Longrightarrow \hat{\mathbf{x}} \to \mathbf{x}$, meaning that the adversary retrieves $\hat{\mathbf{x}} = \mathbf{x}$ if it accurately estimates ρ.

Figure 4.11 depicts as a dark (vertical) short line the likely profiles $\mathbf{x}$ that OQF-PIR might have obfuscated into $\mathbf{y}$. Even when the estimation of ρ has low confidence, the set of likely $\mathbf{x}$ is rather limited.

The point we mark as • in Fig. 4.11 corresponds to another possible observation $\mathbf{y}' = (y'_{\tau_1}, y'_{\tau_2}, y'_{\tau_3})$ such that $y'_{\tau_1} = y'_{\tau_2} < y'_{\tau_3}$. In this case, the DGS generates enough dummies to fill the weakest category (either τ_1 or τ_2), but not enough to bring $\mathbf{y}$ to $\mathbf{y}^t$. Hence, $\mathbf{w}' = (w'_{\tau_1}, w'_{\tau_2}, 0)$ with $w'_{\tau_1} + w'_{\tau_2} = 1$ and $\hat{x}'_{\tau_3} = y'_{\tau_3}/(1-\hat{\rho})$. A dark diagonal line in the upper right corner of the profile space in Fig. 4.11

represents the space of likely real profiles $\mathbf{x}'$. While $\mathbf{y}'$ leaves some room for uncertainty, the set of likely real profiles $\mathbf{x}'$ is still rather limited.

Lastly, Figure 4.12 represents a scenario in which the dummy rate ρ is sufficient to achieve $\mathbf{y} = \mathbf{y}^t$. A dark inner triangle represents the space of likely profiles $\mathbf{x}$ that OQF-PIR obfuscates into $\mathbf{y} = \mathbf{y}^t$ given $\hat{\rho}$. Even in this case OQF-PIR fails to provide a high level of profile protection. Moreover, an adversary may exploit background information to further reduce her uncertainty on $\mathbf{x}$, similarly to the attack on PRAW we examine in Sect. 4.3.4; yet according to OQF-PIR's definition of privacy, a user whose $\mathbf{y} = \mathbf{y}^t$ enjoys perfect privacy protection.

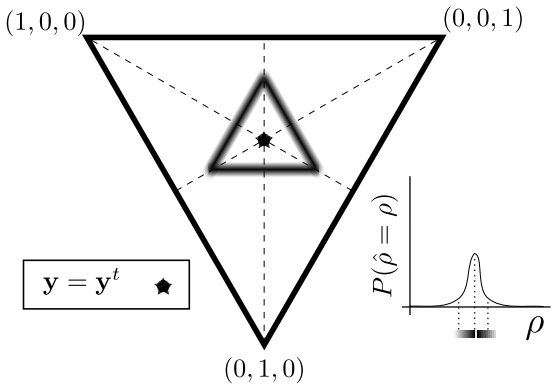


Figure 4.12: Probability of $\mathbf{x}$ given $\mathbf{y} = \mathbf{y}^t$ and dummy rate equals ρ.

Conclusion. OQF-PIR's privacy definition and DGS further illustrate the inadequacy of equating a CBPWS tool's privacy protection with a distance $\ell(\mathbf{x}, \mathbf{y})$ rather than $\ell(\mathbf{x}, \hat{\mathbf{x}})$, as that disregards a strategic adversary's ability to filter out dummies and obtain a better estimation $\hat{\mathbf{x}}$. Moreover, OQF-PIR's underlying water-filling algorithm demonstrates the vulnerabilities *deterministic* DGSs introduce.

4.3.6 Other tools

Bucket. Pang et al. propose *"Bucket"*, a tool that generates dummy search queries to protect user privacy in web search [426]. Whereas not strictly a CBPWS tool, as it relies on the search engine to filter the dummy results from users and therefore requires the service provider's cooperation, Bucket implements a DGS that shares many similarities with GooPIR's and PDS's. Bucket relies on query expansion by adding dummy keywords to the actual

user search query to prevent sending individual real queries and dummy queries. Moreover, it accounts for query *specificity* (analogously to GooPIR's *popularity* or PDS's *plausibility*) and distance on a semantic space (like PDS's *diversity*) to prevent the adversary from exploiting semantics and the prior probability of search queries. Unlike other tools, in its CBPWS model Bucket does account for sequences of queries rather than individual queries alone, considering that the adversary identifies the most probable sequence of queries as real. Moreover, it also acknowledges that an adversary's prior beliefs $P_\beta(\mathrm{r})$ are *"not known in advance and may vary among different adversaries"* as well as that *"quantifying the semantic similarity between two query sequences is an open problem, due to the correlations among the terms across queries."*

However, like other CBPWS tools we have examined so far, Bucket disregards a strategic adversary that exploits all the available information about the tool to undermine obfuscation; Bucket implicitly assumes its profiling function $g_\mathcal{T}$ and underlying semantic classification algorithm SCA match the adversary's, or that the adversary cannot exploit alternative SCAs to undermine query indistinguishability. Furthermore, Bucket neither formalises nor operationalises the privacy property it is after, considering that ensuring semantic diversity within querysets and similar specificity across a real query's accompanying dummies suffices to guarantee (an undetermined level of) user's privacy.

TopPriv To dispense with Bucket's service provider cooperation requirement, Pang et al. propose TopPriv, a CBPWS tool that, similarly to OQF-PIR, seeks to limit the amount of information adversaries gain on users' search interests [425]. TopPriv proposes (ϵ_1, ϵ_2)-privacy, where both ϵ_1 and ϵ_2 capture changes in the adversary's beliefs about a users' interest in a topic. TopPriv considers that the adversary has a prior belief $P_\beta(\mathbf{x}^p)$, which updates with each query r into $P_\beta(x_i \mid r)$ for each topic category i its $\mathrm{SCA}_{\mathrm{TopPriv}}$ considers. If the change in probability $P_\beta(x_i \mid q) - P_\beta(x_i^p)$ exceeds ϵ_1, TopPriv says that user query r reveals the user's interests; hence, TopPriv generates k dummy queries so that $P_\beta(x_i \mid \mathrm{q}) - P_\beta(x_i^p) \leq \epsilon_2$. TopPriv's DGS therefore selects dummy queries to manipulate the adversarial observation so that real queries' topics fall below the ϵ_2 threshold and the adversary assigns dummy queries' topics, which conversely exceed the ϵ_1 topics, as the users' actual interests.

TopPriv's (ϵ_1, ϵ_2)-privacy is analogous to information gain in that it measures changes in an adversary's beliefs about a user's profile $\mathbf{x}$. However, TopPriv suffers from the same vulnerability as previous proposals we have examined earlier: it considers a non-strategic adversary that does not attempt to filter q, in this particular case by exploiting the fact that TopPriv's DGS does not

generate any dummy queries that increase $P_\beta(x_i \mid \mathrm{q}) - P_\beta(x_i^p)$ beyond ϵ_2. Wang et al. demonstrate the feasibility of such an attack [542].

HDGA. Wang et al. propose a new DGS, HDGA, to counter the attacks they demonstrate on TopPriv [542]. Wang et al. provide no definition of privacy and argue that *"HDGA eschews the use of security metrics, other than the number of dummy topics. Even carefully-chosen security metrics can actually decrease security"*, mistakenly identifying privacy measures as the culprit of previous flawed DGS designs, rather than the disregard of strategic adversaries that filter $\mathbf{y}$ into $\hat{\mathbf{x}}$. While seemingly aware of the threat strategic adversaries pose to CBPWS tools, they do not subject HDGA to strategic attacks, assuming that *"[d]ummy queries are semantically coherent, so the adversary cannot identify dummy queries by exploring query semantics"*, yet HDGA's underlying definition of semantic coherence rests on assumptions about $\mathrm{SCA}_{\mathrm{HDGA}}$ that a strategic adversary can exploit, as we have illustrated throughout this section.

Degeling and Herrmann Degeling and Herrmann propose a DGS against online tracking by Google [167]. Not a CBPWS tool per se, as it does not generate dummy queries to the search engine, Degeling and Herrmann's proposal focuses on Google's ability to track users online, attempting to obfuscate the user profiles Google builds and lets users control and review.

Degeling and Herrmann explicitly state that their proposal does not consider a strategic adversary, their main goal being to assist *"understanding profiling and [foster] a privacy literacy that is necessary for users to stay autonomous in the information society"*. They study several strategies to manipulate the profiles that Google builds and lets users control and review, even if they acknowledge from previous work by Datta et al. that *"Google is not reporting all information that is used in a profile"* [167]. Whereas we acknowledge the role that feedback and transparency technologies have in promoting users' understanding of technology and its impact on their privacy, we question to what extent users can gain further understanding on profiling from a DGS that they neither have any control over nor may understand how it works. Degeling and Herrmann do not evaluate their proposal's ability to foster privacy literacy.

Lastly, Degeling and Hermann note that the profiling function g an adversary uses to compute a user's profile $\mathbf{x}$ may not only take as input a user's real queries or visits r, but also other users' interests, i.e. considering queries or sites of users with whom a user's behaviour overlaps, inducing in turn correlations across users' profiles. Moreover, they also observe that Google builds different profiles $\mathbf{y}$ for the same sequence of visits q over time, thus hinting an evolving profiling function g. None of the CBPWS tools we examine in this chapter

consider *network* of *temporal* effects in the profiling function, further casting doubt over the adequacy of relying on fixed assumptions about an adversary's profiling function g and semantic classification algorithm (SCA).

IQP. Ahmad et al. propose "Intent-aware query-obfuscation for privacy-protection" (IQP), an CBPWS tool that, similarly to GooPIR, extends the original query with $k-1$ dummy queries [16]. IQP attempts to preserve user utility and personalisation by building the user profile $\mathbf{x}$ and using it to filter dummy queries and rerank search results accordingly. Ahmad et al. claim IQP's DGS is inspired by *Bayes-optimal privacy* [364], which is analogous to the information gain measure we define in Sect. 3.2.2, yet do not use it in their evaluation, defining two alternative privacy measures instead, a *confusion index* (cIndex) and a *transition index* (tIndex), both essentially measures of *distance* $\ell(\mathbf{x}, \mathbf{y})$, between a user's real and observed profiles.

We note that IQP's design suffers from similar flaws as previous CBPWS tools. It assumes a naive adversary that does not attempt or is unable to filter dummy queries, thereby assuming $\hat{\mathbf{x}} = \mathbf{y}$. Ahmad et al. therefore rely on $\ell(\mathbf{x}, \mathbf{y})$ as a measure of privacy, instead of the distance $\ell(\mathbf{x}, \hat{\mathbf{x}})$ at which a strategic adversary estimates profiles $\hat{\mathbf{x}}$. In particular, IQP's evaluation disregards the fact that a strategic adversary can exploit the following elements of IQP's DGS's cover (i.e. dummy) topic sampling:

Avoidance of similar topics. IQP selects *"only a fraction of cover topics from [a node in a topic tree's] sibling nodes, the rest [...] randomly sampled from the non-sibling nodes of similar prior probability"*. [16]

Enforcement of similar specificity. IQP dynamically adjusts similar topic avoidance according to the generality of the true user query, *"if it is more specific[, it selects] fewer cover topics from its sibling nodes, as they [share] more common ancestors[,] and vice versa"* [16].

Replication of intent transition patterns. IQP attempts to replicate topic transitions in user queries by generating dummy queries following similar transitions. However, Ahmad et al. note that *"if they cannot follow the detected transition on the intent tree[,] we [keep] them intact with probability β, otherwise we use rejection sampling [to] select a new cover topic"* [16].

These features introduce differential treatment of real and cover topics and therefore represent a vulnerability in IQP's DGS design; however, since Ahmad et al. implicitly assume a naive adversary, they do not examine their impact, thus rendering IQP's evaluation flawed. In addition to the above, we note that

IQP, similar to previous designs, neither evaluates the impact of an adversary that exploits an $\mathrm{SCA}_{\mathcal{A}} \neq \mathrm{SCA}_{\mathrm{IQP}}$.

4.4 Discussion, challenges and open problems

In this section we summarise and discuss the main findings in our evaluation of CBPWS tools, pointing to limitations and further challenges in CBPWS's analysis and design.

4.4.1 Privacy requirements

In our analysis framework we introduce three privacy properties in an attempt to capture a variety of privacy requirements CBPWS designers have previously considered. We do not consider however that this set of privacy properties must be the ultimate goal of every CBPWS tool. Designers must select which privacy properties a CBPWS tool must satisfy according to users' privacy concerns and constraints, e.g. users may be unconcerned about disclosing their general interests while wishing to blur particular details about their web search activities; users may also be unwilling to increase the risk of targeting based on sensitive dummy queries, adding a specific constraint to CBPWS design [294]. Moreover, perfect CBPWS seems impractical to achieve in reality and it is unclear whether all users and applications require it, prompting designers to define less demanding privacy requirements and accept trade-offs between what is ideally desirable and what is practically possible.

CBPWS is inadequate to address certain privacy requirements, e.g. it cannot prevent an adversary from raising a flag if the user submits a particular query, even when the probability of such a query being dummy is high. In this particular scenario, CBPWS requires a *critical mass* of users that raises the cost of false positives to the adversary, namely, forcing the adversary to devote an increasing amount of resources to deal with an increasing number of faulty flags that threaten the integrity of the flagging process. Hence, users can no longer rely on CBPWS tools to unilaterally prevent targeting, as they require a cooperative pool of users.

In addition, rather than voicing privacy concerns that relate to their own personal circumstances, users may seek to prevent the service provider from exploiting their data in ways that go beyond the legitimate processing web search provision requires, regardless of what that additional processing entails, e.g. users may oppose the monetisation of their digital labour independently of

the privacy risks they perceive in profiling [220]. Yet the opacity of adversarial profiling poses a challenge to achieve this goal, as CBPWS designers lack the necessary information to design DGSs that target the profiling practices users wish to stop [160].

4.4.2 Privacy measures

Through our evaluation of CBPWS tools in the previous section, we observe that the privacy requirements operationalisation process is often fraught with errors. Existing CBPWS tool designs conflate mechanism-centred analysis with attack-centred analysis, requiring that CBPWS tools generally satisfy a definition of privacy that depends on adversarial performance and knowledge. In other words, CBPWS tool designers impose ACA measures as general design constraints without acknowledging that this type of measures incorporate particular assumptions about adversarial knowledge and attack strategies. In particular, using a distance measure $\ell(\mathbf{x}, \mathbf{y})$ between users' real profiles $\mathbf{x}$ and their corresponding obfuscated profiles $\mathbf{y}$ implicitly assumes a naive adversary that does not filter $\mathbf{y}$ into a filtered profile $\hat{\mathbf{x}}$, thereby overestimating the level of protection a CBPWS tool affords.

Our own set of CBPWS measures merits further discussion.

Information leakage. Information leakage provides an average measurement of the privacy protection a CBPWS tool affords, thereby conflating high and low privacy protection levels across users. As such, ensuring a particular level of information leakage does not guarantee a minimum level of privacy protection to all users. CBPWS designers may alternatively consider complementary measures that capture a *worst-case* privacy level e.g. min-entropy leakage (q.v. Eq. 3.13 in Sect. 3.2.1).

On deniability and undetectability. Both deniability and undetectability depend on the adversary's attack strategy. Deniability is a meaningful property only as long as users have the opportunity to contest claims the adversary makes about their activities, e.g. if users' web search activity become evidence in a trial, they can attempt to *deny* having issued those queries, claiming them to result from the CBPWS tool's DGS instead, and that such dummy queries bear no relation to their own search queries. However, if users do not have the chance to contest the legitimacy of those decisions that, based on their profile, affect them, then deniability becomes meaningless.

Undetectability on the other hand *requires* the adversary to be strategic. This means that the adversary must filter out queries as dummies; only through filtering can users have their real queries discarded, thus *undetectable*.

A strategic adversary deploying an optimal attack should classify dummy queries with probability equal to the dummy rate $\hat{\rho}$, i.e. to find the exact number of dummies the DGS generates. Hence in practice the maximum level of deniability depends on the adversary's estimation of the probability of a query being dummy $P(\rho)$ or the ability of Alice to plausibly claim a more advantageous ρ. For a given adversary and attack, we can expect that a higher value $\hat{\rho} > \rho$ results in higher deniability. Deniability similarly increases if Alice can claim that the DGS generates queries at a rate $\rho' > \rho$. Conversely, the maximum level of deniability may drop if the adversary estimates a probability $\hat{\rho} < P(d)$ and Alice cannot contest the accuracy of the estimation.

Similarly to deniability, the actual level of undetectability on specific, particular cases depends on the adversary's estimation of the rate at which dummies are generated. This means that for certain users undetectability may surpass its maximum expected level if the adversary's DCA classifies users' real queries as dummies at a rate $\hat{\rho} > \rho$.

Lastly, we acknowledge that using CBPWS measures in practice poses several challenges. It is far from trivial to estimate the probability distribution $P(\mathrm{R})$, especially for CBPWS designers that, unlike web search providers, do not have access to large datasets of search queries. Moreover, a strategic adversary does not rely on a unique probability distribution $P(\mathrm{R})$; rather, it exploits additional information to segment the target users and exploit their own subpopulation's probability distributions $P_{\text{subpop}}(\mathrm{R} = \mathrm{r})$, considering elements such as age, language or socioeconomic status. Similarly, designers may be hardly able to account for sporadic trends and changes in $P(\mathrm{R})$, namely, to consider a dynamic, evolving distribution instead of assuming it to be static. The complexity of estimating $P(\mathrm{R})$ thus represents a major challenge when using the measures we introduce in Sect. 4.2.2.

4.4.3 Adversarial assumptions

The CBPWS tools we have evaluated implicitly assume a naive adversary that processes observed profiles $\mathbf{y}$ as if they were the users' real profiles $\mathbf{x}$. They do not however justify how users benefit from an adversary processing $\mathbf{y}$ instead of $\mathbf{x}$, i.e. they do not assess or question the impact of assuming a naive adversary. In fact, it is unclear that profiling based on an obfuscated sequence of search activity q has any advantages over the real sequence of actions r.

Several of the CBPWS tools we have evaluated base their DGS on a particular profiling function $g_{\mathcal{T}}$ and evaluate the privacy protection they offer assuming that the adversary uses the same semantic classification algorithm. We have shown that strategic adversaries can exploit a different $g \neq g_{\mathcal{T}}$ to attack the DGS

and undermine query indistinguishability. Designing DGSs that withstand such an attack is a hard problem, as it is very difficult to account for every profiling function the adversary may exploit. Ye et al. have previously acknowledged this problem, alerting of the negative consequences it can have on CBPWS [563].

CBPWS tools require adversarial service providers to be honest-but-curious (HbC). However, the conditions under which service providers remain HbC require a delicate balance of incentives that may seldom hold in practice. If few users rely on CBPWS tools, their impact on search engines' profiling practices is negligible and search engines can dismiss them as background noise in their operations, effectively behaving as *naive* adversaries. This means that CBPWS users do not benefit from either deniability or undetectability, as the adversary treats all their queries as real. Moreover, they must accept whichever consequences profiling based on the observed profile $\mathbf{y}$ has on them, yet CBPWS tools do not optimise for profiling outcomes —disregarding in fact the consequences of profiling based on either $\mathbf{y}$ or $\hat{\mathbf{x}}$— because their ultimate goal is to thwart the profiling process altogether.

As the number of CBPWS users increases towards a critical mass that disrupts profiling practices, the search engine faces several choices. It can attempt to dissuade users from obfuscating their profiles, it can attempt to filter the dummies from users' profiles or it can entirely discard obfuscated profiles and remove their data from the profiling process. Search engines can ban obfuscation in their terms of service and subject CBPWS users to captchas [295], thereby hindering or outright preventing them from obfuscating their search activity whenever the dummy rate ρ exceeds a given threshold. Since ρ correlates with the level of privacy protection, search engines can enable low dummy rates with negligible impact on profiling while preventing users from deploying higher, more robust dummy rates, insidiously nudging CBPWS users to lower their protection. CBPWS users may however have the power to force search engines to consent to the use of obfuscation, in which case search engines may either attempt to filter dummies or entirely remove CBPWS users' data. Either of these two alternatives represents a positive outcome for CBPWS, the former more likely when CBPWS are vulnerable to attack.

On tool undetectability. Proposals like OQF-PIR [448] and TopPriv [425] implicitly assume that the adversary is unaware of the CBPWS tool, i.e. that the tool is *undetectable*, while TrackMeNot 2.0 explicitly lists undetectability as a requirement. We recall from our discussion in Sect. 3.4.1 that ensuring tool undetectability involves a particular set of design requirements that guarantee that the adversary cannot distinguish CBPWS users from non-users. None of the proposals that implicitly or explicitly rely on tool undetectability however

perform an evaluation of *how undetectable* their tool is under a strategic adversary. Neither do they discuss the consequences of assuming a naive adversary that processes **y** instead of **x** for the user. Howe and Nissenbaum report this user concern when they first introduce TrackMeNot [294]; whenever dummy queries relate to controversial topics, e.g. "bomb", "HIV", or "gay marriage", an undetectable CBPWS tool may prompt the profiler to classify users as involved in subversive activities, suffering a particular disease or having a certain sexual orientation. On the other hand, the opposite strategy (avoiding such keywords in dummy queries) puts users in a delicate position: either they expose themselves or they refrain from issuing queries related to sensitive topics, effectively becoming their own censors. We note that self-censorship conflicts directly with the purpose of private web search, that is, to allow users to freely search for information without revealing their preferences.

4.4.4 Design issues

The magnitude of the universes of both real queries $\mathcal{R}$ and real sequences $\mathbf{R}$ poses an extraordinary challenge to CBPWS designers, who cannot anticipate in advance users' real sequences of activities r to generate a corresponding supersequence $\mathrm{q} = (\mathrm{r} * \mathrm{d})$ to guarantee indistinguishability across a number of sequences in $\mathbf{R}$, i.e. so that $P(\,\mathrm{q} \mid \mathrm{r}_i\,) = P(\,\mathrm{q} \mid \mathrm{r}_j\,),\ \forall \mathrm{r}_i, \mathrm{r}_j \in \mathbf{R}$. Several of the CBPWS tools we have examined propose strategies to overcome this challenge and ensure some level of indistinguishability, but these strategies rely on assumptions about the profiling function and distance across an underlying semantic space that a strategic adversary can exploit. The magnitude of $\mathbf{R}$ further precludes the viability of a flooding DGS or *full padding* to guarantee perfect privacy protection (q.v. Sect. 3.3.2).

Moreover, the majority of tools we have evaluated in the previous section focus on a single aspect of web search, be it search query generation, clicking on results or web browsing to defeat profiling by a search engine that tracks users across the web. In fact, profilers like Google have the ability to monitor not only the users' search queries and clicks on results, but also their web browsing activity beyond the search engine, requiring a CBPWS tool to account for all these activities. Failing to do so introduces vulnerabilities, e.g. adversaries can exploit weak indistinguishability between real and dummy activities in web browsing beyond the search engine to undermine indistinguishability between real and dummy queries.

CBPWS designers must also account for all those features a search engine can exploit to distinguish real from dummy activity, e.g. allowing users to rely on search engines' autocomplete functionality means the DGS must generate

dummy queries using autocomplete too, as otherwise an adversary knows all autocompleted queries are real. Similarly, if adversaries track users' mouse activity, a DGS must simulate dummy mouse activity. We have largely abstracted away from the complexities of implementation and deployment in our examination of CBPWS tools, yet in practice designers must account for all sources of distinguishability.

4.4.5 Hybrid solutions and other alternatives

In this chapter we have examined *"pure"* CBPWS solutions, namely, tools that exclusively rely on chaff to protect web search users from profiling. Other authors have however proposed to combine CBPWS tools with other privacy enhancing technologies (PETs) or to integrate them as part of a larger solution.

We point at the beginning of this chapter to anonymous web browsing systems like Tor as an alternative to CBPWS tools. Anonymisers hinder the creation of search profiles through query unlinkability, whereas obfuscation makes it harder for the adversary to re-identify anonymous users through their queries. Users can combine both, thus obtaining two layers of protection against profiling, although at a higher cost both in terms of bandwidth and time due to increased latency.

Petit et al. propose obfuscation after aggregation [432]. They propose *PEAS* (short for "Private, efficient and accurate web search"), a system that relies on a trusted third party that collects user queries, forwards them to the search engine in addition to dummy queries, then filters out dummy query results and returns to PEAS users their queries' results. Users cannot however unilaterally deploy PEAS, a limitation that sets PEAS off from CBPWS. *X-search*, a similar proposal by Mokhtar et al., relies on a proxy to *mix* several users' queries, incorporating a DGS that populates a pool of dummies with previous users' queries, implicitly assuming that an adversary cannot distinguish dummy queries if these are identical to previous user queries [64].

Other proposals dispense with the trusted third party and rely instead on a network of users that relay user queries on behalf of each other, concealing from the adversary who is the initiator of each query and hindering as a result user profiling [109, 181, 532].

Lastly, designs may also combine utility-preserving chaff-based obfuscation à la CBPWS with utility-degrading query obfuscation à la *query scrambling* of Arampatzis et al. or Sánchez et al., among others [28, 463], which degrades the accuracy of each query in pursuit of a balance between utility and privacy.

4.4.6 Personalisation

One of the purported benefits of profiling is that it enables personalisation of web search so that users can obtain more meaningful results, tailored to their precise informational needs. We have largely abstracted away from personalisation in web search, omitting the negative effect obfuscation may have on personalisation, rendering search results less *useful* for users and thereby introducing a trade-off between utility and privacy that, unless users are uninterested in personalisation, places CBPWS on the realm of utility-degrading obfuscation (UDO).

Previous work has examined the trade-off between privacy and personalisation. Shen et al. distinguish four types of software architectures around web search and personalisation: no personalisation, server-side personalisation, client-side personalisation and client-server collaborative personalisation [486]. Of these four, CBPWS implicitly assumes the first, namely, that personalisation is a non-issue —or even something that users would rather avoid—; however, CBPWS designers may further consider client-side personalisation, whereby the CBPWS tool profiles the user locally and filters and rearranges search results based on $\mathbf{x}$ [146, 516]. In other words, users rely on the CBPWS tool for their own profiling and personalisation, rather than on an adversarial search engine.

On the other hand, to personalise search results search engines may rely on collective profiling rather than individual profiles or on features such as users' language, locations or social network, thus rendering personalisation more resilient to obfuscation (q.v. Sect. 2.2.3).[10] Under collective personalisation, CBPWS users effectively become free-riders on other users' data disclosure, benefiting from profiling practices they do not contribute to.

4.4.7 Ethics and politics

Relying on chaff to introducing noise in users' profiles raises a host of ethical questions, from claims of *"misuse"* of network resources to the *"ungrateful"* and deliberate pollution of data service providers monetise in exchange for web search services. For a discussion on the ethics and politics of chaff in general and CBPWS in particular, we refer the interested reader to the work of Brunton, Howe and Nissenbaum, who have studied at length the ethics and politics of obfuscation [98, 100, 294, 295].

[10] In fact, there are reports that this is currently the case for Google Search [209, 272, 365].

4.5 Conclusion

In this chapter we have studied the deployment of Protos to provide chaff-based private web search (CBPWS). We have instantiated the general Protos model in Sect. 3.1 to capture the particularities of online web search. Furthermore, we have adapted Protos analysis measures both MCA and ACA to illustrate how to measure the level of privacy protection CBPWS tools offer, operationalising a candidate set of privacy properties for private web search.

Equipped with our CBPWS analysis framework, we have evaluated existing CBPWS tool proposals, uncovering systematic flaws and misconceptions related to the operationalisation of privacy properties and evaluation; all the CBPWS tools we have studied implicitly assume a naive adversary that does not attack the system, thereby overestimating the expected level of privacy protection they provide. We have illustrated how a strategic adversary can exploit vulnerabilities in these existing tools to undermine their protection.

CBPWS design involves addressing numerous challenges and problems. Among these, we have highlighted the magnitude of the space of real sequences $\mathbf{R}$ and the inability of CBPWS designers to anticipate users' real activity sequences r to generate obfuscated sequences q that guarantee indistinguishability across a number of sequences r_j large enough to defeat adversarial attacks and provide a meaningful level of privacy protection. Besides, the opacity of adversarial profiling practices means designers cannot tailor DGS to a profiling function they do not know, opening additional vectors of attack for the adversary. These issues complicate the design of robust CBPWS tools at affordable dummy rates ρ to the extent that we must question whether it is possible to design both viable and reliable CBPWS tools. Resorting to hybrid solutions e.g. that combine anonymous web browsing with obfuscation, can counteract the limitations and vulnerabilities of standalone CBPWS.

CBPWS relies on an honest-but-curious (HbC) adversary that processes user web search activities in the presence of obfuscation while attempting to filter out as many dummy queries as possible. We have warned about the delicate balance of incentives that must hold in practice for an adversarial web search provider to be HbC instead of outright block CBPWS users. Ensuring tool undetectability can help CBPWS users to circumvent hostile service providers, yet undetectability gives rise to a host of other issues; it is unclear what the effects of obfuscated profiles are on users, i.e. the impact that algorithmic decision making that takes as input $\mathbf{y}$ instead of $\mathbf{x}$ has on users. Protective optimisation technologies (POTs) represent an alternative line of research to address problems of detectability and algorithmic impact (q.v. Sect. 3.4.1).

Chapter 5

Communication profile confidentiality

> *Deux cents grenadiers ont en quelques heures dressé l'obélisque de Luqxor sur sa base ; suppose-t-on qu'un seul homme, en deux cents jours, en serait venu à bout ?*
>
> —Pierre-Joseph Proudhon, *Qu'est-ce que la propriété ?*

If web search has become an indispensable online service, so have online communication services.[1] The advent of broadband and cheap terminal devices has contributed to the implacable digitisation of human interactions [280]. Lured by the immediacy, convenience and boundless reach of digital communication, people increasingly resort to services like email, instant messaging (IM), voice over IP (VoIP) or social networking sites (SNSs) to fulfil an ever widening spectrum of communication activities such as talking to family and friends, remotely contacting colleagues and clients at work, reuniting with old acquaintances or sharing information and organising collectively.

Popular online communication services like GMail, Whatsapp, Skype or Facebook, however, typically rely on centralised architectures whereby the service provider is able to monitor and monetise all of its users' interactions. Online communication service providers *profile* users according to what they say, to whom, when, where and how often, using that information

[1] According to Alexa, a popular web traffic analysis company, as of April 6th, 2019, Google and Facebook are the most visited and third most visited websites, respectively, globally. See `https://www.alexa.com/topsites`.

to infer users' weaknesses and needs and serve them profitable targeted behavioural advertising [110, 165, 219].

Communication services represent a particularly prized asset to surveillance capitalism, a gold mine of behavioural data. Human communication harbours people's emotions, fears and desires, their affinities, personality traits and impulses [121]; its networked nature enables profilers to infer romantic relationships and sexual orientation, reputation and trust, among several other types of information [205, 300, 406]. Network effects further assist providers to grow and entrench *digital enclosures*, locking-in their user base to stave off competitors and claim exclusive ownership over communication data [26, 538]. Whereas email users can communicate with one another regardless of their email provider, services like Facebook, Whatsapp, Skype, Twitter or Snapchat, to name a few, do not support inter-platform messaging.[2] By denying interoperability with other services and providers, profilers exploit network and bandwagon effects to shackle users to their services, as the more users join a service, the more useful it becomes and the harder it is to leave or for alternatives to compete [503, 538]. SNSs represent a paradigmatic example of the value of networked communications for profilers. These sites gather large amounts of personal information, including private, sensitive communication between users and, as Dean argues, they *"produce and circulate affect as a binding technique"*, thereby encouraging users to spend more time using the service and, as a result, to generate more data [166]. Srnicek points to a *"virtuous cycle"* whereby *"more data means better machine learning, which means better services and more users, which means more data"*, shedding further light on the network effects that contribute to service entrenchment [503].

Interest in mining communication data goes however beyond profilers and advertisers. Digital communications represent a coveted target of state-sponsored surveillance programmes. As whistleblower Edward Snowden has revealed, the US National Security Agency (NSA) and UK's Government Communications Headquarters (GCHQ) have routinely collected —among other types of data— email, video and voice chat, VoIP chats and social networking details, either in transit or stored by cloud service providers, in some cases *"with the ambiguous complicity of Internet companies"* [340, 363, 481].

Against the privacy threats that communication profiling and surveillance pose, a first line of defence involves the use of encryption. End-to-end encryption (E2EE) ensures that only sender and recipient can read the messages they exchange, denying profilers access to the content of communication, i.e. *what* users say.

[2] Providers may however support interoperability across their own services, e.g. Facebook plans to enable users from Facebook Messenger, Whatsapp and Instagram to be able to message each other, all three services property of Facebook, Inc.; similarly, Microsoft enables interoperability between its services *Skype for Business* and *Teams* [382, 582].

Encryption tools such as PGP or Off-the-Record Messaging (OTR) enable users of email and IM, respectively, to unilaterally encrypt their messages end-to-end, regardless of service provider's support for encryption [89]. Alternatively, communication services like Whatsapp and Signal provide E2EE by default.

However, encryption does not prevent the collection and analysis of *communication metadata* such as sender and recipient, time, location, duration of conversation or message volume and frequency. Communication metadata often consists of structured data, rendering it cheap and easy to analyse, and may reveal as much or even more information than content [208, 340]. In particular, it reveals the underlying communication network, enabling inferences that exploit community structure, i.e. the social graph [88, 333]. Previous work has shown how telephone metadata reveals identity, location, relationship status and partner, job type, age, number of family members, sensitive health information or personality traits, among other attributes [122, 163, 372, 574]. NSA General Counsel Stewart Baker's *"metadata absolutely tells you everything about somebody's life. If you have enough metadata, you don't really need content"* and General Michael Hayden's more brazen *"We kill people based on metadata"* highlight the privacy threats that metadata poses [134].

Several alternatives and privacy technologies seek to lessen the privacy threats that stem from metadata. Anonymity systems such as Tor hamper communication profiling by concealing the identities of communication parties, i.e. who the sender and receiver of a given message are.[3] However, for users of communication services that take place within digital enclosures like Facebook, Snapchat, Twitter or Whatsapp, anonymity systems offer no protection against communication profiling. Even if users connect to Facebook or Twitter with Tor, thereby concealing their IP address from the provider, they still need to log-in to their personal accounts on the service, dynamiting their anonymity in the process [402]. Moreover, Tor neither impedes nor alters in any way the providers' ability to monitor all user communications *within* the platform. In practice, deploying anonymous communications against these providers —which amount to global passive adversaries within their own walled gardens— would require dedicated high-latency anonymity networks within the platform, e.g. on Facebook, having users themselves acting as relays.

[3]To be precise, anonymity systems do not *conceal* metadata; rather, they undermine its value —even if they may deploy strategies to prevent the collection of unnecessary metadata, e.g. Tor relies on NoScript to prevent unnecessary scripts from acquiring identifying information about a user. By sending messages through multiple relays, anonymity systems produce more yet less valuable fragments of metadata, thereby increasing the complexity of metadata collection and analysis. Anonymity systems also rely on metadata homogenisation, e.g. making a browser's fingerprint less unique, to prevent user identification [199].

Alternatively, decentralised communication protocols such as XMPP or Diaspora rule out dependence on an IM or SNS provider that monitors and controls all user communications [80]. Decentralisation does not however prevent external adversaries from collecting the metadata the underlying communication network generates, plus it conjures a host of additional vulnerabilities [252]. To provide *metadata free* communication, systems such as Ricochet[4] or RetroShare [454] combine decentralisation with anonymous communications. However, these systems require users to move away from the services they already use, often at unacceptable cost due to strong network effects [206, 503, 577]. In addition, the technical complexity of these systems represents a usability hurdle for non-tech-savvy users (e.g. Ricochet relies on random identifiers instead of usernames), further undermining attempts by privacy-aware users to persuade others to switch services.

Obfuscation on the other hand, while not exempt of its own usability challenges, enables users to continue using their preferred communication services by polluting the *communication profiles* that the adversary builds with users' communication metadata. By relying on Protos to obfuscate communication patterns, we prevent profilers from retrieving users' real communication profiles.

There is however a shortage of research on standalone chaff-based profile obfuscation tools. Early work on communications security propose chaff as an additional, optional means to achieve a high level of traffic analysis resistance through *full padding* of a communication network's links [318, 536]. These proposals however neither examine the level of protection that chaff provides below full padding levels nor alternative strategies to full padding, relegating the use of chaff to little more than an expensive idea. More recent proposals study chaff as part of anonymous communication systems design, instead of a standalone Proto [405, 420]. Hence, in this chapter we provide a first set of contributions towards bridging this research gap.

This chapter is organised as follows. In Sect. 5.1 we instantiate the chaff-based profile obfuscation (CPO) model to the particular scenario of online communication services, introducing communication profile obfuscation tools (cProtos). We introduce the cProtos' analysis framework in Sect. 5.2. We propose communication profile confidentiality (CPC) as reference privacy property with two possible variants: contact-exposed and contact-hidden CPC, both of which we operationalise using information leakage. We demonstrate how to compute mutual information to measure information leakage and assist cProtos' DGS design for SNSs in Sect. 5.3. More particularly, we illustrate DGS evaluation in Sect. 5.3.1 and side-channel information leakage from auxiliary online social network (OSN) metadata in Sect. 5.3.2. We examine issues

[4]See `https://ricochet.im/`

regarding the provision of E2EE in SNSs in Sect. 5.4, focusing on the role social network providers (SNPs) can play as E2EE enablers in Sect. 5.4.1, and attending to users' attitudes and perceptions to third-party E2EE tools (TPETs) in Sect. 5.4.2. Lastly, we discuss additional matters in Sect. 5.5 and conclude in Sect. 5.6.

5.1 Modelling online communication profile confidentiality

5.1.1 System model

We consider an online *communication service*, e.g. an online site or web application that enables a set of users V to communicate with one another. We abstract away from the particular communication type or medium, be it private messages in a dedicated messaging application like Whatsapp, public posts in a social networking site like Facebook or comments in a video sharing platform like Youtube, and consider that users exchange *messages*.

We denote a user message as r and rely on additional subscripts to specify sender and receiver. We denote a message Alice sends to Bob as $r_{\mathrm{A,B}}$ and a message that Alice or Bob exchange, regardless of who the sender is, as r_{AB}. To denote multirecipient messages, we may add subscripts accordingly, e.g. denoting a message Bob sends to Alice, Carol and Frank as $r_{\mathrm{B,ACF}}$. However, to avoid complex notation, we split multirecipient messages into as many individual messages as recipients, e.g. we split $r_{\mathrm{B,ACF}}$ into $r_{\mathrm{B,A}}$, $r_{\mathrm{B,C}}$ and $r_{\mathrm{B,F}}$. We denote the sequence of messages Alice sends as $\mathrm{r_A} = [r_1, r_2, \ldots, r_m]$; with $\mathrm{r_{A,B}}$ denoting the sequence of messages Alice sends to Bob (disregarding Bob's responses) and $\mathrm{r_{AB}}$ the sequence of messages Alice and Bob exchange (including Bob's responses).

Moreover, we consider a set of functions κ_i that evaluate the *relationship* among any two users of the communication service, e.g. binary function κ_F classifies two users as *friends* if the number of messages they exchange is greater than a threshold θ_F. A function κ takes an arbitrary number of variables into account, such as the timing and frequency of messages or the number of people two people share as friends or contacts.

Each function κ induces a *communication graph* $G := (V, E)$, where each vertex $v \in V$ represents a user and each $e \in E$ represents the relationship between two users v_i and v_j that function κ assigns to them, e.g. $G := (V, E_F)$ represents the friends' graph, where an edge e_F exists between two users if they are

friends. Non-boolean functions κ induce a weighted graph G, where each edge e represents the *weight* of the relationship between two users. For each user, say Alice, $G_{E_i}[\mathrm{A}]$ denotes the local subgraph that Alice and her set of type E_i relationships induce, e.g. $G_F[\mathrm{A}]$ denotes the local subgraph that Alice induces through her friend relationships.[5]

Whereas users may communicate using several *interoperable* communication services or use various services that the same provider operates,[6] for simplicity and without loss of generality we consider one service that only one provider is responsible for, which we refer to as, like in previous chapters, the *service provider*.

We assume that user communication is encrypted *end-to-end* (E2EE), namely, users communicate using encrypted messages so that no third party intercepting or enabling the communication process e.g. the service provider can see that users exchange encrypted messages, how many, how often, yet is unable to decrypt them and thus retrieve their content. We however note that end-to-end encryption (E2EE) is not a fundamental requirement or assumption in our model; we show in Sect. 5.1.3 that Protos can incorporate E2EE to systems where it is not available. Lastly, E2EE is still vulnerable to adversaries who either steal the decryption keys or compromise users' devices to eavesdrop on their messages once decrypted on the client side. This is an orthogonal security issue as we assume, like in previous chapters, that user devices are secure, i.e. free from malware that monitors and leaks information about users' communication.

Figure 5.1 provides a graphical depiction of the system and threat model.

5.1.2 Threat model

The online communication service provider observes all user communication, i.e. the sequence of messages r that any user generates and therefore the sequence of messages r_{AB} that any two users exchange. Since users encrypt their messages end-to-end, the service provider is unable to read their contents; however, it can determine the number of messages users exchange, sender and receiver, their timing and frequency.

The provider collects and processes each user's sequence of messages $\mathrm{r} = [r_1, r_2, \ldots, r_m]$ into a *communication profile* $\mathbf{x}$. We model a communication profile as a multinomial distribution $\mathbf{x} = \{x_i\}$, where each element x_i represents

[5] We slightly abuse notation here and use G_F instead of G_{E_F} to avoid double subscripts.

[6] As an example, Facebook Inc. owns Facebook, Whatsapp and Instagram thus provides communication to all three services' users, plus it currently has plans to make interplatform communication possible [56]

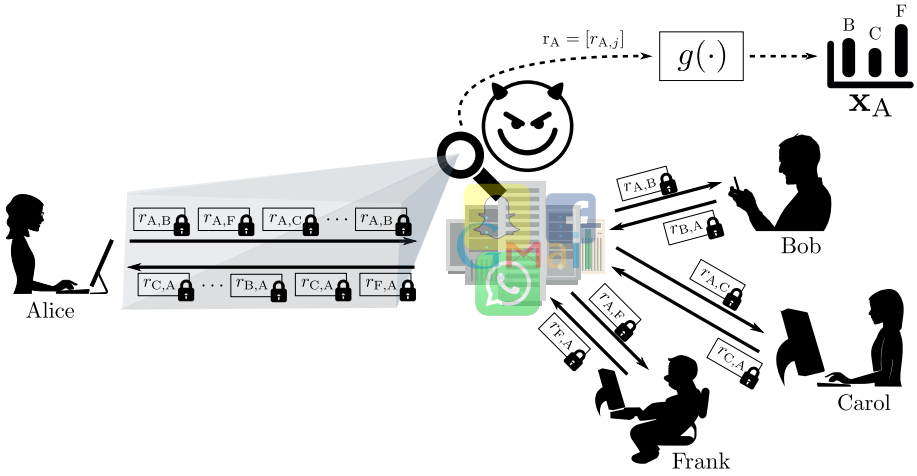


Figure 5.1: Online communication system and threat models.

a *probability* a profiling function assigns to a category i. The choice of categories i and the meaning or interpretation of probability x_i depend on the profiling function $g(\mathrm{r})$. We do not make any assumptions about $g(\cdot)$, i.e. it may map messages to categories according to message size, recipient, sending time or any other contextual information available to the adversary, e.g. each category x_i represents a recipient v_i and the profiling function g simply counts the number of messages user v_i sends to v_j. Under such g, communication profile $\mathbf{x} = \{x_i\}$ denotes the probability that a user, say Alice, sends a message to a recipient v_i. Despite the apparent simplicity of this profiling function, we note that choosing how to count messages between senders and recipients is far from trivial, e.g. if two people communicate over email, we may choose to count recipients in the *CC* and *BCC* fields, count them only if they actively participate in the conversation, or assign them a different *weight* than to participants in the *TO* field.

5.1.3 Communication Protos

A *communication profile obfuscation tool* (cProto) generates dummy messages on behalf of the user to prevent the adversary from retrieving the user's communication profile $\mathbf{x}$.

We denote dummy messages as d and define additional subtypes of communication types such as dummy posts or dummy comments when appropriate.

We mirror our model of chaff-based private web search (CBPWS) tools (q.v. 4.1.3) to conceptualise cProtos. Hence, we define a *privacy property*, a *privacy measure* and *dummy generation strategy* (DGS). We further consider a DGS' dummy rate ρ as the proportion of dummy messages to total messages, and dummy, target and observed profiles ($\mathbf{w}, \mathbf{y}^t$ and $\mathbf{y}$, respectively).

Content indistinguishability and filtering through cryptography.

cProtos differ from CBPWS tools in terms of the relationship between users and the adversarial, uncooperative provider. In CBPWS, users interact with the search engine provider alone, to whom they need to disclose their search queries to obtain a list of search results. In communication services, users interact with each other and do need to expose the content of their messages to the service provider, who is merely an intermediary responsible for message transport and storage. That is why in our cProtos model we assume the deployment of E2EE, to prevent an adversarial provider from accessing messages' content.

E2EE further assists dummy generation strategies (DGSs) design by facilitating content indistinguishability across real and dummy messages. Assuming a semantically secure cryptosystem, such as AES [147], encrypted real messages are indistinguishable from dummy messages *of the same length*, regardless of their corresponding content in plaintext. Hence, encryption not only ensures content confidentiality, but it also relieves cProtos designers of the burden of generating indistinguishable plaintext content for dummy messages. In fact, we recall from Chapter 4 that one of the main hindrances in CBPWS tool design is the large and unknown universe of user queries. CBPWS tools cannot anticipate in advance the queries a user generates, thus complicating the task of mapping real sequences to dummy supersequences. Encryption addresses this issue and in turn simplifies DGS design by making both real and dummy messages indistinguishable content-wise; DGS designers do not need to determine in advance the universe of real messages, as encryption ensures that the adversary cannot exploit it. An adversary can still however exploit message length to tell real and dummy messages apart, a vulnerability cProtos designers can address by splitting and padding real messages [447], fitting them to a predefined length set that all real and dummy messages conform to.

On the other hand, whereas CBPWS tools do not need to disclose which queries are real and dummy to anyone but the querying user herself, cProtos require a mechanism that allows communicating users to filter out dummy messages, as otherwise their own communication becomes polluted by noise. Dummy message filtering must however be available only to communicating users, as otherwise it compromises the indistinguishability between real and dummy

messages towards adversarial parties who do not belong in the conversation, such as the service provider. cProtos can also rely on E2EE to address this issue, using message authentication codes (MACs). Following Rivest's chaffing and winnowing strategy [453], a cProto *tags* real and dummy messages with valid and invalid MACs, respectively. As Rivest notes, a *"typical MAC algorithm (such as HMAC-SHA1) will appear to act like a 'random function' to the adversary, and in such a case the adversary will not be able to distinguish wheat from chaff"* [453]. Communicating users can however easily discard dummy messages. Upon message reception, users' decryption application checks each message's MAC, discarding those messages that carry invalid MACs hence automatically filtering out dummy messages.

cProtos can thus leverage E2EE to ensure both content indistinguishability and (authorised) dummy message filtering. Whereas we assume that E2EE is available by default to highlight the privacy risks involved in profiling communication patterns, we acknowledge that this is not often the case in online communication applications [249]. As we discuss later in Sect. 5.4.1, cProtos designers can therefore choose to deploy E2EE in communication services where it is not available to guarantee content indistinguishability (thus confidentiality) and enable ease of authorised dummy message filtering. Still, we note that despite the advantages that E2EE offers, encryption is not a mandatory component of cProtos design; cProtos designers may attempt to generate plausible dummy plaintext messages like CBPWS designers generate dummy queries and devise alternative filtering strategies, like having users agree out-of-band and in advance on real and dummy positions in a sequence of messages [120]. Given however the superior security guarantees that E2EE offers, we do not study any such alternatives in this chapter.

5.1.4 Adversary model

We mirror CBPWS' adversary model to instantiate the general Protos adversary model (q.v Sect. 3.1.3) in the particular case of cProtos. To avoid reproducing CBPWS' adversary model, we point the reader to Sect. 4.1.4 and provide here a brief summary.

The adversary's goal is to obtain users' communication profiles $\mathbf{x}$. However, once users deploy cProtos, if the adversary processes both real and dummy messages it obtains an *observed profile* $\mathbf{y}$ that may bear no relation to the *real profile* $\mathbf{x}$. To recover $\mathbf{x}$, the adversary deploys *attack strategies* that seek to remove as many dummy messages as possible from q, the sequence of messages, both real and dummy, it observes from the user. As a result of an attack, the adversary obtains a *filtered profile* $\hat{\mathbf{x}}$. The adversary is honest-but-curious

(HbC), thus attempts to remove as much dummy messages as possible from q, yet does not disrupt users' quality of service (QoS).

As a minor point of divergence between CBPWS and cProtos, we do not differentiate across two types of attack strategy, profile-based and message-based. Although it remains a useful categorisation, we do not rely on it in this chapter.

5.2 Analysis framework

We provide an analysis framework for cProtos. We define a set of privacy properties for communication profile confidentiality and operationalise them using the measure of information leakage we introduce in Sect. 3.2. Table 5.1 offers a summary of the specific notation we use throughout this section. We refer the reader to Table 3.1 for general Protos notation we have previously introduced.

5.2.1 Privacy properties

Similarly to CBPWS, there is a plethora of privacy properties a cProto may attempt to provide. Given the impossibility of providing an exhaustive list of users' privacy concerns and requirements, we focus on two flavours of *communication profile confidentiality* (CPC), namely, *contact-exposed* and *contact-hiding*.

Contact-exposed CPC guarantees that the adversary cannot determine a user's communication profile $\mathbf{x}$ among a users' *contacts*, according to a function κ_{contact} that defines what a user contact is. For illustration purposes we generally consider as contacts those individuals with whom users communicate at least once, regardless of whether or not those individuals respond. Hence, contact-exposed profile confidentiality conceals the amount of messages users send to their contacts, but not the list of contacts itself.

Contact-hidden CPC guarantees that the adversary cannot determine a user's communication profile $\mathbf{x}$, including a user's contacts, i.e. it hides both the amount of messages users send to their contacts and who users' contacts are.

Symbol	Meaning
v	User
V	Set of users / Set of graph nodes
$r_{\mathrm{A,B}}$	Message Alice sends to Bob
$\mathrm{r_{AB}}$	Sequence of messages between Alice and Bob
κ	Relationship classification function
θ	A threshold
G	OSN/communication graph
E	Set of graph edges
x	Communication profile component
χ	R.v. over (real) profile components
y	Observed profile component
υ	R.v. over observed profile components
σ	Number of samples
Δ	Quantisation step
M	R.v. over number of messages
F	R.v. over number of friends
Z	R.v. over number of (public) posts

Table 5.1: Summary of notation.

5.2.2 Privacy measure

We choose to measure profile confidentiality through information leakage (q.v. 3.2.1). We choose information leakage, a mechanism-centred analysis (MCA) measure, to abstract away from particular adversaries and attack strategies. Moreover, we favour information leakage over indistinguishability measures because we do not intend to characterise the behaviour of a cProto for every possible probability distribution $P(\,\mathrm{R} = \mathrm{r}\,)$ of input sequences. Instead, we empirically assess information leakage over a selected range of input sequences, as we further illustrate in Sect. 5.3.1.

We recall that to measure information leakage we use mutual information as follows:

$$\begin{aligned} \mathrm{I}(\mathrm{R};\mathrm{Q}) &\equiv \mathrm{H}(\mathrm{R}) - \mathrm{H}(\mathrm{R} \mid \mathrm{Q}) \\ &= \sum_{\mathrm{r}\in\mathrm{R}} \sum_{\mathrm{q}\in\mathrm{Q}} P(\,\mathrm{r},\mathrm{q}\,) \log \frac{P(\,\mathrm{r} \mid \mathrm{q}\,)}{P(\,\mathrm{r}\,)P(\,\mathrm{q}\,)} \end{aligned} \tag{5.1}$$

where R represents the random variable whose domain is the space $\mathbf{R}$ of real sequences of messages r and Q represents the random variable whose domain is the space $\mathbf{Q}$ of obfuscated sequences of messages q.

To measure contact-exposed CPC, we consider random variable $\mathrm{R_A}$ over the space of possible real message sequences $\mathbf{R}_\mathrm{A}$ whose underlying contact list is a subset of Alice's contact list $\kappa(\mathrm{r_A})$, i.e. $\mathbf{R}_\mathrm{A} = \{\mathrm{r}_i : \kappa(\mathrm{r}_i) \subseteq \kappa(\mathrm{r_A})\}$.

To measure contact-hidden CPC, we consider random variable $\mathrm{R_A^+}$ over a space of possible real messages sequences $\mathbf{R}_{\mathrm{A}^+}$ whose underlying contact list *includes and extends* Alice's contact list $\kappa(\mathrm{r_A})$, i.e. $\mathbf{R}_{\mathrm{A}^+} = \{\mathrm{r}_i, \mathrm{r}_j : \kappa(\mathrm{r}_i) \subseteq \kappa(\mathrm{r_A}), \kappa(\mathrm{r}_j) \neq \kappa(\mathrm{r_A}), \forall \mathrm{r}_j \in \mathbf{R}' \subseteq \mathbf{R}\}$.

Information leakage is minimal when $\mathrm{I}(\mathrm{R};\mathrm{Q}) = 0$ and maximal when $\mathrm{I}(\mathrm{R};\mathrm{Q}) = \mathrm{H}(\mathrm{R})$, which corresponds to maximum and minimum levels of CPC. Moreover, whenever $\mathrm{I}(\mathrm{R};\mathrm{Q}) = 0$, contact-hidden CPC offers greater privacy protection than contact-exposed CPC. For intermediate values $0 < \mathrm{I}(\mathrm{R};\mathrm{Q}) < \mathrm{H}(\mathrm{R})$ however, information leakage is, as an average measure, not sufficiently expressive to represent what precisely an adversary learns about a users' communication profile. In contact-exposed CPC, the bits of information a cProto leaks may enable an adversary to determine the people the user contacts the most — or the least. Similarly, in contact-hidden CPC, the number of bits may prevent an adversary from determining a user's real contacts among her least contacted people, yet expose the section of her profile that relates to her most frequent contacts. Accounting for more fine-grained notions of privacy requires computing mutual information between sets of input sequences accordingly, e.g. to capture whether a DGS leaks information about Alice's best friend, say Bob, we consider random variable $\mathrm{R_{BFF}}$ that takes values 0 and 1 according to a function $\kappa(\mathrm{r}) : \mathbf{R} \mapsto (0, 1)$ that defines whether or not Bob emerges from input sequence r as Alice's best friend, respectively, and compute mutual information $\mathrm{I}(\mathrm{R_{BFF}};\mathrm{Q})$. We further illustrate how to compute information leakage for alternative privacy requirements such as concealing best friends in Sect. 5.3.1.

5.3 Design and evaluation of cProtos for social networking sites

OSNs pose a challenging scenario for cProtos design due to the wealth of user information they contain and types of interactions they support. Users engage in various types of online interaction, generating troves of metadata; they post on their friends' pages, participate in other users' conversation threads and comment on and further share news, photos and videos.

A first challenge we face in DGS design is the selection of metadata features we wish to obfuscate or, equivalently, how to model sequences of messages r. Each message users send generates several metadata, most obviously its intended audience, day and time, but also user device and browser, time since last message or time since previous interaction with a particular recipient, among others. Accounting for each of these metadata increases the complexity of DGS design, namely, we need to ensure that none of these metadata leaks information that the adversary can exploit to distinguish real from dummy messages.

Moreover, the adversary has access to the OSN *graph* as well as to user activity that does not leave visible traces on the site. The adversary can exploit graph *topology* features such as who is *"friends"* with whom, how many friends two users have in common or how *"well connected"* users are, as well as metadata from e.g. users' visits to their friends' personal pages or the time they spend examining a particular piece of content.

Since the adversary has access to and collects all OSN users' activity, we need to determine not only the set of metadata we wish to obfuscate, but also auxiliary sources of information. Even if we determine that users require CPC for one type of communication and not for others, say private messages and public posts, respectively, *correlations* between types of communication require a coordinated DGS to prevent *side-channel leakage*; if the user communication patterns we intend to obfuscate correlate with other types of OSN activity, adversaries can exploit unobfuscated activity to undermine a cProto's DGS, e.g. if Alice sends more private messages to those OSN friends on whose posts she comments the most, an adversary can exploit information about unobfuscated Alice comments on her friends' posts to infer her private communication profile. Moreover, *social graph* properties such as the number of friends users have in common or their *betweenness centrality* in the network [239] may also correlate with their communication patterns, as well as user activity that does not leave visible traces on the site, providing side information to the adversary to discard dummy messages. Obfuscating every type of user activity the OSN supports increases DGS design complexity, yet if we dismiss sources of metadata to simplify DGS design we risk overestimating the protection a cProto offers.

In this section we demonstrate how to leverage the measurement of information leakage to assist DGS design. First, in Sect. 5.3.1, we demonstrate how to measure a DGS's information leakage through a series of simplifications that ease and hasten mutual information computation, thereby enabling exploratory DGS evaluation. Then, in Sect. 5.3.2, we measure correlation strength between OSN's metadata sources to identify strongly correlated sources that we must consider as well as weakly correlated or uncorrelated sources that we can safely dismiss to simplify DGS design.

5.3.1 DGS evaluation

To illustrate how to measure a DGS's information leakage, we compute mutual information between real and obfuscated user communication in a series of experiments where we vary the DGS, the OSN's network topology and user communication patterns. We run all experiments by generating synthetic traces of OSN interactions using a Python OSN simulator of cProto users in OSNs [37].

First, we examine practical issues related to the computation of mutual information as well as simplifications that ease DGS evaluation. Then, we present the results of our experiments.

Computing mutual information.

Computing mutual information requires that we obtain the probability distributions $P(\mathrm{R} = \mathrm{r})$ and $P(\mathrm{Q} = \mathrm{q})$, as well as the joint probability distribution $P(\mathrm{r}, \mathrm{q})$. We can obtain random variable R by sampling observations of user communication in OSNs either from existing OSN data or from simulated OSN user interactions following models or known patterns of user communication in OSNs [67, 68, 241, 533, 554]. We however refrain from obtaining the observed sequences random variable Q and the joint variable (R, Q) analytically due to the complexity of modelling interactions between users. Hence, we estimate them from actual observations, namely, by sampling observations of user communication that we obfuscate with cProtos.

Sampling. To estimate random variables R, Q and (R, Q) we run a cProtos simulator that intertwines dummy interactions with user interactions [37]. In each simulation run we obtain a sample of each random variable by selecting a user v_i uniformly at random and storing $(\mathrm{r}_i, \mathrm{q}_i)$. We repeat this process to obtain an arbitrary number σ of samples.

We compute probability $P(\,(\mathrm{R},\mathrm{Q}) = (\mathrm{r},\mathrm{q})\,)$ by counting the number of occurrences $c(\mathrm{r},\mathrm{q})$ of each pair of values (r,q) and dividing it by the total number of samples σ. However, using a finite number of samples introduces an error in the estimation. We model $P(\,\mathrm{R},\mathrm{Q}\,)$ as a multinomial distribution and use Bayesian inference to obtain a bound on this error.

The Dirichlet distribution is a conjugate prior for the multinomial distribution. Its probability density function represents the belief that the probability of occurrence of (r,q) is $P(\,\mathrm{r},\mathrm{q}\,)$ given it has been observed $c(\mathrm{r},\mathrm{q})$ times. We obtain σ' samples $P(\,\mathrm{r},\mathrm{q}\,)$ using the Dirichlet distribution with '$c(\mathrm{r},\mathrm{q})$' as input parameters:

$$P(\,\mathrm{R},\mathrm{Q}\,) \sim \mathrm{Dirichlet}(c(\mathrm{r}_1,\mathrm{q}_1),...,c(\mathrm{r}_{|\mathbf{R}|},\mathrm{q}_1),...,c(\mathrm{r}_{|\mathbf{R}|},\mathrm{q}_{|\mathbf{Q}|})) \tag{5.2}$$

where subindexes i, j as in represent a particular r_i and q_j in the space of real sequences $\mathbf{R}$ and the space of obfuscated sequences $\mathbf{Q}$.

For each sample drawn from the Dirichlet, we calculate mutual information as follows:

$$\mathrm{I}(\mathrm{R};\mathrm{Q}) = \sum_{\mathrm{r}\in\mathbf{R}}\sum_{\mathrm{q}\in\mathbf{Q}} P(\,\mathrm{r},\mathrm{q}\,)\log\left(\frac{P(\,\mathrm{r},\mathrm{q}\,)}{\sum_{\mathrm{r}\in\mathbf{R}} P(\,\mathrm{r},\mathrm{q}\,)\cdot\sum_{\mathrm{q}\in\mathbf{Q}} P(\,\mathrm{r},\mathrm{q}\,)}\right) \tag{5.3}$$

and take the median value of $\mathrm{I}(\mathrm{R};\mathrm{Q})$ as the estimated mutual information value. To estimate error, we consider the interval containing a given percentage of values around the median, e.g. we use the lowest value in the first quartile and the highest value in the third quartile as error bounds, thus considering the 50% of values around the median.

Profiles and profile components to limit sampling requirements. The universe of sequences $\mathbf{R}$ and $\mathbf{Q}$ that we need to sample depends on the metadata we select to characterise message sequences. We may count messages Alice sends to Bob and Charlie, account for time between messages, time of day or number of friends in common, among several other metadata. Each additional variable represents an additional dimension in the sequence space, augmenting the universe of possibilities we need to consider and sample. Each variable may also provide to the adversary additional information to distinguish real from dummy messages.

Any choice of sequence characterisation focuses on a set of metadata we assume the adversary exploits, implicitly dismissing other potentially informative metadata, e.g. by omitting the time of day a user sends messages in a

sequence characterisation, we implicitly disregard the information it provides to the adversary. Whereas omitting metadata disregards potential sources of information leakage, it also simplifies DGS design and evaluation. Moreover, accounting for fewer metadata enables us to measure the amount of information those metadata alone leak, thereby isolating their contribution to information leakage and assisting the identification of those features that leak the most information. Hence, we can first design a simple DGS that considers a small set of metadata and assess the level of CPC it provides before incorporating additional metadata that increase DGS complexity.

Choosing fewer metadata further simplifies mutual information computation. Since each additional source of metadata represents an additional variable (with potentially infinite values) in the sequence space, fewer metadata leads to a smaller universe of potential sequences we need to sample —thus less time we employ collecting samples— and each sample requires less memory or data storage. As a result, with shorter time and fewer memory requirements, computation is more feasible.

In our DGS evaluation we choose to replace each sequence r with a *profile* $\mathbf{x} = g(\mathrm{r})$ that we build as

$$\mathbf{x} = [x_{ij}] = g(\mathrm{r}) = \frac{c(r_{i,j})}{\sum_j c(r_{i,j})}$$

where $c(r_{i,j})$ represents a counter that g increments as follows. If v_i sends a message to v_j as the only recipient, function g increases $c(r_{i,j})$ by one. If v_i sends a message where v_j is one among a group k recipients, function g increases $c(r_{i,j})$ by $1/k$, regardless of any underlying recipient classification.

Hence, we compute mutual information between the random variables X and Y that g induces as:

$$\begin{aligned} P(\,X, Y\,) \;\sim \mathrm{Dirichlet}(&c(\mathbf{x}_1, \mathbf{y}_1) + 1, \ldots \\ &\ldots, c(\mathbf{x}_m, \mathbf{y}_1) + 1, \ldots, c(\mathbf{x}_m, \mathbf{y}_n) + 1) \end{aligned} \tag{5.4}$$

The "+1" in Eq. 5.4 indicate that we assume negligible prior knowledge on the probability values we estimate, namely, we simply assume that all pairs $(\mathbf{x}, \mathbf{y})$ have a probability greater than zero. We calculate mutual information according to Eq. 5.3, simply replacing random variable (R, Q) with the corresponding random variable over profiles (X, Y) we compute in Eq. 5.4.

We acknowledge that replacing sequences r and q with profiles $\mathbf{x}$ and $\mathbf{y}$ underestimates an adversary's ability to distinguish real from dummy messages,

as the profiling function g we consider disregards any information about a sequence of messages other than the number of messages two users exchange. However, this strategy also enables us to significantly shrink the sequence space and focus on the information that obfuscated volume alone leaks about the actual volume of communication between recipients. Moreover, because throughout the simulations we run we only need to store profiles $\mathbf{x}$, which we can compute on the fly, instead of storing each message r a user sends with its corresponding metadata, we drastically reduce simulations' memory requirements. In addition, less informative random variables provide a lower bound on the amount of information a cProto leaks.

To shrink the sequence space and speed up mutual information computation even further, we also consider random variables χ, υ and (χ, υ), over real profile components x, observed profile components y and the joint process (x, y), respectively. This means that instead of computing the information leakage that observed profiles $\mathbf{y} = \{y_i\}$ provide over real profiles $\mathbf{x} = \{x_i\}$, we focus on their individual components y_i and x_i, respectively. In so doing, we compute how much individual observed profile components y_i leak about the corresponding real component x_i disregarding the allocation of a profile's weights among the remaining components y_j, x_j —in turn simplifying computation.

Conversely, it is possible —computational resources allowing— to consider not only sequences with a richer set of features, but also correlations among profiles of several users, thereby incorporating the networked effects of communication, e.g. we may evaluate the system as a whole by considering random variables that capture every communication profile in the social network, rather than individual user profiles. To do so, it suffices to replace the random variables over sequences in Eqs. 5.2 and 5.3 with the random variables over all profiles in the social network.

Quantisation. We quantise profile components x to constrain them to a discrete set of values. The step of quantisation Δ defines the length of the interval where continuous values map to a single discrete value. The effect of quantisation is twofold. On the one hand, increasing the quantisation step shrinks the state space, reducing the number of samples required to compute information leakage and speeding up mutual information computation. On the other hand, large quantisation steps group many x values together, hence further reducing the amount of information we consider available to an adversary.

Moreover, we may select uniform or non-uniform quantisation steps, e.g. we define an arbitrary threshold θ to group a user's communication partners into

"close friends" (i.e. v_j such that $x_{ij} \geq \theta$), and "acquaintances" (i.e., v_j such that $x_{ij} < \theta$), thereby implicitly applying a classification function $\kappa_{\{\text{close friends}\}}$.

Evaluation.

Experimental setup. We generate synthetic traces of OSN communication using Balsa et al.'s simulator of cProtos users in OSNs [37].

Network topology. We consider two toy-example social networks. The first has size $|V| = 20$ users, each user having 6 friends. In particular, $\kappa_F(v_i) = \{v_j\}, j = \{i-3, i-2, i-1, i+1, i+2, i+3\} \mod 20$. The second has size $|V| = 4$ and is fully connected, namely, all users are friends with each other. These networks are orders of magnitude smaller than typical OSNs, yet sufficient to illustrate how to compute information leakage through mutual information.

User behaviour. To illustrate changes in the amount of information a cProto leaks under different models of user communication behaviour, we consider two types of communication profiles. On the one hand, *worst case* profiles model scenarios in which users communicate in pairs, namely, each user exclusively interacts with one of her friends, never with any of her other contacts. Hence, each user profile $\mathbf{x}_i$ has a single weight $x_{ij} = 1$, while the remaining weights $x_{ik} = 0, \forall k \neq j$. We consider this profile to be a *worst case* for cProtos because the DGS must conceal one very strong relationship that concentrates all the user's interactions. On the other hand, *skewed* profiles model more realistic scenarios in which users communicate with all their friends, yet a fraction of friends receives significantly more traffic than others. We generate skewed profiles following Diaz et al.'s method [174].

Dummy generation strategy. We consider two DGSs, a *non-adaptive* DGS and an *adaptive* DGS, to illustrate how mutual information captures the difference in the amount of information different DGSs leak.

The *non-adaptive* strategy selects a set of dummy weights w_{ij} for each user v_i by drawing samples from a uniform distribution, then normalising the resulting vector $\mathbf{w}$. Then, it generates dummy messages from v_i to v_j according to $\mathbf{w}$ alone, this is, without taking into account previous interactions or the real profile $\mathbf{x}$. The non-adaptive strategy's dummy weights w_{ij} are thus independent from real weights x_{ij} and so is each dummy profile $\mathbf{w}_i$ independent from $\mathbf{x}_i$.

The *adaptive* strategy similarly selects a set of *target* weights y^t_{ij} for each user v_i by drawing samples from a uniform distribution, then normalising the resulting vector $\mathbf{y}^t$. However, the adaptive strategy monitors user v_i's communication and generates dummy traffic so that the observed profile $\mathbf{y}$ is as similar as

possible to $\mathbf{y}^t$. To that end, whenever the $\mathbf{y}$ deviates from $\mathbf{y}^t$, the adaptive DGS alters the recipients of subsequent dummy messages to bring $\mathbf{y}$ back to $\mathbf{y}^t$.

Quantisation. We perform uniform quantisation with the number of quantisation steps varying between two and five, i.e. $\Delta = \{1/2, 1/3, 1/4, 1/5\}$. We expect coarser intervals of quantisation, such as $\Delta = 1/2$, to lead to smaller values of mutual information, showing that we lose information by reducing the universe of values. Conversely, we retain more information —thus more accurately measure information leakage— with smaller quantisation steps, e.g., $\Delta = 1/5$.

Moreover, we perform non-uniform quantisation to identify a user's *"best friend"*, which we define as the contact with whom a user interacts the most, i.e. $v_j : j = \arg\max_j(x_{ij})$. To this end, we define a per-user threshold $\theta_i = \max(x_{ij})$, resulting in two quantisation intervals: one that contains the maximum weight $\max_{\mathbf{x}}(x_{ij})$ and another for the remaining weights.

Sampling and error estimation. To minimise estimation error, for each experiment we draw $\sigma = 500\,000$ samples of $(\mathbf{x}, \mathbf{y})$ and (x, y) to compute $P(\mathbf{x}, \mathbf{y})$ and $P(x, y)$, respectively. For each quantisation step Δ we select the median value of mutual information from $\sigma' = 1000$ Dirichlet samples and the values of the first and third quartiles as error estimators.

Results.

We present the results of our evaluation. In all figures, the vertical axis represents mutual information and symbols represent different quantisation steps. The horizontal axis represents the dummy rate ρ. We include $\rho = 0$ to represent the special case where the DGS generates no dummies, so that the profile $\mathbf{y}$ the adversary observes is the real profile, i.e. $\mathbf{y} = \mathbf{x}$. A dummy rate $\rho = 0$ leads to maximal information leakage with mutual information taking value $\mathrm{H}(X)$ in the case of communication profiles and $\mathrm{H}(\chi)$ in the case of profile components.

To better illustrate the influence of other parameters, unless we state otherwise we keep the quantisation step uniform and simulate skewed profiles. For each quantisation step Δ we represent the median value of mutual information. To estimate error, we consider the lowest value in the first quartile and the highest value in the third quartile as error bounds (i.e. the 50% of values around the median); however, the figures below do not show these values as they are almost identical to the median, thereby guaranteeing that we have drawn enough samples of profile pairs $(\mathbf{x}, \mathbf{y})$ and profile components (x, y).

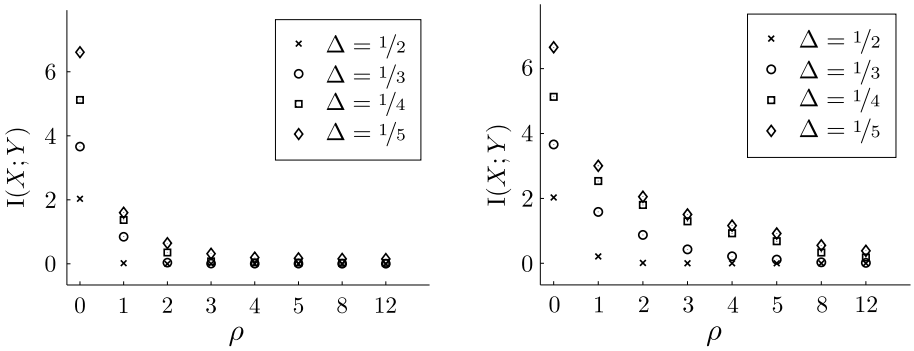
Dummy generation strategy. We examine the difference in terms of information leakage between non-adaptive and adaptive DGSs. We consider the OSN with $|V| = 20$ users. Figures 5.2 and 5.3 show that the adaptive DGS leaks less information than the non-adaptive DGS. Whereas the adaptive DGS dynamically generates dummy messages to hide the real profile $\mathbf{x}$ in an unrelated, random profile $\mathbf{y} \to \mathbf{y}^t$, the non-adaptive DGS discloses a combined profile $\mathbf{y} = \mathbf{x} + \mathbf{w}$, so that for low dummy rates user profile $\mathbf{x}$ remains a prominent component of $\mathbf{y}$.

Dummy rate. Figures 5.2 and 5.3 further illustrate the dummy rate's effect on DGS effectiveness. More dummies decrease the dependence of the observed profile (components) on the real one(s), thus reducing information leakage. Still, higher dummy rates produce diminishing returns; a small amount of dummies considerably reduces information leakage, yet further increasing the budget of dummies does not lead to a proportional decrease in information leakage, arguably due to the profiles' *skewness*: small dummy rates quickly obfuscate a profile's weakest components, while hiding stronger components requires a considerably larger budget of dummies.

Quantisation step. Figures 5.2 and 5.3 also show that the quantisation step has a big influence on information leakage, since smaller quantisation steps retain more information. However, we observe that the decay function is steeper for small steps so that results across quantisation steps converge soon, suggesting that it is possible to perform computationally inexpensive analyses to obtain a first approximation of a DGS's effectiveness.

Profiles and profile components. Figures 5.2 and 5.3 also illustrate the decrease in information leakage when we consider profile components as opposed to whole user profiles. By considering profiles as a whole, we measure the information that interdependencies between interactions leak (e.g. when Alice sends a message to Bob, she is not sending a message to Charlie) improving profile components' estimation accuracy. Still, information leakage similarly decreases with higher dummy rates for both profiles and profile components, thus justifying preliminary DGS evaluation with profile components to minimise computational requirements.

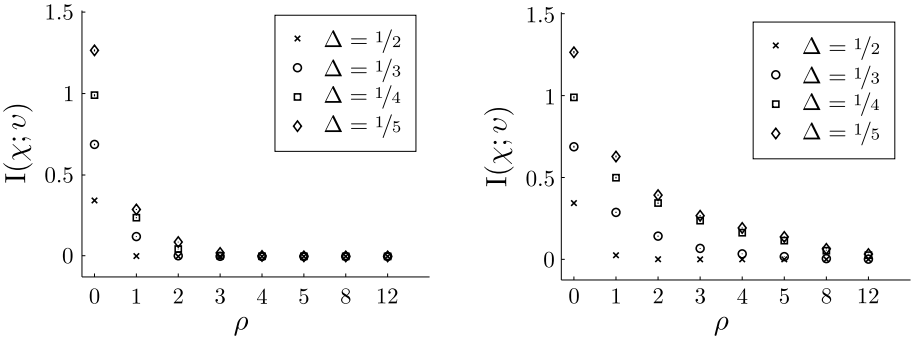
Non-uniform quantisation. Figure 5.4 displays information leakage on profiles and profile components under the adaptive (A) and non-adaptive (nA) strategies, illustrating how we may use a non-uniform quantisation step to capture changes



(a) Adaptive DGS. Profiles

(b) Non Adaptive DGS. Profiles

Figure 5.2: Comparing adaptive and non-adaptive DGSs with information leakage as $\mathrm{I}(X;Y)$, $|V| = 20$ users, skewed profiles.



(a) Adaptive DGS. Profile components.

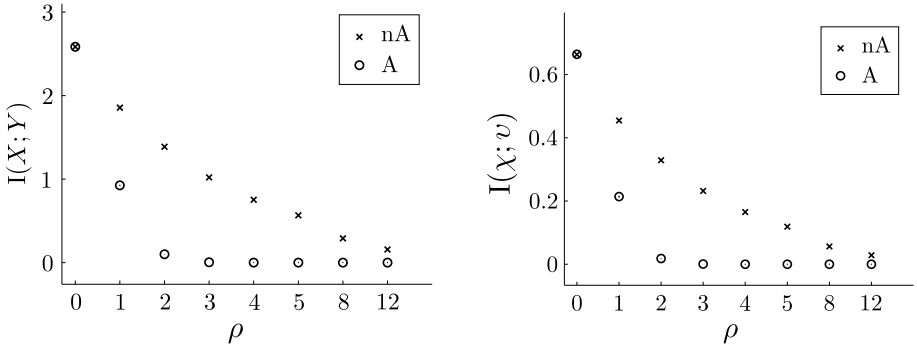
(b) Non adaptive DGS. Profile components.

Figure 5.3: Comparing adaptive and non-adaptive DGSs with information leakage as $\mathrm{I}(\chi;\upsilon)$, $|V| = 20$ users, skewed profiles.

in adversarial goals —in this particular case, to identify a user's best friend. Comparing Figure 5.4 with Figures 5.2 and 5.3, we observe that information leakage decays more slowly for the same dummy rate increases when we use a best-friend non-uniform quantisation step than when we use a uniform binary quantisation step, demonstrating how none of the two DGS designs we consider optimises the allocation of dummy messages to conceal best friends. Indeed, none of the DGSs we consider selectively attempts to conceal any particular component in profile $\mathbf{x}$ and, as a result, they conceal weaker components first

and the strongest profile component (the *"best friend"*) last, requiring the largest budget of dummy messages.

Figure 5.4 further confirms that the adaptive DGS performs significantly better than the non-adaptive and that even if adversaries gain more information by considering full profiles rather than individual profile components, a profile component evaluation provides a good and less computationally expensive approximation.



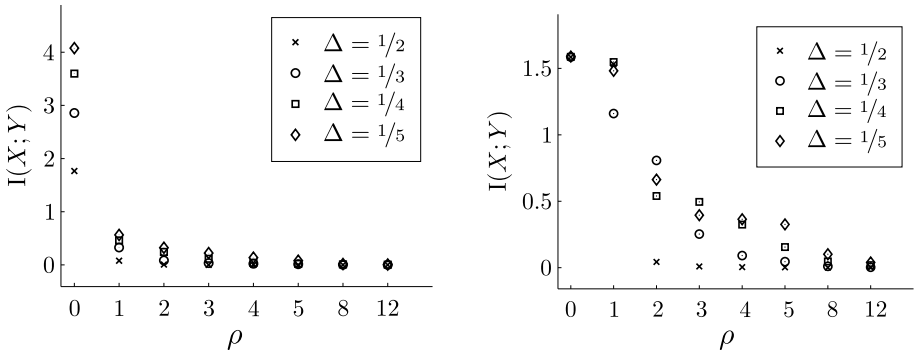
(a) Profiles (b) Profile components

Figure 5.4: Mutual information for non-uniform quantisation for adaptive (A) and non-adaptive (nA) strategies. $|V| = 20$ users, skewed profiles.

Network topology and user behaviour. Lastly, we examine the impact of social graph topology and user behaviour on DGS information leakage. We ran simulations on the full-meshed network of size $|V| = 4$ for both skewed and worst-case profiles, using the adaptive DGS.

Figure 5.5 illustrates how DGS effectiveness depends on user behaviour, capturing the expected negative effect that *worst-case* profiles have on information leakage. Figure 5.5b further shows that at dummy rate $\rho = 0$ diminishing the quantisation step has no effect on information leakage; since users only communicate with one of their friends, regardless of the quantisation step only two quantisation intervals have samples, namely, the ones corresponding to $x = 0$ and $x = 1$. We note that at selected dummy rates Fig. 5.5b reveals the existence of quantisation artefacts, namely, larger information leakage for coarser quantisation steps. Still, mutual information converges for all quantisation steps as the dummy rate increases.

Figure 5.5 also shows that information leakage decreases faster for skewed profiles than in previous experimental settings with a not fully connected network of



(a) *Skewed* profiles (b) *Worst-case* profiles

Figure 5.5: Effect of topology and user behaviour. $|V| = 4$ users, fully connected topology.

size $|V| = 20$ (cf. Fig. 5.2a). These results demonstrate that DGS effectiveness inversely decreases with the number of friends a user has on the OSN; the DGS needs to distribute the same budget of dummies among a larger set of people, thereby having less resources to cover strong profile components.

Conclusion.

In this section we have shown how to compute mutual information to measure a DGS's information leakage. We have tested several experimental parameters, varying the DGS, network topology and user behaviour. Our results show that mutual information captures changes in information leakage accordingly, e.g. showing greater leakage for the non-adaptive DGS or less leakage for skewed instead of worst-case profiles.

Because both the simulator and implementation we use to compute mutual information are not optimised, they remain computationally intensive. Hence, we consider networks which are orders of magnitude smaller than actual OSNs to obtain results in reasonable time. Nevertheless, we show that it is possible to speed up mutual information computation through strategic selection of metadata and quantisation. Our results indicate that the decrease of information leakage with increasing dummy rates is very similar for profiles and profile components, hence we may perform the analysis uniquely on profile components for preliminary DGS evaluation in larger networks. Considering coarser quantisation intervals also provides very similar information to more

fine-grained intervals. Hence, we can rely on coarse quantisation to speed up DGSs evaluation.

5.3.2 Side channel leakage evaluation

Batina et al. argue that *"side-channel analysis can be seen as the problem of detecting a dependence between [two random] variables"* [53]. Like previous work on side-channel leakage analysis in cryptographic implementations [237, 505], we use mutual information to evaluate information leakage from side-channel sources of OSN metadata.

In the previous section we have examined techniques to simplify mutual information computation that enable us to perform preliminary DGS evaluation. We apply these same techniques to examine correlations across different sources of metadata and determine which ones we need to consider in DGS design.

We consider an OSN where users engage in two main types of communication: private and public. Private communication is E2E-encrypted, namely, only the sender and designated recipients can access E2E-encrypted messages' content. Public communication on the other hand is accessible to everyone, with *everyone* denoting a users' friends, all OSN users or everyone on the net, depending on the access control policy that both SNP and users define. We consider a cProto to provide CPC for private communication and examine the leakage of information about private communication that other OSN sources of metadata such as public communication or the OSN's topology reveal.

Experimental setup.

Dataset. We perform our study using Balsa et al.'s dataset from Belgian OSN *Netlog* [40].[7] The dataset comprises interaction metadata from the Dutch-speaking subnetwork in Netlog, e.g. the sender, recipient and time of the messages users exchange, but not messages' content.

We select the following metadata for our analysis:

Friendship requests and acceptances. We consider that two users Alice and Bob are friends when the dataset contains a friendship request from Alice to Bob and a friendship acceptance from Bob to Alice.

[7] As of 2015, Netlog is no longer in service.

Public posts. Public posts are messages that users leave on other users' personal pages in the OSN and are publicly available for other Netlog users to see.

Private messages. Users send private messages to select recipients that, unlike public posts, are only visible to their senders and recipients through a private *inbox*.

We note that as public posts are not E2E-encrypted, an adversary can also exploit their content. However, we choose not to take post content analysis in our analysis and focus exclusively on traffic data.

Moreover, we further note that Netlog did not implement E2EE, meaning that Netlog's service provider had access to all communication in plain text unless users unilaterally deployed encryption (q.v. Sect. 5.4.1). Still, we *simulate* encryption by disregarding private messages' content.

Selected metadata. We consider the following random variables:

Over private messages,

- the number of messages Alice sends to Bob, $M_{\mathrm{A,B}}$, and
- the number of messages that Alice and Bob exchange, regardless of whether Alice is sender or recipient, M_{AB}.

Over network topology features,

- the number of friends Bob has, F_{B};
- the number of friends Alice and Bob have in common, $F_{\mathrm{A}\cap\mathrm{B}}$;
- the number of friends that Alice and Bob have in total, namely, the cardinality of the *union* of their sets of friends, $F_{\mathrm{A}\cup\mathrm{B}}$, and
- the Jaccard coefficient over Alice and Bob's sets of friends, $\mathrm{J}(F_{\mathrm{AB}})$, which equals the number of friends Alice and Bob have in common divided by the number of friends they have in total, characterising the similarity between their sets of friends [539].

Over public posts,

- the number of posts Alice leaves for Bob, $Z_{\mathrm{A,B}}$, and

Data source	Features	Visibility
Private messages	M_{AB} ; $M_{\mathrm{A,B}}$	Private
Friends graph	F_{A} ; $F_{\mathrm{A}\cap\mathrm{B}}$; $F_{\mathrm{A}\cup\mathrm{B}}$; $\mathrm{J}(F_{\mathrm{AB}})$	Public
Public posts	$Z_{\mathrm{A,B}}$; $Z_{\mathrm{B,A}}$	
Posts graph	$F^{Z}_{\mathrm{A}\cap\mathrm{B}}$; $F^{Z^{*}}_{\mathrm{A}\cap\mathrm{B}}$; $F^{*Z}_{\mathrm{A}\cap\mathrm{B}}$; $F^{Z}_{\mathrm{A}\cup\mathrm{B}}$; $F^{Z^{*}}_{\mathrm{A}\cup\mathrm{B}}$; $F^{*Z}_{\mathrm{A}\cup\mathrm{B}}$	

Table 5.2: OSN features in correlation analysis.

the number of posts Alice and Bob exchange, regardless of whether Alice is sender or recipient, Z_{AB}.

Over network topology features that Alice and Bob's *posting friends* induce, namely, those friends Alice and Bob send to or receive posts from,

the number of mutual friends who either Alice or Bob posts to of receives posts from, $F^{Z}_{\mathrm{A}\cap\mathrm{B}}$;

the number of mutual friends who either Alice of Bob posts to $F^{Z^{*}}_{\mathrm{A}\cap\mathrm{B}}$;

the number of mutual friends who either Alice of Bob receives posts from $F^{*Z}_{\mathrm{A}\cap\mathrm{B}}$;

the total number of friends who either Alice or Bob posts to of receives posts from, $F^{Z}_{\mathrm{A}\cup\mathrm{B}}$;

the total number of friends who either Alice of Bob posts to $F^{Z^{*}}_{\mathrm{A}\cup\mathrm{B}}$, and

the total number of friends who either Alice of Bob receives posts from $F^{*Z}_{\mathrm{A}\cup\mathrm{B}}$.

Moreover, we model some random variables over different time periods to determine whether a longer or shorter history of available user behaviour information leads to better correlations. We denote random variables over alternative time periods with a superscript T, e.g. Z^{T}.

Table 5.2 summarises the features we choose to examine.

Sampling. We use the sampling method we describe in Sect. 5.3.1. However, rather than sampling from simulated data, we *"sample"* from the Netlog dataset.

We compute probability $P(\,(M,Z) = (m,z)\,)$ by counting the number of occurrences '$c(m,z)$' of each pair of values (m,z) and dividing it by the total

number of samples σ. Since a finite number of samples introduces an error in the estimation, we model $P(M, Z)$ as a multinomial distribution and use Bayesian inference to obtain a bound on this error as we describe in Sect. 5.3.1.

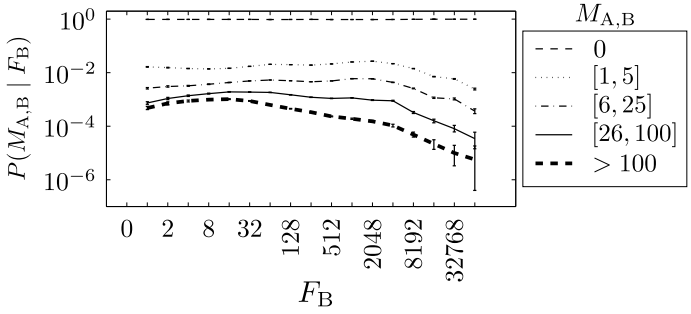
We take median values of mutual information as estimated values. To estimate error, we consider the interval containing 99% of the values around the median.

Results.

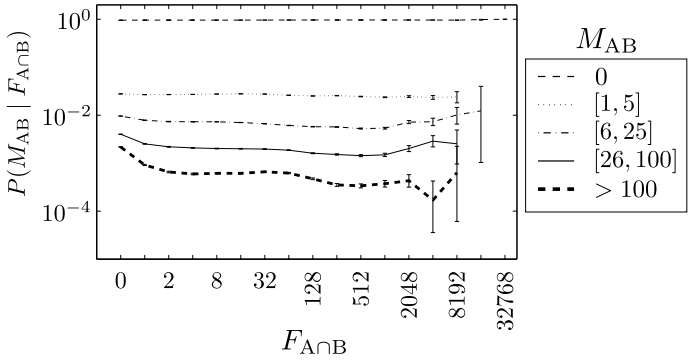
We present the results of our experiments. Unless we state otherwise, all figures in this section follow the same representation formula. They display conditional probability distributions of the random variable we intend to obfuscate (e.g. number of messages Alice sends to Bob) given the random variable over the side-channel source (e.g. the number of friends Alice and Bob have in common). The abscissa or horizontal axis represents values of the random variable over side-channel metadata whereas the ordinate or vertical axis represents conditional probability values. The figures also feature error bars whenever significant, representing the standard error on a 99% confidence interval.

Private messages given network topology features. Figure 5.6 shows the probability of the number of messages two users exchange given a number of topological features in their local network. Figure 5.6a shows that the number of messages Alice sends to Bob is independent from the number of friends Bob has. In fact, the entropy of the random variable over the number of messages Alice sends to Bob equals $\mathrm{H}(M_{\mathrm{A,B}}) = 0.2044$ bits, whereas conditioned on the number of friends Bob has it barely drops to $\mathrm{H}(M_{\mathrm{A,B}} \mid F_{\mathrm{B}}) = 0.2037$ bits (i.e., $\mathrm{I}(M_{\mathrm{A,B}}; F_{\mathrm{B}}) = 0.0007$ bits). Hence, the number of friends Bob has does not leak information about the number of messages Alice sends to him.

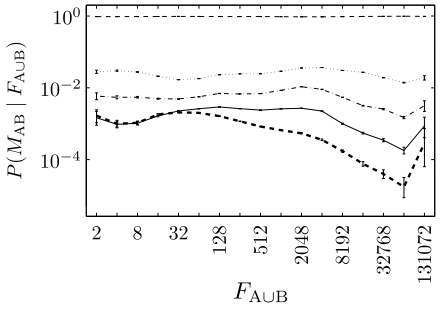
Similarly, neither the number of friends they have in common, the number of friends they have in total or the Jaccard coefficient over their sets of friends provides information about the number of private messages two users exchange, as Figures 5.6b, 5.6c and 5.6d attest, respectively. The probability of any number of messages stays relatively constant for numbers of mutual friends below 1024. Beyond that number the error increases significantly —as few users have more than 1024 mutual friends—, but with no indication of a potential change in trend. Table 5.3 provides the list of information leakage values for the topological features we evaluate, supporting the results in Fig. 5.6.



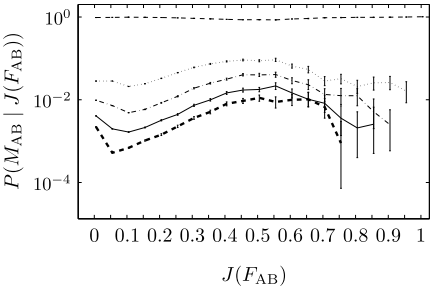
(a) Probability of number of private messages sent given number of recipient's friends



(b) Probability of a number of private messages exchanged given number of mutual friends



(c) Probability of a number of private messages exchanged given union set of friends



(d) Probability of a number of private messages exchanged given Jaccard coefficient over friend set

Figure 5.6: Probability of number of private messages given graph topology.

	Bits
Ref.: $\mathrm{H}(M_{\mathrm{AB}})$	0.2751
$\mathrm{I}(M_{\mathrm{AB}}; F_{\mathrm{A}\cap\mathrm{B}})$	0.0006
$\mathrm{I}(M_{\mathrm{AB}}; F_{\mathrm{A}\cup\mathrm{B}})$	0.0017
$\mathrm{I}(M_{\mathrm{AB}}; \mathrm{J}(F_{\mathrm{AB}}))$	0.0013

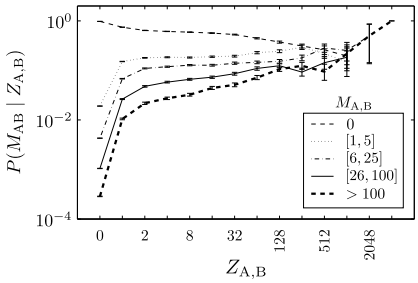
Table 5.3: Information leakage of various topological features.

Private messages given public posts. Figures 5.7a represents the probability that Alice sends a number of messages to Bob given the number of posts she writes to him in the same period of time (i.e. 6 months), whereas Figure 5.7b represents the probability of the number of messages Alice sends to Bob given the number of posts Alice receives from Bob in the same period. Both figures show that the probability that Alice sends one or more messages to Bob significantly rises when Alice leaves a post on Bob's wall or she receives a post from him, steadily increasing for even larger numbers of posts. However, the number of posts Alice sends to or receives from Bob does not precisely determine the number of private messages she sends to him, as the probability of sending a particular number of private messages is similar for any number of posts (despite the increasing gradient in Figures 5.7a and 5.7b, which is negligible considering the logarithmic y-axis).

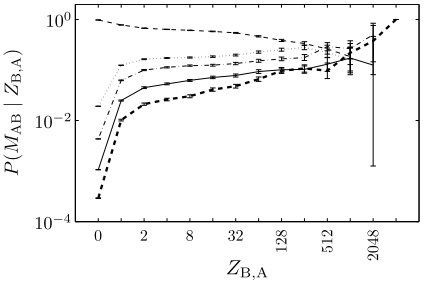
Figure 5.7c represents the probability that Alice sends a number of private messages to Bob in a 6-month period given that Alice leaves a number of posts for him in the previous 9 years, whereas Figure 5.7d represents the probability that Alice sends a number of private messages to Bob on a 6-month period given that Bob leaves a certain number of posts for her in the previous 9 years. In both cases, the probability that Alice sends messages to Bob increases with the number of posts, yet the correlation between number of posts and private messages is weak, suggesting that communication profiles are unstable, that previous posting history is not as reliable a predictor of recent messaging behaviour as recent posts.

Table 5.4 shows the mutual information between the random variables we examine in Fig. 5.7, confirming that the number of posts two users exchange does not provide significant information about the number of private messages they exchange.

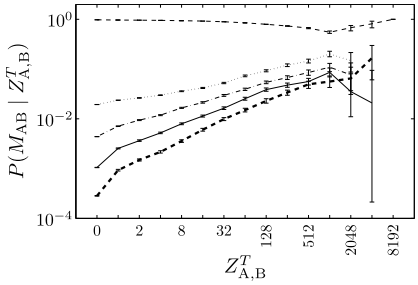
Private messages given posting friends. Figures 5.8a and 5.8b show that the number of posting friends Alice and Bob have in common provides little



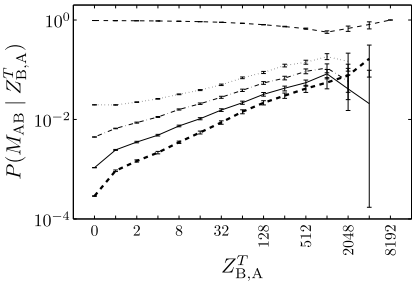
(a) Probability of the number of sent messages given number of sent posts.



(b) Probability of the number of sent messages given number of received posts.



(c) Probability of the number of sent messages given number of posts sent over 9 years period.



(d) Probability of the number of sent messages given number of posts received over 9 years period.

Figure 5.7: Probability of messages sent given sent and received posts.

information about the number of private messages they exchange. When Alice and Bob have more than one posting friend in common, the probability that they exchange at least one message increases, yet does not provide sufficient information to determine the precise amount of private messages they exchange. We obtain similar results considering the union of posting friends, namely, those friends that either Alice or Bob have sent a post to or received a post from, in Fig. 5.8c. Table 5.5 shows information leakage from posting friends, confirming the results in Figure 5.8 —including further experiments which we do not include a figure for due to the similarity across results.

	Bits
Ref.: $\mathrm{H}(M_{\mathrm{A,B}})$	0.2044
$\mathrm{I}(M_{\mathrm{A,B}}; Z_{\mathrm{A,B}})$	0.0055
$\mathrm{I}(M_{\mathrm{A,B}}; Z_{\mathrm{B,A}})$	0.0048
$\mathrm{I}(M_{\mathrm{A,B}}; Z^{\mathrm{T}}_{\mathrm{A,B}})$	0.0013
$\mathrm{I}(M_{\mathrm{A,B}}; Z^{\mathrm{T}}_{\mathrm{B,A}})$	0.0011

Table 5.4: Information leakage of public posts.

	Bits		Bits
Ref.: $\mathrm{H}(M_{\mathrm{AB}})$	0.2751		
		$\mathrm{I}(M_{\mathrm{AB}}; F^{Z_{\mathrm{A,B}}}_{\mathrm{A}\cap\mathrm{B}})$	0.0025
$\mathrm{I}(M_{\mathrm{AB}}; F^{Z_{\mathrm{AB}},\mathrm{T}}_{\mathrm{A}\cap\mathrm{B}})$	0.0007	$\mathrm{I}(M_{\mathrm{AB}}; F^{Z_{\mathrm{B,A}}}_{\mathrm{A}\cap\mathrm{B}})$	0.0030
$\mathrm{I}(M_{\mathrm{AB}}; F^{Z_{\mathrm{A,B}},\mathrm{T}}_{\mathrm{A}\cap\mathrm{B}})$	0.0007	$\mathrm{I}(M_{\mathrm{AB}}; F^{Z_{\mathrm{AB}},\mathrm{T}}_{\mathrm{A}\cup\mathrm{B}})$	0.0015
$\mathrm{I}(M_{\mathrm{AB}}; F^{Z_{\mathrm{B,A}}}_{\mathrm{A}\cap\mathrm{B}})$	0.0021	$\mathrm{I}(M_{\mathrm{AB}}; F^{Z_{\mathrm{A,B}}}_{\mathrm{A}\cup\mathrm{B}})$	0.0014
$\mathrm{I}(M_{\mathrm{AB}}; F^{Z_{\mathrm{AB}}}_{\mathrm{A}\cap\mathrm{B}})$	0.0039	$\mathrm{I}(M_{\mathrm{AB}}; F^{Z_{\mathrm{B,A}},\mathrm{T}}_{\mathrm{A}\cup\mathrm{B}})$	0.0021

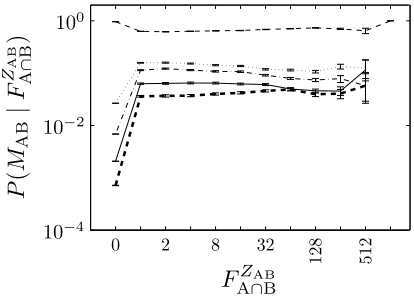
Table 5.5: Conditional entropies given posting friends sets.

Conclusion.

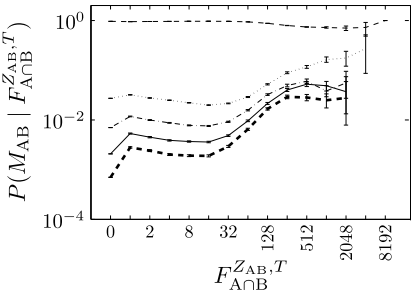
The results above show that the side-channel sources of metadata we examine leak no information about the number of private messages OSN users send or exchange, suggesting that, at least in this particular OSN, it is possible to disregard users' public posts and certain properties of the subgraph they induce to generate dummy messages and provide private-message CPC.

Still, given the limited sample of metadata we examine, we must not assume the complete absence of correlations between the sources of information we have evaluated. Other metadata variables such as the specific time users send a post or the time elapsed since the previous post may leak more information. Moreover, these results are specific to Netlog and we cannot generalise them or claim the absence of correlations between these random variables for every OSN.

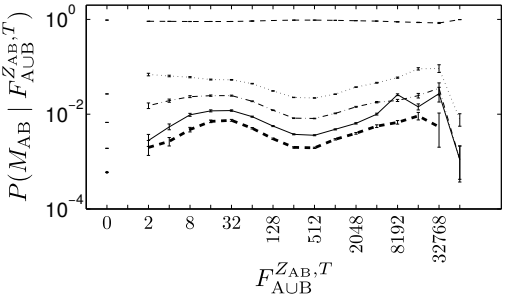
Lastly, whereas we acknowledge that it is impossible to perform such measurement for every source of metadata in OSNs, the method we propose



(a) Probability of a number of exchanged messages given number of mutual posting friends.



(b) Probability of a number of exchanged messages given number of mutual posting friends (over 9 years period).



(c) Probability of a number of exchanged messages given union of posting friends set size (over 9 year period).

Figure 5.8: Probability distributions of exchanged messages given posting friends.

enables us at least to confirm or rule out *expected sources* of side-channel leakage rather than blindly enlisting or excluding them from DGS design.

5.4 E2EE in SNSs

Whereas the cProtos model assumes that users exchange E2E-encrypted messages, currently E2EE is not available for most SNS users. Instant messaging applications such as *Signal*, *Telegram* and *Whatsapp* have successfully adopted E2EE in recent years, yet popular SNSs like Facebook, Twitter, LinkedIn

or Flickr, to name a few, do not provide E2EE yet.[8,9] Despite numerous breaches into their systems [36, 370, 395, 556, 557] and promises old and new to move towards E2EE, most SNSs do not protect users' content with E2EE [250, 582], hiding behind claims such as too much complexity and usability problems [250, 384].[10]

E2EE does come with challenging key distribution and key discovery problems, e.g. such as those that underlie PGP's complexity and usability problems as an E2EE mechanism for email. However, unlike email providers, who need to ensure interoperability between users of other providers, SNPs have the ability to unilaterally push changes in their respective OSN platforms as well as the ability to leverage the implicit trust and social relationships between their users [530].

Of course SNPs also have strong incentives to avoid the deployment of E2EE as it threatens their business model, based on the monetisation of user data for targeted, behavioural advertising. Moreover, pressure from governments and law enforcement agencies to guarantee unfettered access to users' data further discourages the adoption of E2EE, even if increased calls for content moderation and responsibility on the spread of misinformation provide counterincentives to retain access to content [250, 403].

At the same time, in response to the numerous privacy breaches in SNSs and SNPs' inaction to strengthen their users' privacy, researchers and developers have advanced several proposals to enable E2EE in SNSs, be it through third-party tools like browser plugins that users can deploy on the site, or through alternative designs that propose new OSN platforms altogether [35, 57, 256, 301, 361]. However, these tools and alternatives have seen little adoption in practice [4, 531].

[8]We note that some may argue that communication services like instant messaging or email implicitly constitute OSNs too. However, we consider a narrower definition of OSN, namely, boyd and Ellison's, defining OSNs as *"web-based services that allow individuals to (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"* [92].

[9]Facebook provides E2EE as an option on its messaging mobile application *Messenger*, while its subsidiary company, messaging mobile application *Whatsapp*, encrypts E2E by default. Interactions on the Facebook site are not E2E-encrypted.

[10]In June 2014, in the wake of the Snowden revelations and much hype around end-to-end encryption, Google launches a project to bring E2EE to its email service, Gmail. In February 2017, Google announces it discontinues the project, after almost 3 years and scant results [250]. In March 2019, Mark Zuckerberg, Facebook's CEO, announces a move to bring E2EE to Facebook [582]; however, this move seems to exclusively apply to the messaging services of Facebook and its subsidiaries (WhatsApp, Instagram), rather than the whole platform [298, 403].

Security and privacy experts have attempted to explain why OSN's users lack interest in privacy technologies. Some argue that there is little interest in privacy enhancing technologies (PETs) because these tools conceptualise privacy as secrecy and secrecy is *"antithetical to the notion of social interaction"* [275]. Others point at the poor understanding of privacy problems in OSNs that users have [373]. Despite the abundance of user studies about privacy in OSNs, the perceptions and attitudes of users towards TPETs for SNSs remain largely unexplored [531].

In the first half of this section we investigate the role that SNPs may play in E2EE deployment, the advantages and disadvantages with respect to usability and security. We consider three different possible roles SNPs can play in enabling E2EE and examine the consequences of each of these roles for the design and deployment of cProtos. We conclude that SNPs benefit from an unparalleled position to deploy E2EE in their respective social networking platforms, whereas TPETs face multiple usability and deployment obstacles. However, the unparalleled position that SNPs enjoy also enables them to easily undermine E2EE, whereas TPETs rely on and benefit from distributed trust.

Hence, in the second half of this section we present the results of a user study on user perceptions towards TPETs for SNSs. We analyse participants' responses to identify obstacles that may impede the adoption of these tools, thereby questioning the viability of TPETs as a vehicle for cProtos.

5.4.1 SNP as E2EE provider

A successful implementation of E2EE provides confidentiality, authenticity and protection against man-in-the-middle (MitM) attacks. In this section we consider three roles online SNPs can play in providing E2EE, outlining advantages, challenges and threats, as well as discussing the implications for cProtos deployment.

SNP as E2EE provider.

The SNP runs E2EE either integrated on the SNS or using client-side scripting. The SNP is a trusted central authority that provides each user with a public-private key pair or the means to generate them, e.g. through integrated client-side code that runs on users' browsers. The SNP works as certificate authority (CA) and central directory, namely, it certifies, stores and ensures the availability of users' public keys. Users' public keys are linked to users' OSN profiles, i.e. to user's identities in the OSN. Users manage and store their own private keys

and verify the authenticity of other users' profiles, while the SNP ensures the availability and authenticity of the public key linked to every profile, thereby lessening the complexity of key management and its impact on users' experience.

Challenges and responsibilities. Users may lose their private keys or access the SNS from different devices, hence the SNP may add support for key recovery, backup and synchronisation mechanisms, e.g. providing a public-private key pair per device, similarly to iMessage [336].[11] The SNP may enable users to encrypt their private keys with a password (e.g. the one they use to log in to the SNS) and store them on SNP servers so that users can later download them to other devices, similarly to *Firefox Sync*'s synchronisation mechanism [496]. The SNP may also implement an identity-based encryption (IBE) system defining users' OSN identifiers (IDs) as public keys [87], thus becoming a private key generator with the ability to regenerate private keys at any given time, enabling users to synchronise new devices or restore lost keys. IBE further reduces the complexity of key management by enabling OSN users to select as their public key relatable IDs such as their e-mail address or telephone number [58]. However, we note that IBE does not allow users to reuse IDs that became revoked public keys, e.g. a user whose revoked public key was her email address can no longer use her email address as a new public key. To support revocation it is possible to attach an expiration date to each ID [87].

Possible threats. This model requires users to rely on the SNP as a trusted party. As IBE private key generator, the SNP can decrypt all private messages. If key generation happens client-side, the SNP may introduce backdoors that allow it to covertly transfer a copy of all messages or private keys back to its servers [416]. In both cases, law enforcement may compel SNPs to retrieve and expose private keys, thereby defeating the very purpose of E2EE. Moreover, as CA, the SNP can certify its own keys and use them to impersonate users, launching hard-to-detect MitM attacks. SNPs can alleviate users' concerns through transparency, relying on schemes such as *certificate transparency* to distribute trust among CAs [342].

SNP as participant in federated ID-based public key generation.

The SNP is part of a *federation* of multiple, independent *semi-trusted* entities that implement a multiple-trusted authorities IBE scheme [119]. Each entity of the federation acts as a private key generator (PKG) and follows a distributed key generation (DKG) protocol based on verifiable secret sharing (VSS)

[11] Alternatively, Whatsapp, currently the most popular E2EE implementation, exclusively relies on users' phones as key storage and single decrypting device. However, the Whatsapp model is inadequate to enable users general access to SNSs from different devices, as it requires users to keep their phones connected at all times.

which enables them to generate a *master secret* so that at least t entities in the federation (assuming one secret share per entity) are necessary to reconstruct it [58].

Upon user request and based on a public key the user chooses (e.g. her SNS ID), each entity uses its own share of the master secret to provide a *share* of an ID-based secret key. Each user's private key thus results from the combination of θ out of n shares, one per each independent entity in the federation. The user collects one share from each entity and computes her private key client-side. The ultimate goal is to prevent any of the providers or a coalition of less than θ of them from retrieving the user's private key, so that only users can retrieve their own private keys [58].

Because each share is ID-based, each entity can provide its share from the public key at any time. This way, in the event of private key loss or upon using a new device, users can easily retrieve their private keys from θ of the share holders. A federation of providers thus offers the convenience of IBE-based key management while averting the threats that a single SNP acting as CA poses.

Challenges and responsibilities. Whereas distributed IBE provides convenient key management, it still requires the selection of k entities to generate users' secret keys. Holding users responsible for the manual selection of these entities represents a usability challenge, therefore user-friendly methods must assist users in the selection process. In particular, to reduce the risk of θ or more entities *colluding* to reconstruct users' private keys, the selection of share holders must ensure that they operate under different jurisdictions to avoid that governments are able to coerce enough share holders to recover users' private keys. Moreover, users must *authenticate* with each of the share holders to prevent unauthorised parties to collect shares of their secret keys.

Possible Threats. Whereas this model shifts user trust from one single trusted SNP to several *semi-trusted* federated entities, it is still vulnerable to collusion of t or more SNPs. Hence, key security factors include the number n of different entities, their incentives to collude, as well as the existence of an entity that can coerce SNPs to disclose their shares, e.g. a government or supranational entity.

SNP as supporter.

The SNP supports third-party tools that provide E2EE encryption in OSNs, typically browser plug-ins that independent developers and researchers design, implement and maintain [57, 256, 326, 361]. Developers must release and maintain plug-ins for different browsers, ideally for both desktop and mobile. Since developers typically rely on limited time and resources to ensure constant,

smooth tool maintenance and keeping up with constant browser and OSN updates is hard, the SNP pushes changes to the OSN platform with care, transparently, minimising impact on the plug-in.

To preserve user experience (UX), TPETs capture each HTML page before the browser displays it to the user, decrypt any encrypted content (that the user is able to decrypt) and replace it with its corresponding plaintext. The SNP facilitates this task by providing parsing functionalities through an API. An API provides a stable, black-box way of parsing the site that developers can rely on independently from changes to the HTML.

The SNP further helps advertise and boost the visibility of these plug-ins, e.g. similarly to Facebook's new service trials, the SNP can recruit a subset of users for early testing [509].

Challenges and responsibilities. With weaker SNP involvement, TPETs need to overcome additional challenges. First, regarding key management, TPETs require a mechanism that allows users to verify the authenticity of public keys, rather than expecting non-tech-savvy users to import public keys from a key server and perform offline fingerprint verification à la PGP [248, 480, 551]. TPETs must deal with key verification, loss and revocation effectively. Popular E2EE messaging applications such as Whatsapp or Signal offer ease of verification through QR code scanning of a *"safety number"*, yet user studies have shown that users still fail to understand and therefore successfully execute the verification process [1, 281, 475]. OTR suffers from similar usability issues [506]. Alternative proposals include key verification services based on key transparency and monitoring such as CONIKS, whereby users' clients monitor name-to-key bindings to detect malicious or bogus key directories entries [379]. SNPs can support the deployment of key verification services by acting as identity providers, auditors and key directories, storing users' public keys to guarantee their availability and authenticity.

Management of private keys presents further usability challenges. TPETs typically generate public-private key pairs on the client side, storing users' private keys on their devices, which they implicitly assume to be trustworthy and secure. Hence, users need to install the tool on every browser and device they use to access the social networking site and synchronise or import private keys across browsers and devices, a challenging process for non-expert users. Tool developers can enable support for multiple key pairs, one for every browser and device, yet this requires Alice to send messages to Bob encrypted with all his public keys, and vice versa, adding further complexity to public key management. To prevent private key loss and to ease access and storage, TPETs could encrypt private keys with a password and upload them to the OSN; however, users typically choose insecure passwords [9] and tend to forget them [214]. Developers may

opt for using personal mobile phones as storage, yet this represents a brittle alternative at best, as phones can unexpectedly stop working and are easily lost and stolen.

User adoption represents another challenge. Whereas SNPs can seamlessly integrate E2EE in the OSN, even turning it on by default, TPETs require that users find and install them. Users may not be aware of the existence of these plugins or realise that E2EE tools provide better protection than what the SNP already offers [1, 39]. Moreover, users need their friends to also install the tool, thereby posing a bootstrapping problem as early adopters are unable to use E2EE until they convince their friends to adopt it too. Third-party tools therefore depend on users' ability to find them, to understand them and to socialise their use.

Interoperability between E2EE tools from several third parties represents yet another challenge, potentially leading to a landscape of tools so fragmented that users require several tools to communicate with their OSN friends [69, 529]. Developers can tackle this problem by agreeing on an open standard to base their tools on [460].

Possible threats. SNPs can abuse and subvert their supporting role for third-party tools in several ways. SNPs can compromise the integrity and the availability of encrypted messages by altering or removing ciphertext from their servers. If an SNP serves as trusted public key directory, it can attempt to impersonate users through a MitM attack; with no verification mechanism in place —e.g. if users do not check fingerprints out of band— this attack goes undetected.

Discussion.

Any of the three SNP roles we outline above enables or supports the deployment of E2EE, thereby helping privacy engineers to build cProtos on top. However, not every SNP role aligns with the cProtos' adversary model.

If the SNP acts as either private-key generator or CA, it can undo E2EE and defeat a cProtos that relies on encryption for content indistinguishability. Hence, from a security and privacy perspective, this model is unacceptable, as the SNP can impersonate OSN users and decrypt their messages, plus it spreads a false sense of security among users, deceiving the public under claims of E2EE.

A federation of SNPs offers additional guarantees by distributing trust and offers the convenience of having SNPs as providers while lessening the trust users need to place on them. A federation also requires cooperation between SNPs thus incentivises the deployment of standards that ensure interoperability.

Third-party tools on the other hand minimise the trust that users need to place on any SNP even further, but at the expense of diminished usability, convenience and support.

Regardless of whether SNPs or third parties provide a client-side E2EE tool, its code must be open source. Otherwise, users need to trust the developers as they trust the SNP in the first model. Code auditing and oversight to ensure that the tool is free from backdoors and unintentional vulnerabilities represents an additional challenge.

We consider cProtos as independent third-party tools that users can install to achieve CPC; cProtos designers may choose to either *enhance* existing standalone TPETs with CPC capabilities or build separate cProtos that rely on E2EE available in the SNS. One of the key underlying Protos design principles however is to enable users to protect their privacy without the service provider's cooperation. Hence, in the next section we examine users' attitudes to TPETs in SNSs. We aim to study users' perceptions of E2EE in SNSs and the use of TPETs that cProtos designers may choose to extend.

5.4.2 Attitudes towards encryption in SNSs

We hypothesise two main problems motivate the adoption of TPETs:

- *Insufficient protection against the SNP.* Either because the SNP is malicious or negligent of privacy.
- *Inadequate access control tools.* Because the access control tools available at the SNS are insufficient or inadequate (e.g. Facebook's *privacy settings* do not enable arbitrary access control policies).

Throughout this section we refer to these reasons as *access control problems* (ACP) and formulate the following hypotheses:

H1. OSN users are concerned about ACPs.

H2. OSN users feel responsible for addressing ACPs themselves.

H3. OSN users should use a (technical) tool to address ACPs.

H4. OSN users understand and agree with how TPETs address ACPs.

To validate these hypotheses, we perform a user study consisting of two questionnaires and a guided tour to a TPET, namely, *Scramble!* [57].

Design and setup. We validate hypotheses H1, H2 and H3 through an entry questionnaire of 36 questions, of which we analyse a subset that we list in Appendix A.1. We validate hypothesis H4 through a guided tour to *Scramble!* and an exit questionnaire of 11 questions (see Appendix A.3). In addition, we use the system usability scale (SUS), a 10-item attitude Likert scale widely used in usability studies [96], to evaluate *Scramble!*'s usability.

Because we want participants to evaluate how *Scramble!* solves ACPs (H4), right before the guided tour we introduce them to ACPs on Facebook (see documentation in Appendix A.2). Afterwards, we give participants a *manual* to use *Scramble!*, including both an introduction to what *Scramble!* is, how it works and the instructions for the guided tour. The guided tour involves all the steps a new *Scramble!* user needs to follow to use the tool, including download, installation and instructions to encrypt messages. We encourage them to send encrypted messages to other participants.

We run the study in a laboratory environment during the first week of September 2013. At arrival, we instruct participants to log into one of 8 available computers in the laboratory and send them an email with a link to the entry questionnaire. They have 15 minutes to complete it, at which point they receive the documentation for a 30-minutes guided tour to *Scramble!*. Then, they receive a second email with the link to the exit questionnaire, which they have 10 minutes to complete.

We invite 52 students (42% female, average age = 21.5, SD = 2.6) from the Center for Behavioral Decision Research Pool at Carnegie Mellon University to participate in a study to "Test Scramble! – A Facebook app." We pay participants $10. All participants have been using Facebook for at least 2 years and identify as active Facebook users.

Survey analysis. We rely on *emergent coding* [343] in two rounds of analysis to process participants' responses. In the first round we develop *in-vivo* codes, namely, terms the participants themselves provide that summarise the concept they refer to. In the second round we use thematic, hierarchical coding to group *in-vivo* codes into themes and themes into broader, more general themes.

Results.

We present the results of our study. We use "quotation marks" to refer to questions in the questionnaire and both *"italics and quotation marks"* to refer to participants' responses. We place participants' quotes between parentheses when we provide several examples of a certain attitude, perception or position.

We do not aim to provide quantitative results, yet we mention the number of participants that articulate a position to indicate whether it represents a majority or a small minority.

Concern about ACPs (H1). We ask participants "Which privacy problems, if any, do you encounter using Facebook?" (Q13) to find out whether users refer to ACPs (H1). We classify participants' responses in two categories: *lack of control over how information flows* and *lack of control over how others use their information*. Within the first category, participants express concern over Facebook's *privacy settings* (*"privacy settings always seem to be reset to a lower level after each update"*) and their granularity (*"[...] would be very helpful to be able to decide exactly who can see each post / friend's post / photo / piece of info [...]"*).[12] Other participants refer to the difficulty of deleting information (*"even [if] you delete something still it can be seen by a search tool"*). These concerns are in line with ACPs. However, participants also voice concerns that E2EE cannot address such as a lack of control over somebody else's activities, e.g. *"I am able to stop people from seeing my posts, but not posts I'm tagged in"*.

Similarly, relating to the *lack of control over how others use their information*, participants mention problems that E2EE can solve, e.g. *"my data being used to target ads at me"*, and cannot solve e.g. *"Facebook tracking you across the web"*, *"random people message me even though I don't know them"*, or *"Some random strangers sending me friend requests"*.

We ask participants "What, if anything, would you add to, modify or delete from Facebook's privacy settings?" (Q17). Many participants point to the inadequacy and lack of granularity of Facebook's privacy settings, e.g. *"want to put limit to the photos one by one not the whole album"*, *"cover photos are all public. I would change that"*, *"I would make it possible to hide specific things from specific people"*. Some participants point to enhanced protection from web search, e.g. *"[...] a function that makes your profile unsearchable for a certain amount of time [...]"*, *"[...] that no one can find [my information] even if googled"*. Encryption offers fine-grained access control and disrupts search engines' ability to index content. However, some participants express a desire *"to see who has viewed my page"*, *"if any random user [...] viewed my pictures"*, while another participant requests the opposite *"would never allow people to see the profiles I've looked at"*. Encryption does not offer this type of audit capabilities. Some participants refer to Facebook's terms of service and, more

[12] At the time of the study, Facebook provides privacy controls for items like posts but not for e.g. single photos in an album. Besides, the predefined categories available on Facebook limit a user's ability to manage a particular post's audience, e.g. Facebook users can share a post with a particular subset of *friends* but not a particular subset of *friends of friends*, even if sharing with *all* friends of friends is possible.

specifically, to "what Facebook does with your data". E2EE prevents access to content —not metadata—, thus precludes its collection,

To questions "What privacy issues you have, if any, that you are not able to solve with Facebook's privacy settings?" (Q18) and (if any) "Which strategies do you use to solve those privacy issues?", most participants respond *"none"*. We classify responses from participants who did have issues as:

Lack of control over other people's activities (*"Individuals comment inappropriately on [my] status"*, *"Friend requests from strangers"*), with coping strategies including *"deleting the post"*, *"reporting [to Facebook]"* and *"nothing"*. TPETs do not provide solutions to these problems.

Lack of control or knowledge on Facebook's uses of data (*"to not sell my data to companies"*, *"Tracking you across the web"*, *"Seeing who my 'top friends' are on chat or on my profile"* or *"The adds* [sic] *I see in facebook are related to even my google searches, they interfere in to every space of mine."*). Coping strategies include *"ask explicitly if they are okay"*, *"Firefox add-ons"*, *"nothing"* and *"never login in to facebook"*, respectively. TPETs mitigates some of these problems by denying data access to the SNP.

Lack of control over cover photos with self-censorship as a solution (*"Only post [...] "appropriate" cover photos [...]"*); TPETs address this problem.

Hence, all in all, we observe that users' privacy concerns on Facebook overlap with the reasons that motivate the adoption of TPETs.

Responsibility and control (H2). We assess participants' desire for greater control over their privacy and whether they feel responsible for taking measures to mitigate ACPs. We ask participants "Who should decide...?" for a set of decisions related to the visibility of their data (Q7) such as "...who is able to see what you post on the site" or "...who is able to see your personal details". For all decisions but one, most participants think that they should have control over those decisions themselves, being the average percentage of *You* and *Facebook* across all decisions M = 81.3% (SD = 12.9%) and M = 23.9% (SD = 14.1%), respectively. This desire for control clashes with the participants' unwillingness to be the only responsible for several privacy-related issues [93]. We ask participants "Who should be responsible for the following [privacy related] decisions?" such as "setting the proper privacy settings on your profile" or "making sure private companies do not have access to the data you post to the site without your permission." Most participants consider that Facebook should be responsible for issues such as "making sure your privacy settings

work" or "making sure strangers cannot see your photos/posts online." A few participants even declare that Facebook should be responsible for "setting the proper privacy settings on your profile" or even "making sure your friends do not post photos of you that you do not like." Most users attribute to Facebook greater responsibility than control. Across all decisions, the percentage of *You* and *Facebook* is M = 67.4% (SD = 21.1%) and M = 52.2% (SD = 27.8%), respectively. We notice certain trends in how users assign responsibility. Users attribute to Facebook responsibility for issues such as "preventing strangers from logging in to your account" and "preventing people other than your friends from reading your messages and seeing your photos", i.e. issues that are out of their control by default on Facebook, whereas they attribute to themselves the responsibility for "what your friends can see in your profile", i.e. matters for which Facebook provides privacy controls.

In fact, to the question "On Facebook, what do you feel responsible for with respect to your own privacy?" (Q11), most participants refer to *what they post* (47%) and *how they use the privacy controls that Facebook provides*. For instance, one participant writes: *"[Block people. I]t is then facebook's job to make sure that they cannot message me from that point on"*, or *"I am not the one who can guarantee the execution of [the privacy settings], Facebook does it. At this level what choice do I have? To trust Facebook"*. Participants also feel responsible for the public content they post, but not what they say over private messages. Participants explicitly mention that Facebook's *private domain* falls beyond the scope of their responsibility, e.g. *"I feel responsible for the content of my public posts/comments and photos posted to the public. I feel I should not have to further manage private messages [...] which I want to remain private and have selected as such"*. Users focus on social privacy problems and barely mention surveillance problems [260]; they consistently disregard the fact that privacy settings do not prevent the service provider (Facebook) from being able to access all their content, both public and private, e.g. *"I would rather friends send me private messages if they want to share something fun with me"*.

Hence, even if participants show a desire for greater control over their privacy on Facebook, they indicate no intention of taking matters into their own hands, weakening the case for TPETs' adoption.

Awareness of and attitudes towards alternative privacy controls (H3). We ask participants "Which strategies or mechanisms do you know, even if you do not use them, to prevent unintended recipients from having access to your messages and information you send or post on Facebook?" (Q27). Most participants (81%) respond "None". Other participants point out to strategies such as *"limiting the amount of people [added] as friends"*, tightening

their privacy settings (including blocking people) and *"deleting facebook"*. One participant mentions *"encrypting messages/posts"*. Further, we ask them "Why would you, or would you not, use such a tool?" Most participants simply refer to increased privacy as a reason (*"I like to increase my privacy"*), with some providing more elaborate answers (*"it would add an extra layer of protection against marketering* [sic] *companies and online hackers"*). At this point in the questionnaire however, it is obvious that privacy plays a central role in the study, hence participants' answers may be motivated by a strong social desirability bias [213]. A few other participants show more scepticism, e.g. *"All ready, I'm concerned with one such thing I'm using. I don't want to involve something else and provide my data to more sources"* or *"if it could actually protect me from something I needed to be protected from"*. Participants who respond that they would *not* install such tools (20%) also declare that the tool itself can be unreliable or leak their data (*"I would not like to broadcast my privacy settings. I would use the app only if it remains anonymous."*, *"Not sure if it is safe to install"*) even suggesting that *"they can be unreliable unless facebook certifies the tool themselves"*. Other participants dismiss the usefulness of such tools as *"I control what I share and I trust facebook to a certain extent"* or *"too much effort for a trivial thing"*. In short, the fact that users take advantage of non-technical strategies to manage their privacy and the mistrust in the effectiveness of alternative technical tools seem to offer little support for TPETs' adoption.

Attitudes towards *Scramble!* (H4). We ask participants "Can you describe, in a few words, your experience using *Scramble!*?" (Q36), and they express a wide range of opinions. Many responses focus on the lack of *usability*: a steep learning curve (*"the learning curve took too much time"*), how cumbersome *Scramble!* is (*cumbersome*, *"[*Scramble! *requires] too many steps [...] to send a message"*). Lack of usability explains why *Scramble!*'s SUS falls barely above the middle score (M = 52.9, SD = 18.35, MAX = 95, MIN = 15). Many perceive *Scramble!* to be useful but only for *very private information*, e.g. *"It's a very great idea, but only useful for messages that really needed to be protected"*, *"awesome for the people who want to send private [i]mportant [...] text messages"*, further linking this perception to poor usability, e.g. *"It was effective, but too complex to be integrated into my everyday routine. I am not THAT concerned about my private messages to go through the hassle"*. Some participants call into question the benefits of *Scramble!*, e.g., *"Easy, interesting, not sure about the benefits, though"*. Others describe the experience as both *easy* and *fun*.

We also ask participants "What would be the advantages, if any, of using a tool like *Scramble!* over, or in combination with, other privacy controls?" (Q37). Many participants (70%) give succinct answers (*"more private communication"*,

"better privacy") or simply repeat what they read in the documentation we give them (see Appendix A.2). A couple of participants mention however that there are no advantages of using *Scramble!* or that *"I wouldn't worry so much about my Facebook messages being in the open."* One participant writes that *"There seems to be no advantage, as again facebook has all our public names and email ids associated with that".*

Participants show further scepticism when we ask them "*Scramble!* encrypts messages before you send or post them on Facebook. Do you think this is a secure way to prevent unintended recipients from having access to them?". Even if most of them (77%) simply reply *"yes"*, others express e.g. *"having a simple private and public key mechanism may not be robust enough"*, *"yes, kind of, I am sure they will find another way to decode it"*, *"To a point... unless someone can figure it out and de-crypt it"*, *"that would require a second, and third level of encryption, which I think is illogical".* Other participants do not see cryptography as the source of mistrust, but rather its particular implementation on *Scramble!* and the developers, e.g. *"Probably yes, but remember we don't know whether scramble is a government controlled plug-in or actually developed by Facebook itself"*, *"no, till proper and full information about, what scramble is, how and why it encrypts our data".* Lastly, some participants' concerns derive from a misunderstanding of how *Scramble!* works, e.g. *"what if you accidentally send a message to someone who has scramble but they were an unintended recipient can they still read your message? or do you have to add them to your contact list first?"*, while it does not matter who installs *Scramble!* or who is on the user's contact list as long as the public keys users select to encrypt a message correspond to the intended recipients.

We ask participants "What do you think are the differences, if any, between what *Scramble!* does and the privacy settings of Facebook?". Most participants (70%) refer to the documentation we give them (see Appendix A.2), while some participant admit being unable to understand the differences between what *Scramble!* does and what the privacy settings of Facebook do, e.g. *"I don't know if Facebook uses encryption between two individual users like Scramble does"*, *"im not exactly sure of how facebooks privacy stuff operates".*

To the question, "What, if anything, did you dislike about *Scramble!*?" (Q45), one participant mentions that *"[b]ecause messages sent to me were automatically scrambled, I wasn't sure if the person sent a scrambled message or a normal one. I'm also not sure if anyone with scramble would be able to read a scrambled message, or I would have to have them on my contacts list first".* One participant asks *"[w]hat if someone hacked into my facebook?"* while another wondered *"[whether it would] be possible for third parties to figure out the encryption mechanism".* Both responses suggest a lack of transparency and feedback to

enable users to better understand how *Scramble!* works, contrasting with many participants' demands for higher *automation*.

In short, users' inability to understand neither the type of protection E2EE offers nor why E2EE works, a perception of encryption as too excessive a tool to protect OSN communication and a mistrust in cryptography and third-party tools all weaken the prospect of self-motivated TPETs' adoption.

Discussion.

We have investigated user attitudes towards TPETs to determine whether TPETs provide a suitable platform for designers to build cProtos on top, or rather cProtos benefit from SNP-supported E2EE.

The study results indicate that OSN users' attitudes are in line with the privacy problems that TPETs aim to solve (H1), thus satisfying one of the prerequisites for TPETs adoption. However, participants in our study seem to perceive TPETs as disproportionate and ineffective, at too high a usability cost (H3, H4). Their responses suggest that the burden of learning, adopting and using TPETs offsets the low benefit they perceive E2EE provides.

Participants question the reliability of cryptography and third party tools, showing scepticism towards their effectiveness. Some participants remark that by using TPETs they need to trust the tool developers instead (or on top) of the SNP. TPETs are typically open-source, thus under scrutiny by anybody. Computer scientists rely on this property to justify that one does not need to trust the developers, as anybody can examine the code (and change it) to make sure it does what it is supposed to [286]. This in turn distributes the trust users need to place on a single developer to the whole community. However, the general Internet user may be unaware or unwilling to rely on this property and current TPET designs do not tackle this issue. For other participants, the complexity and obscurity of cryptography prevents them from understanding how E2EE enhances content confidentiality over the mechanisms they already use, as several participants demonstrate through their responses (cf. [1]).

Several authors obtain analogous results in the context of online tracking and third party anti-tracking tools [369, 380, 471] (q.v. Sect. 3.3.3). Hence, we conclude that users are unlikely to rely on TPETs if they do not understand or know how to evaluate the protection these tools promise. In fact, TPETs may leave users worse off if the latter do not understand how the former work. In this study we choose not to test users' ability to authenticate the public keys they use to encrypt messages for each other, a task that users consistently disregard [281, 475].

Participants indicate a desire for seamless, nearly automatic integration between the TPET and the social network (cf. [33]), thus supporting greater SNP involvement in E2EE provision even if greater automation is likely to diminish users' oversight of their own privacy protection. Participants also raise concerns with respect to the trust they need to place on yet another entity; SNPs can enhance their trustworthiness through transparency. From a security engineering perspective however, the less trust users need to place on the SNP, the better. Yet participants' actual perception is different: since they are not familiar with the security advantages of open source code and trust distribution, they are more willing to trust a powerful and popular entity such as Facebook than independent developers they do not know.

Hence, unless designers manage to securely automate some tasks and engage users in the kind of decision-making that TPETs require (e.g. leveraging transparency and user feedback better [1, 443]), enhancing TPETs with cProtos' capabilities seems pointless. SNS users are unlikely to seek TPETs on their own and, by extension, cProtos.

In fact, participants' attitude to TPETs casts doubt on their need for Protos. If OSN users do not understand E2EE or perceive the need for content confidentiality against the SNP it is at best doubtful that they seek CPC, thus inhibiting Protos' adoption [179]. Still, users may find obfuscation more intuitive and easy to conceptualise than cryptography, even fun. cProtos designers may therefore need to investigate avenues for greater OSN integration and SNP involvement, in addition to user engagement [458].

5.5 Discussion

Privacy requirements.

In this chapter we propose two variants of CPC, namely, contact-exposed and contact-hidden. Similarly to our selection of privacy requirements in CBPWS, this selection illustrates one particular choice among many. Users may voice other privacy concerns that require more specific definitions, e.g. concealing a user's k most contacted friends or concealing a user's most stable relationships, thereby introducing timing constraints that we have disregarded throughout this section.

Adversarial assumptions.

cProtos' successful deployment depends on similar assumptions about the adversary to the ones we have examined in Sect. 4.4.3, namely, the adversary must be HbC. Whether or not this assumption holds in practice determines the viability of Protos deployment. However, unlike in CBPWS, where users can single-handedly deploy a CBPWS tool, cProtos require cooperation among several users. If the dummy messages a cProto generates never elicit a response from their recipients, an adversary can filter them out as dummies —unless the user's real messages never obtain a response either. Hence, as users need to cooperate with one another, *by design* there are greater chances to reach a critical mass that forces the adversary to acquiesce with obfuscation.

cProtos design.

There are a host of cProtos design issues that we have not dealt with in this chapter. We have shown how to use information leakage with metadata selection and simplification to perform preliminary DGS evaluation, as well as to measure correlations between side-channel sources of information leakage that inform DGS design. However, we have not addressed the numerous challenges that cProtos DGS design itself poses.

To illustrate the challenges in DGS design, we briefly discuss here two of them. First, cProtos design requires simulating *conversations*. If cProtos send dummy messages which never elicit a response, an adversary can easily discard them as dummies —except in the extreme case where users never obtain any response to their messages. Moreover, real communication follows a set of patterns such as time between messages or number of messages per conversation, among others, which the adversary can exploit to filter out dummies. A user's cProto must therefore coordinate with other users' cProtos to generate plausible conversations, e.g. a cProto cannot unilaterally pursue a target profile $\mathbf{y}^t$ (q.v. adaptive strategy in Sect. 5.3.1), as it requires responses to make the dummy messages it sends indistinguishable from real messages. Moreover, for real-time communication services such as IM, cProtos must deal with the additional constraint that communication partners must be online, e.g. a cProto that assigns a high value to the target profile component y_j^t corresponding to a user v_j that is seldom online is likely to fail to plausibly ensure that $y_j = y_j^t$.

Second, one of the two variants of CPC we propose, contact-hidden, requires that users send dummy messages to people who are not in their contact list, thereby requiring a *"dummy contact"* discovery mechanism for users to send dummy messages to cProtos' users outside of their contact list. Adding dummy

contacts further requires a strategy to select plausible contacts, e.g. randomly adding users that do not speak a common language or have no other friends or context in common may enable an adversary to distinguish between real and dummy conversations.

Hybrid solutions and other alternatives.

In CBPWS we refrain from exploring solutions that involve cooperation with other users —such as those that involve submitting queries in other users' behalf or populate a pool of real user queries— to minimise dependence on third parties and warrant individual users as much autonomy as possible. Conversely, cProtos require cooperation between users by default, which means that solutions that further leverage other users' cooperation impose no additional trust requirements in cProtos.

Systems like Drac rely on a user's trusted friends to provide anonymous communications with a user's (non trusted) contacts and *unobservable* communications with a user's (trusted) friends [153]. Hybrid approaches may thus combine CPO with message relaying through common friends or other third parties —essentially constituting an anonymity system with dummy traffic support [74]— to improve communication profile confidentiality.

Moreover, we have not explored the possibility of setting up Sybil accounts or *benign social bots* to assist cProtos achieve contact-hidden CPC, i.e. enabling users to send dummy messages to fake accounts on top of other cProtos users [210]. The introduction of social bots further undermines profiling by preventing profilers from determining not only which messages are real and dummy but also which users are real and which are dummy. Introducing plausible Sybil accounts is however far from trivial, as a large body of work on Sybil attacks detection demonstrates [157, 540, 567]. Moreover, it also poses additional ethical questions, e.g. regarding interactions with individuals that do not use cProtos or the automatic spread of misinformation [210].

5.6 Conclusion

In this chapter we have studied the deployment of Protos to provide communication profile confidentiality (CPC). We have instantiated the general Protos model in Sect. 3.1 to capture the particularities of online communication. Furthermore, we have proposed a set of privacy properties for CPC, contact-exposed and contact-hidden CPC, and operationalised them using information leakage as introduced in Sect. 3.2.1.

Equipped with this analysis framework, we have proposed a set of techniques to more efficiently compute mutual information and measure information leakage. We have applied these techniques to evaluate the level of CPC a DGS offers in the context of online social networking and measure side-channel information leakage in SNSs, showing the suitability of using mutual information to measure information leakage.

In addition to the performance gain that we obtain through the selection of simpler metadata variables and coarse quantisation steps, there is ample room for additional research on techniques that further speed up mutual information computation. Bayesian inference techniques that use sampling to reduce complexity is one such possible approach, as proposed in prior work on mix networks' analysis [158].

Furthermore, we have studied the role that SNPs can play in enabling E2EE in SNSs, illustrating the trade-offs involved in relying on SNPs for E2EE provision. SNPs stand in an exceptional position to enable E2EE on their platforms; they can seamlessly integrate E2EE and simplify many of the technical and usability challenges related to key management. However, trusting that SNPs do not abuse their position of power on the platform to undermine E2EE is incompatible with the Protos' adversary model, whereas preventing such abuses by technical means entails a host of additional challenges. Third-party E2EE tools (TPETs) seek to address and forsake the need to trust the SNP by distributing trust and oversight to the community of developers and users [286]. However, these tools require explicit user adoption and give rise to numerous usability challenges.

To evaluate TPETs' viability as a solution to bring E2EE to SNS and a platform on top of which we build cProtos, we have studied users' attitudes to TPETs. We conclude that the complexity, users' perception and understanding of E2EE represents a major barrier to TPETs adoption. Opting for E2EE integration on SNSs to minimise the complexity and usability challenges that users need to deal with opens additional challenges with respect to transparency and oversight as well as users' perceptions to cProtos. We do not delve into these issues in this thesis. Future work should explore solutions that account for users' scepticism and mistrust in TPETs through integrated designs that minimise usage complexity while promoting user engagement.

Chapter 6

Engineering privacy through chaff: a profile obfuscator

Privacy research often takes place within the confines of a particular system or application, be it web search privacy, communication confidentiality, location privacy or anti-tracking protection, to name a few. Researchers study these systems, the design particularities that make them problematic for users' privacy and propose solutions in the course of a never-ending arms race where researchers uncover vulnerabilities and propose attacks that in turn lead to further improvements and better solutions.

The general Protos model we provide in Chapter 3, however, shifts the focus from a particular system or application where privacy problems arise, such as web search or communication confidentiality (as we examine in Chapters 4 and 5, respectively) to the defence mechanism itself, this is, chaff. The general Protos model therefore represents an overarching framework that abstracts away domain-specific particularities to focus on the methods that underpin privacy protection through the use of chaff. As a conceptual construct, Protos bring a variety of systems and solutions under the same analytical umbrella, encouraging the study of chaff-based solutions across applications and the emergence of universal design principles that generally apply to the deployment of chaff against profiling.

Eliciting generic privacy solutions and design principles is the main goal of privacy engineering. A novel, emerging research discipline, privacy engineering seeks to lay out the fundamental principles that underlie privacy technology design. Where privacy enhancing technologies (PETs) represent particular

solutions to well-defined privacy problems, privacy engineering seeks to extract the rules, guidelines and techniques privacy experts resort to when they design privacy technologies. In other words, privacy engineering seeks to systematise the knowledge that privacy experts have acquired through long experience and practice designing —and breaking— privacy technologies [262, 477].

The shortage of privacy engineers or software developers with experience on privacy preserving systems' design calls for the advancement of privacy engineering as a means to assist developers and non-experts in incorporating privacy protection in the software development life cycle, either from the beginning, as *privacy by design* mandates, or as a way to improve or amend privacy-invasive systems [143].

In this chapter we contribute to privacy engineering by defining a new *privacy design pattern (PDP)*, namely, a *chaff-based profile obfuscator* (CBOR). CBOR contributes to privacy engineering in several ways. First, it recasts the abstract Protos model as a generic solution that privacy engineers can resort to to tackle profiling, helping designers understand the fundamental principles that underlie the use of chaff as a tool to protect against the privacy risks that derive from profiling. Secondly, defining CBOR as a PDP helps to systematise the knowledge on chaff-based profile obfuscation tools. The pattern makes explicit the relationship between all solutions that rely on chaff to avert the privacy problems profiling poses, thereby promoting a dialogue between designers of Protos across several application domains. Lastly, it contributes to both a privacy pattern catalogue and pattern language by specifying a relationship with alternative patterns, subpatterns and patterns from other systems, applications and domains, e.g. formalising the relationship between patterns in the domain of anonymity systems and chaff-based obfuscation. A pattern language strengthens the ties between the work of privacy engineers across domains, e.g. by unifying terminology and making explicit that some of the privacy solutions engineers in different domains rely on are equivalent and follow the same underlying principles.

This chapter is structured as follows. In Section 6.1 we provide a brief introduction to privacy engineering and privacy design patterns. In Section 6.2 we introduce chaff-based profile obfuscator (CBOR) as a new PDP. In Section 6.3 we discuss the implications and limitations of adding CBOR to the existing privacy pattern catalogue. Lastly, we conclude and provide an overview of future lines of work in Section 6.4.

6.1 On privacy engineering and design patterns

Privacy engineering is an emerging discipline that deals with the study of the methods, techniques and strategies that underlie the design and deployment of privacy preserving systems; it deals with the systematisation of knowledge, models and principles that underpin *"the technological and organisational components that implement privacy and data protection principles"* [155].

As an academic discipline, privacy engineering emerges in part as a response to the lack of expertise and know-how in privacy-aware systems' design [143]. Spiekermann indicates that *"privacy is simply not a primary consideration for engineers when designing systems"* [500], while Lahlou et al. report that systems' designers perceive privacy as *"either an abstract problem; not a problem yet [...]; not a problem at all [...], not* their *problem"* [339], i.e. people seldom code or design systems with privacy in mind.

Research on the economics of privacy justifies the lack of privacy expertise among systems' designers. Among others, Acquisti has argued that, in the absence of regulation, service providers have no incentives to bear the costs of redesigning or amending their systems to address privacy issues, owing to users' biases (incomplete information, bounded rationality and systematic deviations from rationality) and an online business model built around the exploitation of user data [4, 6, 400, 500]. This lack of incentives in turn extends to systems' designers, who have no need to acquire the skills they require to deal with privacy requirements. Moreover, acquiring these skills is hard, as designing for privacy is far from trivial [143, 400]

Recent developments have however altered the balance of incentives, leading to an increased interest in the development and advancement of privacy engineering. Prominent privacy-related incidents such as the Snowden revelations or the Facebook–Cambridge Analytica scandal have contributed to heightened public awareness and scrutiny over online companies' attitudes and behaviour towards user privacy [169, 556]. In the European Union (EU), changes on the regulatory framework that governs the acquisition and processing of online user data through the general data protection regulation (GDPR) have prompted companies to devote more resources to privacy management [204, 288].

Kenny and Borking's early definition of privacy engineering as *"a systematic effort to embed privacy relevant legal primitives into technical and governance design"* conveys the discipline's multidisciplinarity, spanning research fields such as, among others, requirements engineering, policy specification and human-computer interaction (HCI) or software engineering [255]. Across these disciplines —and among other methods such as best practices or guidelines—, researchers have resorted to *design patterns* to capture and share privacy

engineering knowledge across practitioners [124]. *Privacy design patterns* (PDPs), like their architecture or software engineering counterparts, describe general solutions to common or recurrent privacy problems in such a way that it is possible to reuse the solution in different systems or scenarios where those problems occur [81, 229, 285].

PDPs are not PETs; they may describe an abstract, generic PET or subcomponents which multiple PETs rely on. As Bier and Krempel emphasise, whereas PETs represent concrete tools or standards and specifications, privacy patterns are *"technology-independent"* descriptions that apply to *"a variety of use cases with a similar context"* [81]. Hafiz, as one of the earliest proponents of PDPs, highlights that *"[a]lthough each domain has its own PETs, there are design decisions that can be generalized. Understanding these design choices and how they are used to solve a problem in a context benefits PET researchers to develop solutions for new privacy challenges"* [267]. PDPs represent such an instrument, they generalise design decisions underlying multiple privacy solutions to address new privacy challenges.

Doty and Gupta further add that *"[u]nlike guidelines, regulations or best practices, patterns are descriptive, rather than normative, facilitating discussion and debate and providing education rather than insisting on particular solutions or practices. Design patterns are also easily composable for differing situations and at different levels of granularity, while remaining more actionable compared to privacy design principles such as data minimization or transparency"* [183]. Indeed, several authors have acknowledged PDPs' instrumental value to implement privacy by design (PbD), as PbD describes principles that are too vague to enable direct application, while PDPs describe concrete, albeit generalised solutions to build privacy-preserving systems or PETs [262, 285, 528].

Moreover, because privacy patterns ought to be domain-independent, they establish bridges across the various disciplines that privacy engineering encompasses. The relationships and interactions between patterns, their arrangement in hierarchies and the abstraction or merger of design patterns across domains contributes to the emergence of a common *language* of privacy engineering which, in turn, helps prevent *"duplicate effort [and facilitates] communication between developers from different branches of technology"* [81]. However, work on privacy patterns remains limited —with work on privacy languages more limited still [348, 558].

We argue that it is possible to generalise many of the decisions that underlie the design and deployment of Protos into a privacy design pattern. Whereas the use cases we study in Chapters 4 and 5 show that each scenario requires different decisions in the design and deployment of Protos, they also illustrate

that some design concepts such as *supersequences* or *indistinguishability* (such as those we discuss in Sect. 3.3) are applicable to both.

Proposing a pattern for Protos' design provides several advantages. It formalises and further roots a common analytical and design framework across Protos, promoting cross-domain dialogue and the emergence of generic solutions and methodologies that underlie every Proto, as well as a *common language* across Protos and other PETs. As a result, a design pattern facilitates the design of Protos in new situations or scenarios, as developers can more easily obtain solutions by instantiating the pattern.

However, the current paucity of Proto's designs —let alone implementations— limits our ability to derive universal design principles that apply to *any* possible Proto. Chung et al. refer to new, emerging patterns that capture rare solutions as *prepatterns*. They argue that even if prepatterns are *"not set in stone"* and likely to *"evolve over time"*, they represent useful tools towards early systematisation of knowledge and cross-domain communication [124]. We propose such a prepattern. Rather than an attempt to capture the decisions that Protos designers face, the pattern we propose, a *chaff-based profile obfuscator* (CBOR), represents a common framework and vehicle to build a common language around Protos and across other PETs and patterns. We expect CBOR to evolve and grow in detail as more Protos' research and implementations emerge, eventually becoming a complete pattern.

6.2 Chaff-based profile obfuscator

In this section we introduce *chaff-based profile obfuscator* (CBOR, [ˈsiːbɔːʳ]), a privacy design (pre)pattern that describes a generic solution involving the use of chaff to thwart profiling.

We rely on a template consisting of a selection of elements from previous templates, specially those used for the privacy pattern catalogues available online, that we consider of special relevance for the description of CBOR [103, 266, 229, 438, 439].

We construct a pattern language around CBOR through the template's field *related patterns*, specifying the relationships between CBOR and other privacy patterns previously proposed in the literature.

Lastly, a note on terminology. We explicitly distinguish between CBOR and a Proto, a CBPWS tool or a cProto to designate, through the former, the abstract solution that the privacy design pattern captures and, through the latter, actual implementations of CBOR, that is, PETs.

Chaff-based profile obfuscator

Name: *Chaff-based profile obfuscator (CBOR).*

Intent: A chaff-based profile obfuscator automatically generates dummy, fake activity (*"chaff"*) on an online service or set of services on a user's behalf, mixing the dummy actions it generates with the user's.

Profilers monitor and collect users' online activities, processing them into individual *profiles* to derive information about each user. By injecting dummy actions into a user's activity flow, CBORs pollute and degrade user profiles, preventing profilers from obtaining and exploiting users' information.

CBORs thwart profiling and provide *profile confidentiality*.

Also known as: No aliases known.

Motivating example: Online tracking and behavioural profiling.

Advertising networks track Internet users online to gather information about the sites users visit, how long they spend on each webpage, the keywords that catch their attention or the sites they come referred from, among other types of data advertisers use to *profile* users.

To track users online, these *profilers* use a variety of mechanisms, from (third-party) cookies to more sophisticated tracking techniques such as device or browser fingerprinting [199]. The information profilers collect enables them to build *consumer profiles* that they can use to more effectively target advertising. Consumer profiles convey information about how old users are, where they live, their socioeconomic status and what they are interested in, among other personal attributes. Profilers can also package and sell these profiles to others, be it advertisers, companies or governments. Tracking enables profilers to *"measure user engagement and the effectiveness of ad campaigns"* as well as to ensure that *"the same ad is not shown repeatedly to a given browser or user"* [517].

To prevent tracking and profiling online, Internet users can resort to various anti-tracking techniques, such as cookie removal or the use of anti-tracking tools like Ghostery, uBlock or Privacy Badger [369], as well as explicitly signalling that they oppose tracking by enabling the Do Not Track setting on their browsers [193]. Moreover, users may also browse the Internet anonymously using anonymous communication systems such as Tor [176]. Still, these techniques present several limitations. Anti-tracking tools must adapt to new tracking

techniques in an endless arms race and may cause undesirable side effects such as preventing content to load or distort webpages' appearance [369]; nothing forces profilers to honour Do Not Track headers, as they do so voluntarily [193], and the use of anonymous communications such as Tor requires users to cope with slower browsing speeds, the inability to view certain content and other usability challenges [521].

Alternatively, CBOR generates fake, dummy interactions with other websites. The CBOR imposes no changes and has no impact on the websites users visit while it pollutes the information profilers gather about them, potentially rendering that information useless.

Context / Applicability: An individual or set of individuals uses one or several digital services. In so doing, each user explicitly or implicitly generates data that an entity collects and processes into a *profile*, with adverse consequences for the user's privacy.

The service provider offers no protection against profiling and the user has few workarounds or alternatives to that service, e.g. because no other provider offers such a service or because profiling is pervasive across service providers. Network effects and costs (e.g. switching costs) may further shrink the space of alternatives.

The service provider is uncooperative, namely, it does not intend to support or assist in providing a solution to profiling; however, *under certain conditions* (which we discuss in *Forces* below) it does not oppose the deployment of privacy solutions.

Problem: An individual uses an online service generating data, both explicitly and implicitly (metadata), that a profiler collects and processes into a *profile*. Profiling has a negative impact on the user's privacy as it reveals personal, sensitive information that the profiler exploits or misuses to the user's detriment. The service provider does not offer a solution to avoid profiling and there are no alternative services that enable the user to accomplish the same set of tasks without exposing herself to profiling.

We wish to enable the user to continue benefiting from the same online service with no utility loss, while preventing profilers from obtaining her user profile.

Forces.

A solution to the problem above must balance the following forces:

No cooperation from the service provider. The service provider is unwilling to collaborate in deploying privacy safeguards against profiling, be it, among other reasons, because of a lack of incentives, (e.g. due to cost of deploying privacy solutions) or a vested interest in perpetuating profiling (e.g. loss of revenue from profilers).

Risk of denial of service (DoS). The service provider faces a trade-off between consenting to the deployment of antiprofiling technologies and denying service to users of such technologies, e.g. due to the reputational risk involved in mistreating users who oppose profiling.

Cost for the provider. The cost for the service provider plays a major role in determining whether it will consent to or oppose antiprofiling technology. Shifting the cost of preventing profiling to the service provider tips the balance in favour of DoS.

Quality of service (QoS). The user is unwilling to tolerate QoS degradation.

User participation. The user is willing and amenable to install additional software or manage additional tools to protect herself against profiling.

User cost. The solution must not significantly increase user cost.

Minimise third-party trust. The solution must minimise the trust users need to place on external third-parties to escape profiling.

Solution: Provide to the user a tool that automatically generates *dummy* or *fake* user activity, polluting as a result the data the profiler acquires to build profiles. The tool must generate dummy activity that is indistinguishable from user activity to prevent profilers from discarding dummy actions and in a sufficiently large volume and variety to leak no information about the user's activity (see design issues below).

Design Issues.

We distinguish two axes in CBOR design: its dummy generation strategy (DGS) and its user interface (UI).[1]

Dummy generation strategy. The DGS is the set of rules that governs how the CBOR generates dummy actions, when, how many, which type and how to mix them with the user's real actions.

[1]We note that the description of design issues draws heavily from content we have already introduced in Sect. 3.3. We reorganise and include that content here for completeness.

Selection of goal, adversary and measure of success. CBOR tailors the generation of dummies to a particular protection goal, whether it is to conceal certain user actions (a profile component) or disclose no information to the profiler (the whole profile). Designers who implement CBOR must also determine the capabilities of the profiler CBOR withstands and protects against as well as a measure of success to determine the extent to which CBOR meets the protection goal.

The budget of dummies. Each dummy action has a cost in terms of processing power and bandwidth, both the user's and the provider's. The processing power that CBOR requires to generate each dummy limits the number and frequency of the dummies it can generate without slowing down the user computer, in turn degrading user experience (UX). Users may also need to pay for each dummy, e.g. as data on a limited mobile data plan, thus limiting the budget of dummies CBOR can harness. The service provider may also cap the number of actions it accepts from any given user before subjecting them to antibot measures such as captchas.

Action diversity. The type of dummy actions CBOR generates has a direct impact on the amount of information it leaks about a user's profile. In the behavioural profiling example we provide earlier, the user may visit a handful of websites as part of her daily routine. On the one hand, to conceal from the profiler the actual websites the user visits, CBOR simulates visits to additional websites the user does not visit. On the other hand, if the user does not wish to conceal the actual visited websites but only the frequency and volume of visits, CBOR does not (need to) simulate visits to any other websites the user does not herself visit.

User behaviour and indistinguishability. CBOR generates dummy actions which are indistinguishable from real actions. CBOR considers any user action's observable characteristics that enable a profiler to discern real from dummy actions, including content, timing, frequency, among any other data or metadata, as well as the prior probability that a user generates a particular action rather than having been automatically generated. CBOR also considers relationships across *sequences of actions* such as logical order or semantic relationships that a profiler can exploit to discard dummies.

Universe and probability of real actions. Depending on the context, it may not be possible to determine the universe and a priori probability of possible user actions, e.g. in web search, it is impossible to determine a priori all possible user queries and how the probability evolves over time. This has a negative impact over the DGS's ability to achieve indistinguishability between real and dummy actions.

Lack of knowledge on profiling strategy. How profilers build profiles is in most cases unknown, thus designers implementing CBOR must optimise the generation of dummies against an unknown profiling strategy.

User interface. CBOR's UI communicates with the user to ensure a correct operation and meet the user's requirements.

Minimise UX disruption. CBOR avoids or minimises disruption to UX.

Ensure user understanding. CBOR communicates to the user the privacy guarantees it offers, ensuring that the user does not overestimate her level of protection against profiling.

Enable customisation. CBOR allows users to customise the amount of resources they spend on obfuscation, the level of protection they obtain and the kind of protection they obtain, e.g. whether they choose to conceal their usage patterns with respect to usage frequency or the type of activities they perform on the service.

Consequences.

The pattern has the following *positive* consequences:

Benefits.

Profile confidentiality. Adding dummies to the stream of user actions pollutes the data profilers gather about the user, thereby concealing the actual user profile.

Social privacy. Profilers can either choose to discard obfuscated profiles or attempt to remove the impact of dummies to incorporate a *filtered* profile to its database. Regardless of the strategy the profiler chooses —but assuming that profilers are unable to perfectly discard dummy actions—, CBOR reduces the utility of the data profilers collect, in turn undermining profiling in behalf of everyone.

The pattern has the following *negative* consequences:

Liabilities.

Obfuscated profiles' side effects. Profilers may be unaware of or choose to ignore the presence of obfuscated data in their dataset. Hence, the actions and decisions these profiles inform no longer depend on the actual users' profiles, but on an obfuscated version, thereby bringing upon users a series of side effects that their real profile might not have triggered. Whereas these effects may not necessarily be worse than those from real profiles —and even improve outcomes for certain users—, profile obfuscation may also harm users, leaving them worse off than before.

Obfuscated profiles' effect may further spread to other system's users and the online service as a whole, subverting and potentially damaging it [100]. Strategic DGS design can mitigate some of these effects.

Moreover, to inform profilers about the use of obfuscation and dissuade them from further processing users' obfuscated profiles, CBOR may explicitly signal the use of obfuscation by using a header like DoNotTrack's or incorporating any other tag that raises the profiler's attention [193]. We however acknowledge that as online trackers may have incentives to disregard a DoNotTrack header, profilers may also choose to ignore tags that signal the use of obfuscation.

Waste [100]. One may consider the use of chaff wasteful if it draws on valuable resources, be it through the generation of chaff itself (e.g. use of bandwidth and electricity) or through the subsequent processes obfuscated profiles trigger (from filtering and processing dummies to the wasteful decisions dummies inform). Users and designers must decide whether the benefits of chaff-based profile obfuscation (CPO) are worth the cost.

Known uses.

Private web search. Tools such as TrackMeNot (TMN), GooPIR and PRAW rely on dummy web searches and web transactions to pollute the web search profile search engines build on their users [294, 182, 198].

Anti-tracking and anti-behavioural advertising. AdNauseam extends content-filtering, ad-blocking tool *uBlock Origin* with a dummy click generator to obfuscate the profiles that online trackers build on Internet users [295].

Location privacy. Researchers have proposed multiple designs to achieve location privacy through the generation of dummy user locations [274, 321, 356, 358, 411].

Communication profile confidentiality. Balsa et al. propose the idea of a profile obfuscator to provide communication profile confidentiality in social networking sites [37].

Related patterns.

We classify related patterns into two categories: *disclosure* patterns and *feedback and awareness* patterns. We borrow the term *disclosure* from Gürses et al.'s subcategories of data minimisation [262]. Under the *feedback and awareness* category we succinctly list a number of privacy design strategies that focus on HCI, this is, a set of patterns that assist communication between CBOR and user.

Disclosure patterns. The following set of patterns relate to CBOR's ability to minimise information disclosure. Figure 6.1 depicts the relationship between CBOR and these patterns.

Cover traffic [266, 267] or *use of dummies* [438, 439]. In the context of anonymous communications, cover traffic refers to the use of dummy traffic to enhance the level of anonymity the network provides. Cover traffic is therefore equivalent to chaff, dummy traffic or dummy messages and therefore represents a subpattern to CBOR.

Length padding [266, 267]. If the protocol messages that a users' actions trigger on the online service and the protocol messages that CBOR generates differ in size, an adversary can exploit size differences to filter out dummy messages. To avoid that, CBOR uses *length padding* of both real and dummy messages, padding all protocol messages to a unique size or predefined set of sizes. Length padding therefore represents a subpattern to CBOR.

Random wait, delayed routing and batched routing [266]. CBOR may alter the timing of real actions by imposing a random wait between the time the user generates them and the service provider observes them as well as simultaneously send both real and dummy actions in batches. Whereas Hafiz originally formulates these patterns in the context of anonymous communications, the same privacy engineering principle applies to CBORs, i.e. CBOR strategically distorts the temporal patterns of a users' actions to make them indistinguishable from dummy actions. Random wait, delayed routing and batched routing represent subpatterns to CBOR.

Encryption with user-managed keys [439]. This pattern mandates the use of encryption with a user's own keys so that the service provider cannot access user data in plaintext. Encryption not only enables users to store or exchange content

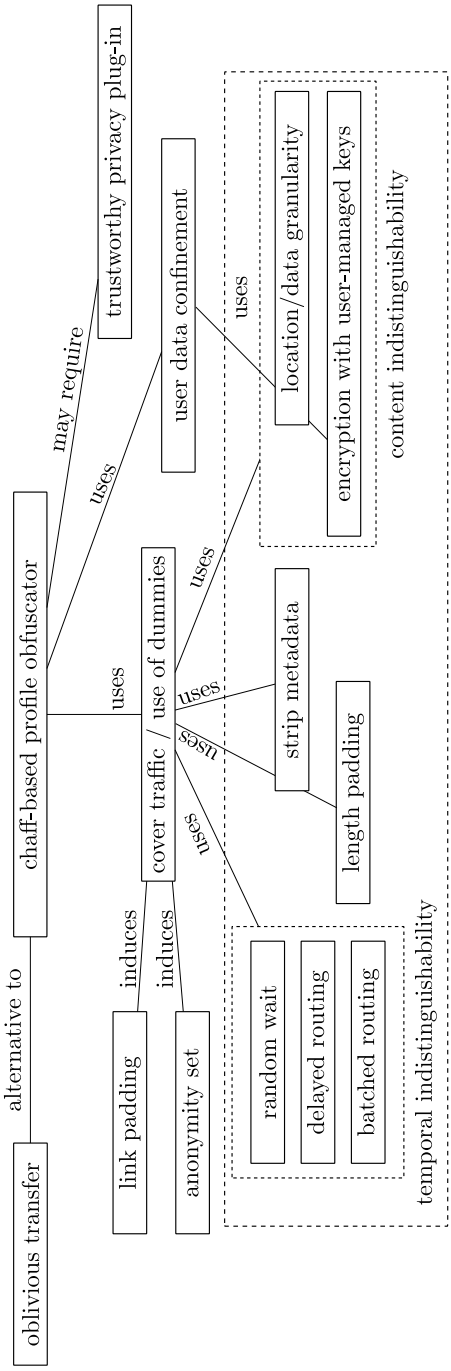


Figure 6.1: CBOR and related *minimise disclosure* patterns.

confidentially, it also enables CBOR to guarantee content indistinguishability. Hence, this pattern is a subpattern of CBOR.

Location granularity [439]. In location based services, location granularity helps users control the level of detail of the information they provide to the service provider. CBOR must similarly manipulate location granularity to achieve indistinguishability between real and dummy locations if the user relies on a privacy preserving mechanism that implements this privacy pattern. Moreover, CBOR may impose a particular regime of location data granularity on the user to ensure indistinguishability between real and dummy locations; however, this may come at the expense of user utility, thus at odds with utility-preserving obfuscation (UPO) principles (q.v. Sect. 2.2.1). Location granularity —as well as a as-yet-to-be-specified *data granularity* pattern that applies more generally to any domain beyond location based services— are subpatterns to CBOR.

Strip metadata [438, 439]. Strip metadata mandates the removal of metadata that is superfluous or unnecessary to preserve user utility. CBOR strips metadata to enhance indistinguishability between real and dummy actions, specially whenever its DGS cannot plausibly generate dummy actions with indistinguishable metadata and such metadata has no impact on user utility.

Link padding [266]. Researchers have proposed the use of link padding to prevent traffic analysis attacks. Link padding involves modifying traffic patterns between two entities —e.g. the number of protocol messages and their timing— to prevent an adversary from learning information these entities intend to conceal, e.g. as in website fingerprinting [192]. Link padding resorts to dummy messages to ensure a constant or variable traffic rate [192, 218]. CBOR resorts to strategies analogous to link padding, i.e. it manipulates the traffic patterns of the protocol messages it relays (both real and dummy) to prevent an adversary from distinguishing dummies. Length padding thus represents a subpattern to CBOR.

Anonymity set [266, 438, 439]. An anonymity set designates the smallest set of individuals to whom we may assign the authorship of a particular action or message, e.g. in anonymous communications, the sender of a message is not identifiable within an anonymity set of potential senders of that message, whereas in database anonymisation an anonymity set designates the set of data subjects whom an anonymous data record may belong to. Similarly, CBOR's DGS generates *supersequences* of actions by interweaving real and dummy actions, thereby engendering analogous relationships to an anonymity set. As each supersequence contains several subsequences among which only one is real, the set of possible subsequences represents a privacy construct analogous to an anonymity set. Whereas the notion of identity or anonymity does not apply to CBOR analysis, from the point of view of privacy engineering they represent an analogous privacy design pattern, namely, hiding data *by mixing it with*

data from other sources [266]. Hence, anonymity set represents, by analogy, a CBOR subpattern.

Trustworthy privacy plug-in [438, 439]. Proposed in the context of privacy-preserving smart metering [438, 451], this pattern represents a privacy-preserving strategy that applies to a much wider variety of domains. A trustworthy privacy plug-in mediates interaction between users and a privacy-invasive service provider, manipulating the data that users generate and send to the service provider to minimise the privacy risks that derive from those data. A standalone implementation of CBOR necessarily requires a trustworthy privacy plug-in (e.g. a browser extension) that users can deploy to generate dummy activity on the online service of their choice. Hence, trustworthy privacy plug-in is a subpattern of CBOR. However, as a component of a wider privacy-preserving system, CBOR does not necessarily require a trustworthy privacy plug-in — although it may still be part of one if the system itself is a plug-in.

User data confinement [438, 439] and *personal data store* [439]. These patterns mandate shifting the processing and storage of user data from the service provider to the user, i.e. in a client-server architecture, move the processing of user data from the server to the client, preventing the service provider from accessing any of the data involved. CBOR may rely on these patterns to counter utility losses that derive from the use of obfuscation, e.g. in web search, CBOR may counter the loss of personalisation by building a search profile and running a personalisation algorithm on the client side.

Oblivious transfer [267]. Oblivious transfer represents an *alternative* privacy design pattern to CBOR in several contexts, e.g. in private web search, a *cooperative* search provider may implement an oblivious transfer protocol to enable users to privately query the search engine, this is, so that the provider cannot determine what users search for. Furthermore, there are conceptual similarities between both privacy design patterns in that CBOR *forces* a weakened version of 1-out-of-n oblivious transfer upon uncooperative service providers by generating n actions out of which only 1 is the user's real action.

Feedback and awareness. Several authors have proposed PDPs to better communicate to users the privacy risks involved in data collection and processing, as well as to design privacy controls and inform users about them [183, 212, 455]. However, these patterns focus on the description of engineering principles that a *data processor* can adopt according to a privacy-by-policy approach, rather than the privacy-by-architecture approach that CBOR represents [285, 501]. CBOR design considers data processors adversarial and the privacy policies the latter implement insufficient or abusive [172]. Still, we can recast some of the design patterns that enable a data processor to inform users about privacy risks

or that enable a user to manage its privacy preferences to assist CBOR interface design. In fact, CBOR implicitly takes on a data processor role by mediating communication between users and profilers; therefore, some of the engineering principles that assist a data processor to communicate with users can assist CBOR design too. CBOR may further act as substitute for a non-cooperative or adversarial data processor by informing individuals of the privacy risks involved in using a service and how chaff-based profile obfuscation can help them mitigate such risks. Hence, we select a sample of feedback and awareness patterns, illustrating how they apply to CBOR design. These patterns point in turn to other auxiliary or compound patterns that we do not examine here; we refer the reader to the relevant literature instead [438, 439].

Increasing awareness of information aggregation [439]. This pattern mandates to *"provide users with knowledge of data aggregation's ability to reveal undesirable information to prevent them from over sharing* [439]" and therefore applies to CBOR design as a mechanism to provide users feedback on the amount of information they leak and on the dummy budget they must allocate to prevent profiling. CBOR may present users with several hypothetical profiling scenarios, showing them what a profiler may learn about them and the protection options the CBOR offers.

Impactful information and feedback, *awareness feed* and *privacy mirrors* [439]. CBOR must incorporate an awareness feed component that informs the user of any limitations to the protection CBOR offers, informing the user of actions or behaviour it cannot obfuscate, e.g. a chaff-based private web search (CBPWS) tool with no protection against *vanity searches* must inform the user about the consequences of such searches [495].

Privacy aware wording [439]. This privacy pattern mandates to communicate to users *"privacy related information using easily parsed and low difficultly vocabulary, with short concise sentences and enough flow to persuade the user to process it"*. CBOR must incorporate privacy aware wording to enable users to understand the protection it affords, its limitations and how to configure the available options to maximise their privacy protection.

Icons for privacy policies / (Appropriate) Privacy icons / Privacy labels / Privacy colour coding [439]. These design patterns describe the use of icons to simplify complex privacy policies, communicating to users data processing practices in an easy-to-understand, unambiguous way. CBOR may also resort to privacy icons to, on the one hand, simplify a service provider's privacy policy whenever the latter is too complex for users to understand and, on the other hand, communicate to users the effect of obfuscation on their privacy, according to the DGS or privacy budget they select, if applicable.

Classification.

First, we consider CBOR a high-level PDP because, unlike other lower level PDPs, an instantiation of this pattern results in a standalone PET. Secondly, as Hafiz et al. note, there are multiple classification schemes for security (and therefore privacy) patterns [268]. We may consider classification across lines of the security or privacy property a pattern aims to guarantee (e.g. confidentiality) its context (application and involved stakeholders) or problem domain (i.e. the threat model). In this pattern description, we consider the two classification schemes for privacy design patterns that we are aware of, namely, Gürses et al.'s *data minimisation strategies* and Hoepman's *privacy design strategies* [262, 285].

Regarding Gürses et al.'s data minimisation strategies, CBOR relies on the *minimise disclosure* strategy [262]. This pattern highlights the relevance of Gürses et al.'s subcategories of data minimisation strategies because, paradoxically, it adheres to data minimisation principles by providing *more data* to profilers.

Regarding Hoepman's privacy design strategies, CBOR implements the *hide* strategy, even if it relies on multiple auxiliary subpatterns that implement other strategies, such as *minimise* (e.g. by using *"strip metadata"*) or *inform* (e.g. by using *"increasing awareness of information aggregation"*) [285].[2] More particularly, according to Colesky and Hoepman's *privacy tactics*, a subcategorisation of privacy design strategies (PDSs), CBOR implements the tactic *obfuscate* [135].

6.3 Discussion

Limitations.

Design patterns capture knowledge that derives from long experience, abstracting well-tried solutions for recurrent problems. CBOR should ideally be based on a large number of chaff-based profile obfuscation tools and systematise the underlying engineering principles behind these solutions. However, because of the limited number of Proto's designs and implementations, CBOR cannot but represent just a first proposal or approximation to that goal. As a *prepattern*, CBOR requires changes and updates as more research on Protos

[2] Hafiz et al. highlight that it is difficult to find a classification scheme in which all patterns neatly fit in, specially for more general or abstract patterns, which often straddle multiple categories [268].

and implementations thereof materialise. Through that process, CBOR requires additional research on its applicability and suitability as a pattern [348].

We do not intend developers with no previous knowledge on privacy engineering to be able to implement a Proto with the information CBOR provides. We have not provided any pseudocode or implementation guidelines that developers may use as guidance for their own instantiation of CBOR [229]. This is however a limitation that most privacy patterns suffer from, as by themselves they are *"not sufficient to provide insight into the process through which [they] can be applied"* [262].

In its current definition, CBOR's value resides in its ability to bring Protos together under the same analytical framework and nudge researchers into thinking about chaff-based obfuscation tools collectively, about the common design principles and building blocks that underpin them, as well as interlinking patterns across subfields in privacy engineering. This will in turn allow other researchers to reuse the same design principles in other scenarios, preventing them from incurring in the same mistakes others made before, as well as contribute to a common privacy engineering language.

In their critical assessment of privacy patterns research, Lenhard et al. argue that there is little research on how to connect several patterns in the development process [348]. We have specified the relationship between CBOR and previously defined patterns, mostly subpatterns that intervene as a building block in Protos implementation. However, we have refrained from specifying how these subpatterns interact within CBOR or how CBOR interacts with other patterns as a subcomponent of a larger system, e.g. that integrates patterns for anonymity or data security. Still, because CBOR represents a high-level privacy design pattern that captures a standalone solution rather than a small component to multiple systems, we do not consider the definition of interaction with other high-level patterns a priority.

Building on top of previous patterns.

We have specified the relationship between CBOR and previously defined patterns, even if the latter often *"lack [...] detail and clarity in their descriptions"* as well as *"vary strongly in their precision and their level of abstraction"* [348], this is, we have not attempted to redefine or amend existing patterns, we have integrated them in their current form, even if ill-defined.

As a matter example, let us consider two patterns: *"use of dummies"* —of paramount importance as a subpattern to CBOR— and *"location granularity"*. In its current definition, *use of dummies* is both vague and flawed, e.g. defining

its *'Context'* as *"applicable when it is not possible to avoid executing, delaying or obfuscating the content of an action"* when, in fact, CBOR may rely on content encryption and action delays to ensure indistinguishability between real and dummy actions. Moreover, as we have shown in Sect. 2.4.1, the use of dummies extends beyond profile obfuscation to applications such as steganographic hiding and mimicry —applications that the current definition of this pattern does not acknowledge.

As for *location granularity*, this pattern describes the operation of increasing and decreasing the amount of precision in location data as a trade-off between utility and privacy. However, in privacy engineering such an operation applies to any other type of data as long as it is possible to represent data values in varying levels of precision or abstraction. It would therefore be more appropriate to refer to a more generic privacy pattern *"data granularity"* instead of *Location granularity*; however, such a pattern has not been formalised yet.

Moreover, as we note earlier, the patterns we have classified under the category *"feedback and awareness"* have been previously cast as solutions for the data controller, within a *privacy-by-policy* approach to privacy engineering [439, 501]. However, many of these patterns describe general HCI methods that privacy engineers can repurpose in *privacy-by-architecture* PET design.

We may argue that building CBOR on top of faulty patterns is unwise, yet all patterns require further evaluation and validation, CBOR included, therefore we leave this task for future work [348].

6.4 Conclusion

In this chapter we have introduced CBOR, a new privacy design pattern that describes a generic solution to thwart profiling through the use of *chaff*. CBOR recasts the abstract model we propose in Chapter 3 as a generic solution that privacy engineers can resort to to tackle profiling, capturing the knowledge and experience we have acquired through the study of chaff-based profile obfuscation in Chapters 4 and 5.

CBOR contributes to *privacy engineering* in several ways. It establishes a common framework for the study, design and development of chaff-based profile obfuscation tools, encouraging researchers and developers across domains to think about Protos collectively, reusing components and adapting solutions from other domains to their own, e.g. even if the CBPWS and communication profile confidentiality (CPC) use cases we have examined in Chapters 4 and 5 impose different design requirements, CBPWS tools and CPC tools share the

same general Proto's underlying principles, such as reliance on *action diversity* that ensure coverage for real sequences of actions among dummy sequences and the *probability of real actions* to prevent adversaries from identifying real sequences due to the low probability of dummy sequences.

Furthermore, CBOR contributes to a common language for not only Proto's developers but also for privacy engineers at large. We have shown how privacy design patterns formulated in the context of anonymous communications are analogous to design principles that underlie Protos, such as *anonymity set* or the various operations to ensure indistinguishability between reals and dummies, such as *length padding* and *batched routing*. CBOR thus contributes to the harmonisation of privacy design patterns across application domains.

Still, due to the paucity of existing Proto's designs and implementations from which we can draw general design principles, CBOR represents a *prepattern*, i.e. it does not offer a complete description of a solution that developers can implement to address profiling. CBOR requires further research on the components and subcomponents on which it depends, as well as validation from developers that attempt to use it in practice. A mature privacy pattern could however become the basis for standardisation efforts, e.g. leading to a standard such as ISO/IEC PDTR 27550 in privacy engineering [299].

Chapter 7

Conclusion

> *Some will say that all we have are the pleasures of this moment, but we must never settle for that minimal transport; we must dream and enact new and better pleasures, other ways of being in the world, and ultimately new worlds.*
>
> —José Esteban Muñoz,
> *Cruising Utopia: the then and there of queer futurity.*

> *It's been important for me to realise that hopelessness is a feeling, it's not a fact, and it actually has very little bearing on what's to come.*
>
> —Anohni.

In this final chapter we review the main findings we have contributed with, discuss unresolved problems and identify avenues for future research. We start by revisiting the seven main objectives we have set at the beginning of this thesis (q.v. Sect. 1.2). We examine the extent to which we have addressed them and lay out the limitations of our findings. Then, we identify gaps, discuss further implications of our research and suggest promising lines of research to be addressed in future work.

Our first objective was to delimit the conceptual boundaries across obfuscation tools in computer security and privacy and develop a conceptual framework that enables us to focus on a particular subset of obfuscation tools, namely, those that protect users' privacy against profiling without taking a toll on user utility. We have identified three main subcommunities devoted to the study of obfuscation

methods and tools —namely, software engineers, cryptographers and privacy engineers— and examined the meaning that each subcommunity attributes to obfuscation, outlining key characteristics of the notion of obfuscation within that subcommunity. From our examination we have shown that both software engineers and cryptographers work on a well-defined, concrete problem, that of code obfuscation, yet their approach is different in terms of the methods they use and the security guarantees they seek. On the other hand, privacy engineers rely on obfuscation as a means to achieve a panoply of privacy properties, lacking the focus of a single goal (e.g. code obfuscation) or set of obfuscation methods. We have therefore argued that within privacy research the meaning of obfuscation is vague and context-dependent, and there is no consensus or universally accepted definition of what obfuscation entails or means, either in terms of privacy goals or mechanisms of protection.

Whereas the vagueness of obfuscation as a process prevents us from identifying a closed and well-defined set of obfuscation tools, we have not attempted to delimit or constrain the notion of obfuscation within privacy research and practice. Instead, we have proposed an abstract model that enables us to focus on one particular type of obfuscation-based privacy enhancing technology (ObPET), namely, those that rely on obfuscation as data degradation. This categorisation has enabled us to distinguish between data-degradation tools such as TrackMeNot (TMN) or differentially private mechanisms and other obfuscation techniques such as onion routing or traffic morphing [176, 319].

Moreover, we have introduced the concepts of personal utility and adversarial gain to establish a separation between two types of obfuscation tools: utility-degrading obfuscation (UDO) tools that seek to strike a balance between personal utility and adversarial gain, utility-preserving obfuscation (UPO) tools that minimise adversarial gain taking no toll on personal utility. By further introducing the concept of social utility we have reasoned about the conditions that call for using either type of tools. On the one hand, the provision of personal utility alone does not impose trade-offs between utility and privacy, thereby enabling the use of UPO. On the other hand, the provision of social utility *necessarily* requires UDO to provide robust meaningful privacy guarantees against adversaries with arbitrary background knowledge.

We have examined the conditions that favour the use of tools that rely on obfuscation as data degradation as opposed to other privacy enhancing technologies (PETs) such as cryptographic solutions or anonymous communication systems, highlighting the role of uncooperative system providers that prevent the deployment of cryptographic protocols such as private information retrieval (PIR) or the role of service providers that impose user identification as a precondition to using their services, among other hindrances to the deployment of these PETs.

Lastly, we have identified *chaff* as a key protection mechanism to enable UPO, that is, the automatic generation on a user's behalf of fake, dummy activity that bear no relation to the utility user expects from their own activity while degrading the quality of the data adversaries exploit. We have reviewed the computer security and privacy engineering literature to examine previous uses of *chaff*. Through this exercise we have distinguished between the use of *dummies* and *decoys*, the former denoting fake activity that contributes to *hide* or *disguise* information, the latter denoting fake activity that lures adversaries into traps to simultaneously protect an asset and acquire further information about the adversary. Furthermore, we have highlighted the role of the adversary model to distinguish between the use of dummies to openly hide information and the use of dummies to *undetectably* hide information, that is, through mimicry, as a form of steganographic hiding.

The conceptual framework around ObPET that we have proposed establishes a clear boundary between the kind of obfuscation tools we aim to study, i.e. tools that rely on data degradation, as well as the mechanisms of obfuscation, i.e. tools that rely on *dummies*. We acknowledge however that this is not the only categorisation possible, that other conceptualisations and categorisations of obfuscation tools may provide additional insights and prove instrumental in the advancement of the theory of obfuscation. Future work must therefore further study the relationship between different types of obfuscation, their properties and their uses across computer security and privacy engineering.

Our second objective was to provide an analytical framework that enables the systematic study of Protos. To that end, we have proposed a general model of Protos that abstracts away from a particular service or system of application. As part of the model, we have defined a reference adversary, which we have assumed to be honest-but-curious (HbC) as a prerequisite for the viability of Protos. Through our analysis of previous work on the measurement of the privacy level that obfuscation tools afford we have introduced a toolbox of privacy measures that assist Proto design and evaluation. We have categorised these measures according to their abstraction level, that is, the number of assumptions on adversary knowledge that underlie their application. This has enabled us to distinguish two groups of privacy measures: mechanism-centred analysis (MCA) measures, which abstract away from particular adversaries and attacks, and attack-centred analysis (ACA) measures, that focus on particular adversaries, attacks and instances of background knowledge.

Our third objective was to develop a conceptual framework to assist the design of profile obfuscation tools (Protos). To that end, we have articulated the design of dummy generation strategy (DGS) around the concept of *supersequences* [545]. We have introduced the common shortest supersequence problem that underlies

optimal DGS design and examined the trade-off between the budget of dummies and the level of indistinguishability a particular budget allows.

Our fourth objective was to assist Protos' design in terms of usability. Due to the scant number of previous work on Protos' usability, we have examined the literature on the usability of privacy and reviewed previous findings on the usability of similar tools that extrapolate to Proto design. Moreover, we have performed a user study on the viability of third-party E2EE tools (TPETs) as a platform on top of which to implement Protos. Our results have supported previous findings on the usability challenges involved in the deployment of end-to-end encryption (E2EE) tools, third-party plug-ins and browser extensions.

Our fifth objective was to demonstrate the viability and adequacy of the analytical and conceptual frameworks we have proposed for Protos' analysis and design. We have done so through two use cases, private web search and communication profile confidentiality. We have instantiated the general Protos' model to chaff-based private web search (CBPWS) tools and communication profile confidentiality (CPC) tools, and proposed a set of privacy properties in each of these contexts, using the Protos' analytical framework we have proposed to operationalise them. We have leveraged the Protos' analysis and design frameworks to revisit previous CBPWS designs, deconstructing these designs and exposing the flaws that render them vulnerable to attack, as well as highlighting fundamental challenges that underlie CBPWS tool design, e.g. the immeasurable space of user queries and the epistemic asymmetry between adversaries and Protos designers. In addition, we have proposed a set of techniques to speed up the computation of information leakage measures in the context of communication profile obfuscation tools' (cProtos) evaluation and illustrated how information leakage can inform the selection of features a DGS must consider, thereby further assisting DGS design.

Our sixth objective was to set the basis of a design methodology for chaff-based profile obfuscation. We have proposed a new privacy design (pre)pattern, a *chaff-based profile obfuscator*, as a generic solution that privacy engineers can resort to to tackle profiling. Through this pattern we have captured the knowledge and experience we have so far acquired through the study of chaff-based profile obfuscation, thereby setting the basis for additional advances in Protos' design. Chaff-based profile obfuscator (CBOR) further contributes to privacy engineering by promoting a common Protos' design language and making explicit the link to privacy patterns used in other PETs. However, as a *pre*pattern, we acknowledge that CBOR is a preliminary result and it requires further contributions from Protos' designers, thereby opening up a promising avenue for future research.

Our seventh and last objective was to identify major gaps in our work as well as future avenues of inquiry. In addition to the shortcomings we have documented throughout the thesis, below we discuss further implications of our research, exposing lacunas and promising lines of research.

7.1 Discussion and outlook

In this thesis we have developed a conceptual and an analytical framework to enable the systematic analysis and design of Protos. Whereas the toolbox of measures we have proposed assists Protos' design, we have not delved into the practical development and implementation, i.e. we have neither proposed a new Proto nor implemented one. Our main contribution thus involves the critical examination and systematisation of previous work on Protos, with a special focus on measurement and analysis. However, Protos' design and implementation involves many practical challenges that we have not addressed in this thesis, such as the generation of plausible dummy activity or the management of trade-offs in the design of DGSs against adversaries' profiling practices of which we know little. These issues remain to be dealt with in future work.

Early on we have recognised obfuscation tools' potential to modulate users' consent (q.v. Chapter 1) [224]; however, this is a possibility we have not fully explored in this thesis. In our study of obfuscation tools we have implicitly assumed that users refuse *any* adversarial gain instead of considering that they may be willing to selectively and purposefully disclose certain bits of information, e.g. in exchange for service provision. Whereas UDO tools must necessarily trade privacy protection off for social utility —thereby implicitly modulating consent by design— we have not considered UPO designs that strategically give away information. In fact, we have implicitly considered that UPO tools' information leakage necessarily results from a shortage of resources —i.e. a lack of dummies— rather than strategic disclosure. Future work must examine how to strategically craft DGSs to selectively disclose information and thereby enable a more nuanced approach to user consent —i.e. giving users a more granular choice than simply refusing secondary processing to the extent that their dummy budget allows.

We have examined Protos' ability to *protect* user privacy in a trade-off with their potential for *expression* and *subversion*, in particular with respect to the adversarial assumptions that underlie each of those aims. However, we have not fully questioned the extent to which Protos enable users to contest and resist the advance and entrenchment of surveillance capitalism [581].

As privacy protection tools, Protos contribute to the narrative of market freedom and user choice, i.e. users can use Protos to evade profiling if they choose to do so, if they value their privacy enough. Coll observes that *"privacy [...] has been assimilated and reshaped by and in favour of capitalist structures, notably by being over-individualized[;] privacy seems to have become, somehow, a consumer good"* [136]. Moreover, to expect users to adopt Protos or any other tools to defend themselves against profiling and the negative impact of automated decision making fuels a culture of *responsibilisation* which is, as Barnard-Wills and Ashenden argue, *"a key feature of liberal governmentality"* [49, 50]. By expecting users with no previous technical training to identify online profiling as a problem and be able to select and successfully use the tools that protect them against it, we are *"putting the responsibility back on the private user and side-stepping the need to create a mature civil society around managing data"* [99].

Users require Protos because of the failure to implement systems according to principles such as *privacy by design* and *privacy by default*. And yet, as Munster argues, *"[we] cannot simply champion privacy and the individual against ubiquitous surveillance and the corporation"* [392].

We can better understand PETs' inadequacy to address the broader economic, political and social issues at stake when we consider one of the basic tenets underlying privacy engineering, namely, the provision of a given functionality under data minimisation constraints [262]. When it is the functionality itself that undermines users' and society's welfare, like in the case of automated decision making or experimentation that users cannot control [261], little it matters if the system is privacy preserving or not, i.e. whether it processes a minimum amount of data or whether users remain in control of their own data. Online behavioural advertising remains problematic for its ability to manipulate and discriminate users regardless of whether such functionality runs on the server or the client side [311, 331, 522]. Similarly, be it through differential privacy or multiparty computation (MPC), federated learning still enables *a trusted curator* to acquire a population's behavioural model and use it to its own advantage [236, 548], contributing to the very power and epistemic asymmetries that privacy invasive systems engender. In short, to the extent that privacy technologies enable problematic *secondary uses of data* without exposing or collecting user data, they remain part of the problem, not the solution. Hence, marginal, individual Protos' adoption is indeed unlikely to make a dent to profiling practices. As long as obfuscated profiles represent nothing but an outlier, Protos pose no threat to surveillance capitalism.

Brunton and Nissenbaum have likened obfuscation's role to Scott's *weapons of the weak*, i.e. *"acts of petty resistance"* that *"leave dominant symbolic structures intact"* [99, 478]. As individual privacy protection tools, Protos fit right in with this description. However, Scott further argues that when *"widely practiced by*

members of an entire class against elites or the state, they may have aggregate consequences all out of proportion to their banality when considered singly" and that *"many regime crises may be precipitated by the cumulative impact of everyday forms of resistance that reach critical thresholds"* [478]. From Scott's argument it follows that Protos, if adopted *en masse*, have the potential to disrupt the profiling process, as obfuscating users no longer represent outliers but a significant sample of the target population.

Yet through our examination of the conditions that enable Protos to disrupt profiling practices (q.v. Sect. 3.4.1), we have argued that privacy protection is at odds with the subversion component that enables Protos to inject noise in the profilers' databases, thereby disrupting profiling practices. It is through *undetectability* that Protos can prevent profilers from discarding obfuscated user data, echoing Scott's observation that *"those who employ everyday forms of resistance avoid calling attention to themselves"* [478]. In this thesis we have focused on the study of Protos as tools whose main underlying goal is privacy protection, e.g. the analytical framework we have proposed in Sect. 3.2 entirely comprises measures of information leakage, as opposed to other properties that capture or represent obfuscation tools' subversion potential. Future work must tend to the design and analysis of obfuscation technology for subversion purposes.

Outside the scope of this thesis, we have carried a first step in this direction through our formalisation of protective optimisation technologies (POTs) [261]. POTs represent a set of tools that rely on selective, strategic obfuscation to alter the outcomes of the optimisation processes that profiling informs. Protos contest profiling by attempting to render collected data meaningless, yet as we argue in Sect. 3.4.1, they do not have control over the optimisation or decision making processes that profiling informs. POTs address this gap by strategically altering user inputs in an attempt to elicit particular responses from the optimisation or decision making systems that feed on user profiles. POTs goals include, among others, *correcting* imbalances and improving outcomes for populations put at disadvantage or *sabotaging* and *boycotting* the system [261]. By breaking free from the privacy corset, POTs represent a promising framework for the study of obfuscation as a mechanism to contest the ills of profiling and optimisation systems as well as surveillance capitalism itself.

Leaving an eminently individualistic notion of privacy behind means that we envision POTs as a collective solution, rather than a tool that each individual can resort to of their own accord to protect themselves. POTs open the door for the development of responses coordinated among several users, the design of strategies for the benefit of the group —instead of individual users.

Moreover, throughout this thesis we have emphasised the importance of a sound security analysis that prevents us from overestimating the privacy protection

Protos afford to their users; we have warned against designs that rely on *security through obscurity* as a result of unrealistic assumptions about the adversaries these tools actually confront. However, we acknowledge that even faulty, flawed designs that do not provide as much privacy protection as expected still play a role in subverting the systems they attempt to defend users against. In his work on the use of deception techniques in computer security, Cohen makes an excellent case for the use of less-than-optimal chaff, arguing that *"[a]s long as the chaff costs less than the risks it mitigates, it is a good defense, and as long as simple deceptions reduce risk by more than the cost to deploy and operate them, they are good defenses as well"* [133]. Howe makes a similar point arguing that *"[e]ven in cases where the removal of [...] noise is possible, one must consider the resources required to do so"* [293].

Whether through PETs, Protos or POTs, we acknowledge that either of these solutions represents a technocentric approach to a complex set of political, economic and social problems that we cannot expect to address and negotiate through technology alone [389, 410]. As the regulatory framework that governs the collection and processing of users' data evolves, new opportunities and challenges arise, shaping in turn the design of technology that contests and protects from these practices.

The European Union's (EU) recent general data protection regulation (GDPR) represents a prominent example of how regulation may assist and inform the design of Protos and POTs [204]. GDPR's Article 22 prohibits profiling without the *data subject's explicit consent*, while recital 42 specifies that consent *"should not be regarded as freely given if the data subject has no genuine or free choice or is unable to refuse or withdraw consent without detriment"*. Hence, the relevance and need for Protos depends on the Court of Justice of the European Union's (CJEU) interpretation of the conditions that enable users to freely give consent, i.e. law replacing Protos as an instrument to modulate consent [224, 410].

Similarly, the provisions that the GDPR sets on the *accuracy* of the profiles that undergo further processing, e.g. through Article 5.1(d) and recital 71, may impact Protos' design [288]. If the CJEU were to interpret obfuscated data as *inaccurate*, profilers may see their ability to further processing obfuscated profiles limited, as profilers cannot ensure that the risks of errors for data subjects is minimised. This interpretation would positively affect the deployment of Protos, as they would effectively function as a mechanism to prevent further processing. If, on the contrary, the CJEU were to interpret obfuscated data as accurate, purposefully curated user data, e.g. according to the principle of *informational self-determination* [224], users could face negative consequences from further

processing of their obfuscated data.[1] At the same time however, by legitimising the strategic manipulation of users' inputs, this interpretation would favour the deployment of POTs.

Brkan further questions whether the GDPR's provisions on the transparency of decision making may lead to a *right to explanation*, as *"the GDPR obliges the controller to provide the data subject with 'meaningful information about the logic involved'"* [95]. To the extent that it provides designers with a greater understanding of the profiling and further decision making practices, transparency can further assist Protos and specially POTs, e.g. facilitating the reverse-engineering process required to effectively manipulate POTs' inputs or design DGSs that target a particular profiling function.

Hence, recent calls for increased algorithmic fairness, accountability and transparency inform Protos' and POTs' design in several ways. On the one hand, they may simultaneously legitimise the practices that these tools attempt to prevent and oppose. On the other hand, they may assist tool design by helping to rebalance the epistemic asymmetries designers face.

Come what may, Protos remain a patch, a temporary solution, a guerrilla tactics. They do not represent the real fix, a sustainable solution to online profiling. However, in the meantime, they provide a means for expression and resistance.

[1] This would effectively force adversaries to stay *naive* (q.v. 3.4.1), even if ensuring proper enforcement of the rule would pose additional, non-trivial challenges.

Bibliography

[1] R. Abu-Salma, E. M. Redmiles, B. Ur and M. Wei. Exploring user mental models of end-to-end encrypted communication tools. In *Proceedings of Workshop on free and open communications on the Internet (FOCI)*. USENIX, 2018. | Cit. on pp. 97, 205, 206, 214, 215.

[2] R. Abu-Salma, M. A. Sasse, J. Bonneau, A. Danilova, A. Naiakshina and M. Smith. Obstacles to the adoption of secure communication tools. In *Proceedings of Symposium on security and privacy (S&P)*, pp. 137–153. IEEE, 2017. | Cit. on pp. 41, 96.

[3] M. S. Ackerman and L. Cranor. Privacy critics: UI components to safeguard users' privacy. In *CHI'99 Extended Abstracts on Human Factors in Computing Systems*, pp. 258–259. ACM, 1999. | Cit. on p. 95.

[4] A. Acquisti. Privacy and security of personal information. economic incentives and technological solutions. In *Economics of information security*, pp. 179–186. Springer, 2004. | Cit. on pp. 201, 221.

[5] A. Acquisti. Privacy in electronic commerce and the economics of immediate gratification. In *Proceedings of Electronic Commerce (EC)*, pp. 21–29. ACM, 2004. | Cit. on p. 3.

[6] A. Acquisti. The economics of personal data and the economics of privacy. Technical report, 2010. Commissioned by the OECD for the OECD Roundtable on the Economics of Privacy and Personal Data, Paris. | Cit. on p. 221.

[7] A. Acquisti and R. Gross. Imagined communities: Awareness, information sharing, and privacy on the Facebook. In *Proceedings of Workshop on Privacy enhancing technologies (PET 2006)*, pp. 36–58. Springer, 2006. | Cit. on p. 3.

[8] A. Acquisti and J. Grossklags. Privacy and rationality in individual decision making. *IEEE Security & Privacy*, 3(1):26–33, 2005. | Cit. on p. 3.

[9] A. Adams and M. A. Sasse. Users are not the enemy. *Communications of the ACM*, 42(12):40–46, 1999. | Cit. on pp. 96, 205.

[10] A. Adams, M. A. Sasse and P. Lunt. Making passwords secure and usable. In *People and Computers XII*, pp. 1–19. Springer, 1997. | Cit. on p. 97.

[11] C. C. Aggarwal and P. S. Yu. A general survey of privacy-preserving data mining models and algorithms. In C. C. Aggarwal and P. S. Yu, editors, *Privacy-preserving data mining: models and algorithms*, pp. 11–52. Springer, 2008. | Cit. on p. 21.

[12] D. Agrawal and C. C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms. In *Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems (PODS'01)*, pp. 247–255. ACM, 2001. | Cit. on p. 73.

[13] D. Agrawal and D. Kesdogan. Measuring anonymity: The disclosure attack. *IEEE Security & Privacy*, 1(6):27–34, 2003. | Cit. on p. 133.

[14] R. Agrawal and R. Srikant. Privacy-preserving data mining. *ACM SIGMOD Record*, 29(2):439–450, 2000. | Cit. on p. 22.

[15] C. Aguilar-Melchor, J. Barrier, L. Fousse and M.-O. Killijian. XPIR: Private information retrieval for everyone. *Proceedings on privacy enhancing technologies (PoPETs)*, 2016(2):155–174, 2016. | Cit. on p. 115.

[16] W. U. Ahmad, K.-W. Chang and H. Wang. Intent-aware query obfuscation for privacy protection in personalized web search. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR'18)*, pp. 285–294. ACM, 2018. | Cit. on p. 159.

[17] R. Al-Rfou', W. Jannen and N. Patwardhan. TrackMeNot-so-good-after-all. Technical report, Stony Brook University, December 2010. | Cit. on p. 142.

[18] J. P. Albrecht. How the GDPR will change the world. *European Data Protection Law Review (EDPL)*, 2(3):287–289, 2016. | Cit. on p. 41.

[19] C. Alexander and I. Goldberg. Improved user authentication in Off-The-Record Messaging. In *Proceedings of Workshop on privacy in the electronic society (WPES'07)*, pp. 41–47. ACM, 2007. | Cit. on pp. 41, 42.

[20] J. Allar. Faircom c-treeACE database weak obfuscation algorithm vulnerability. Vulnerability note VU#900031, CERT Coordination Center, Software Engineering Institute, Carnegie Mellon University, 2013. | Cit. on p. 20.

[21] M. S. Alvim, M. E. Andrés, K. Chatzikokolakis, P. Degano and C. Palamidessi. Differential privacy: on the trade-off between utility and information leakage. In *Proceedings of Formal Aspects of Security and Trust (FAST)*, pp. 39–54. Springer, 2011. | Cit. on p. 77.

[22] M. S. Alvim, K. Chatzikokolakis, C. Palamidessi and G. Smith. Measuring information leakage using generalized gain functions. In *Proceedings of Computer security foundations symposium (CSF)*, pp. 265–279. IEEE, 2012. | Cit. on pp. 74, 75.

[23] J. A. Ambrose, R. G. Ragel and S. Parameswaran. RIJID: Random code injection to mask power analysis based side channel attacks. In *Proceedings of Design automation conference (DAC)*, pp. 489–492. ACM/IEEE, 2007. | Cit. on p. 49.

[24] B. Anckaert, M. Madou, B. De Sutter, B. De Bus, K. De Bosschere and B. Preneel. Program obfuscation: a quantitative approach. In *Proceedings of Workshop on quality of protection (QoP'07)*, pp. 15–20. ACM, 2007. | Cit. on pp. 17, 18.

[25] R. Anderson. Why information security is hard – An economic perspective. In *Proceedings of Annual computer security applications conference (ACSAC)*, pp. 358–365. IEEE, 2001. | Cit. on p. 97.

[26] M. Andrejevic. Surveillance in the digital enclosure. *The Communication Review*, 10(4):295–317, 2007. | Cit. on p. 170.

[27] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis and C. Palamidessi. Geo-indistinguishability: differential privacy for location-based systems. In *Proceedings of Computer & communications security (CCS'13)*, pp. 901–914. ACM, 2013. | Cit. on pp. 36, 67, 69, 83.

[28] A. Arampatzis, G. Drosatos and P. S. Efraimidis. Versatile query scrambling for private web search. *Information Retrieval Journal*, 18(4):331–358, 2015. | Cit. on p. 165.

[29] I. Arapakis, X. Bai and B. B. Cambazoglu. Impact of response latency on user behavior in web search. In *Proceedings of SIGIR conference on Research & development in information retrieval (SIGIR'14)*, pp. 103–112. ACM, 2014. | Cit. on p. 32.

[30] C. A. Ardagna, M. Cremonini, E. Damiani, S. De Capitani di Vimercati and P. Samarati. Location privacy protection through obfuscation-based techniques. In *IFIP Annual Conference on Data and Applications Security and Privacy (DBSec'07)*, pp. 47–60. Springer, 2007. | Cit. on pp. 15, 16, 22, 100.

[31] Article 29 Data Protection Working Party. Opinion 05/2014 on Anonymisation techniques, April 2014. | Cit. on p. 22.

[32] G. Ateniese, L. V. Mancini, A. Spognardi, A. Villani, D. Vitali and G. Felici. Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers. *International Journal of Security and Networks*, 10(3):137–150, 2015. | Cit. on p. 33.

[33] E. Atwater, C. Bocovich, U. Hengartner, E. Lank and I. Goldberg. Leading Johnny to water: Designing for usability and trust. In *Proceedings of Symposium on usable privacy and security (SOUPS)*, pp. 69–88. USENIX, 2015. | Cit. on pp. 98, 109, 215.

[34] M. Backes, A. Kate, P. Manoharan, S. Meiser and E. Mohammadi. Anoa: A framework for analyzing anonymous communication protocols. In *2013 IEEE 26th Computer Security Foundations Symposium*, pp. 163–178. IEEE, 2013. | Cit. on p. 85.

[35] R. Baden, A. Bender, N. Spring, B. Bhattacharjee and D. Starin. Persona: an online social network with user-defined privacy. In *SIGCOMM Conference on Data communication (SIGCOMM)*, pp. 135–146. ACM, 2009. | Cit. on p. 201.

[36] N. Badshah. Facebook to contact 87 million users affected by data breach. In *The Guardian*, at `https://www.theguardian.com/technology/2018/apr/08/facebook-to-contact-the-87-million-users-affected-by-data-breach`, April 2018. Last accessed on 2 August 2019. | Cit. on p. 201.

[37] E. Balsa. DummySN: Privacy-preserving social networks. Design and evaluation. Master's thesis, KU Leuven, 2010. | Cit. on pp. 182, 186, 230.

[38] E. Balsa, F. Beato and S. Gürses. Why can't online social networks encrypt? In *Workshop on privacy and user-centric controls*. W3C, 2014. | Cit. on p. 13.

[39] E. Balsa, L. Brandimarte, A. Acquisti, C. Diaz and S. Gürses. Spiny CACTOS: OSN users' attitudes and perceptions towards cryptographic access control tools. In *Workshop on Usable Security (USEC'14)*. Internet Society, 2014. | Cit. on pp. 12, 13, 206.

[40] E. Balsa, C. Pérez-Solà and C. Diaz. Towards inferring communication patterns in online social networks. *ACM Transactions on Internet Technology (TOIT)*, 17(3), 2017. | Cit. on pp. 13, 192.

[41] E. Balsa, C. Troncoso and C. Diaz. A metric to evaluate interaction obfuscation in online social networks. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 20(06):877–892, 2012. | Cit. on pp. 12, 13, 22.

[42] E. Balsa, C. Troncoso and C. Diaz. OB-PWS: Obfuscation-based private web search. In *IEEE Symposium on Security and Privacy (S&P'12)*, pp. 491–505, 2012. | Cit. on pp. 6, 12, 15, 22, 142.

[43] S. Banescu and A. Pretschner. A tutorial on software obfuscation. In *Advances in Computers*, volume 108, pp. 283–353. Elsevier, 1st edition, 2018. | Cit. on pp. 17, 18.

[44] B. Barak. Hopes, fears, and software obfuscation. *Communications of the ACM*, 59(3):88–96, 2016. | Cit. on p. 19.

[45] B. Barak. The complexity of public-key cryptography. In *Tutorials on the Foundations of Cryptography*, Information Security and Cryptography, pp. 45–77. Springer, 2017. | Cit. on pp. 19, 20.

[46] B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. Vadhan and K. Yang. On the (im)possibility of obfuscating programs. In *Proceedings of Advances in cryptology (CRYPTO)*, pp. 1–18. Springer, 2001. | Cit. on p. 19.

[47] P. Baran. On distributed communications: IX. security, secrecy, and tamper-free considerations. Technical report, The Rand Corporation, 1964. | Cit. on p. 47.

[48] M. Barbaro, T. Zeller and S. Hansell. A face is exposed for AOL searcher no. 4417749. In *The New York Times* at `https://www.nytimes.com/2006/08/09/technology/09aol.html`, Aug 2006. Last accessed on 2 August 2019. | Cit. on p. 114.

[49] D. Barnard-Wills. E-safety education: Young people, surveillance and responsibility. *Criminology & Criminal Justice*, 12(3):239–255, 2012. | Cit. on p. 244.

[50] D. Barnard-Wills and D. Ashenden. Public sector engagement with online identity management. *Identity in the Information Society*, 3(3):657–674, 2010. | Cit. on p. 244.

[51] G. Barthe and B. Kopf. Information-theoretic bounds for differentially private mechanisms. In *Proceedings of Computer security foundations symposium (CSF)*, pp. 191–204. IEEE, 2011. | Cit. on p. 77.

[52] R. Bassily and A. Smith. Local, private, efficient protocols for succinct histograms. In *Proceedings of Symposium on theory of computing (STOC)*, pp. 127–135. ACM, 2015. | Cit. on pp. 68, 69.

[53] L. Batina, B. Gierlichs, E. Prouff, M. Rivain, F.-X. Standaert and N. Veyrat-Charvillon. Mutual information analysis: a comprehensive study. *Journal of Cryptology*, 24(2):269–291, 2011. | Cit. on p. 192.

[54] C. Baum, I. Damgård and C. Orlandi. Publicly auditable secure multi-party computation. In *Security and Cryptography for Networks (SCN)*, pp. 175–196. Springer, 2014. | Cit. on p. 40.

[55] E. Bayamlioğlu, I. Baraliuc, L. A. W. Janssens and M. Hildebrandt, editors. *Being profiled: Cogitas ergo sum. 10 years of profiling the European citizen.* Amsterdam University Press, 2018. | Cit. on p. 104.

[56] Facebook to integrate WhatsApp, Instagram and Messenger. In *BBC News*, at `https://www.bbc.com/news/technology-47001460`, January 2019. Last accessed on 5 August 2019. | Cit. on p. 174.

[57] F. Beato, M. Kohlweiss and K. Wouters. Scramble! your social network data. In *Privacy Enhancing Technologies Symposium (PETS)*, pp. 211–225. Springer, 2011. | Cit. on pp. 201, 204, 207.

[58] F. Beato, S. Meul and B. Preneel. Practical identity-based private sharing for online social networks. *Computer Communications*, 73(B):243–250, 2016. | Cit. on pp. 203, 204.

[59] S. M. Beitzel, E. C. Jensen, O. Frieder, D. D. Lewis, A. Chowdhury and A. Kolcz. Improving automatic query classification via semi-supervised learning. In *International Conference on Data Mining (ICDM)*, pp. 42–49. IEEE, 2005. | Cit. on p. 117.

[60] M. Bellare and A. Boldyreva. The security of chaffing and winnowing. In *Proceedings of Advances in cryptology (ASIACRYPT)*, pp. 517–530. Springer, 2000. | Cit. on p. 49.

[61] M. Bellare, S. Tessaro and A. Vardy. Semantic security for the wiretap channel. In *Proceedings of Advances in cryptology (CRYPTO)*, pp. 294–311. Springer, 2012. | Cit. on p. 73.

[62] S. M. Bellovin. Cryptography and the Internet. In *Proceedings of Advances in cryptology (CRYPTO)*, pp. 46–55. Springer, 1998. | Cit. on p. 42.

[63] S. M. Bellovin, M. Blaze, S. Landau and S. K. Pell. It's too complicated: How the Internet uppends *Katz*, *Smith*, and electronic surveillance law. *Harvard Jounal Law and Technology*, 30(1):1–101, 2016. | Cit. on p. 51.

[64] S. Ben Mokhtar, A. Boutet, P. Felber, M. Pasin, R. Pires and V. Schiavoni. X-Search: Revisiting private web search using Intel SGX. In *ACM/IFIP/USENIX Middleware Conference (Middleware)*, pp. 198–208. ACM, 2017. | Cit. on pp. 115, 165.

[65] R. Benbunan-Fich. The ethics of online research with unsuspecting users: From A/B testing to C/D experimentation. *Research Ethics*, 13(3-4):200–218, 2017. | Cit. on p. 2.

[66] A. Bendiek and M. Römer. Externalizing Europe: the global effects of European data protection. *Digital Policy, Regulation and Governance*, 21(1):32–43, 2019. | Cit. on p. 4.

[67] F. Benevenuto, T. Rodrigues, M. Cha and V. Almeida. Characterizing user behavior in online social networks. In *Internet measurement conference (IMC)*, pp. 49–62. ACM SIGCOMM, 2009. | Cit. on p. 182.

[68] F. Benevenuto, T. Rodrigues, M. Cha and V. Almeida. Characterizing user navigation and interactions in online social networks. *Information Sciences*, 195:1–24, 2012. | Cit. on p. 182.

[69] A. Bennaceur, V. Issarny, R. Spalazzese and S. Tyagi. Achieving interoperability through semantics-based technologies: The instant messaging case. In *International Semantic Web Conference (ISWC)*, pp. 17–33. Springer, 2012. | Cit. on p. 206.

[70] S. Berkovsky, Y. Eytani, T. Kuflik and F. Ricci. Enhancing privacy and preserving accuracy of a distributed collaborative filtering. In *Proceedings of Recommender systems (RecSys)*, pp. 9–16. ACM, 2007. | Cit. on p. 16.

[71] D. Bernhard, V. Cortier, O. Pereira, B. Smyth and B. Warinschi. Adapting Helios for provable ballot privacy. In *Proceedings of European symposium on research in computer security (ESORICS)*, pp. 335–354. Springer, 2011. | Cit. on p. 42.

[72] D. J. Bernstein. Introduction to post-quantum cryptography. In *Post-quantum cryptography*, pp. 1–14. Springer, 2009. | Cit. on p. 20.

[73] A. Berson. *Client/Server architecture*. McGraw-Hill, 1992. | Cit. on p. 60.

[74] O. Berthold, H. Federrath and S. Köpsell. Web MIXes: A system for anonymous and unobservable Internet access. In *Workshop on design issues in anonymity and unobservability*, pp. 115–129. Springer, 2000. | Cit. on pp. 114, 217.

[75] O. Berthold and H. Langos. Dummy traffic against long term intersection attacks. In *Proceedings of Workshop on privacy enhancing technologies (PET)*, pp. 110–128. Springer, 2002. | Cit. on p. 48.

[76] D. Besnard and B. Arief. Computer security impaired by legitimate users. *Computers & Security*, 23(3):253–264, 2004. | Cit. on p. 96.

[77] K. Best. Living in the control society: Surveillance, users and digital screen technologies. *International Journal of Cultural Studies*, 13(1):5–24, 2010. | Cit. on pp. 3, 4.

[78] K. Bicakci, I. E. Bagci and B. Tavli. Lifetime bounds of wireless sensor networks preserving perfect sink unobservability. *IEEE Communications Letters*, 15(2):205–207, 2011. | Cit. on p. 48.

[79] A. J. Biega, R. Saha Roy and G. Weikum. Privacy through solidarity: A user-utility-preserving framework to counter profiling. In *Conference on research and development in information retrieval (SIGIR)*, pp. 675–684. ACM, 2017. | Cit. on p. 114.

[80] A. Bielenberg, L. Helm, A. Gentilucci, D. Stefanescu and H. Zhang. The growth of Diaspora – A decentralized online social network in the wild. In *IEEE Conference on Computer Communications Workshops (INFOCOM)*, pp. 13–18. IEEE, 2012. | Cit. on p. 172.

[81] C. Bier and E. Krempel. Common privacy patterns in video surveillance and smart energy. In *International conference on computing and convergence technology (ICCCT)*, pp. 610–615. IEEE, 2012. | Cit. on p. 222.

[82] V. Bindschaedler and R. Shokri. Synthesizing plausible privacy-preserving location traces. In *Proceedings of Symposium on security and privacy (S&P)*, pp. 546–563. IEEE, 2016. | Cit. on pp. 77, 89, 94.

[83] P. Bloom. *Monitored: Business and surveillance in a time of big data*. Pluto Press, 2019. | Cit. on p. 3.

[84] A. Bogdanov, editor. *Proceedings of Lightweight cryptography for security and privacy (LightSec)*. Springer, 2016. | Cit. on p. 42.

[85] P. Bogetoft, D. L. Christensen, I. Damgård, M. Geisler, T. Jakobsen, M. Krøigaard, J. D. Nielsen, J. B. Nielsen, K. Nielsen, J. Pagter, M. Schwartzbach and T. Toft. Secure multiparty computation goes live. In *Proceedings of Financial cryptography and data security (FC)*, pp. 325–343. Springer, 2009. | Cit. on pp. 39, 40.

[86] P. Bogetoft, I. Damgård, T. Jakobsen, K. Nielsen, J. Pagter and T. Toft. A practical implementation of secure auctions based on multiparty integer computation. In *Financial Cryptography and Data Security (FC)*, pp. 142–147. Springer, 2006. | Cit. on p. 40.

[87] D. Boneh and M. K. Franklin. Identity-based encryption from the Weil pairing. In *Annual international cryptology conference on advances in cryptology (CRYPTO)*, pp. 213–229. Springer, 2001. | Cit. on p. 203.

[88] J. Bonneau, J. Anderson, R. Anderson and F. Stajano. Eight friends are enough: social graph approximation via public listings. In *Proceedings of the Second ACM EuroSys Workshop on Social Network Systems*, pp. 13–18. ACM, 2009. | Cit. on p. 171.

[89] N. Borisov, I. Goldberg and E. Brewer. Off-the-record communication, or, why not to use PGP. In *Workshop on privacy in the electronic society (WPES)*, pp. 77–84. ACM, 2004. | Cit. on p. 171.

[90] A. Boulanger. Open-source versus proprietary software: Is one more reliable and secure than the other? *IBM Systems Journal*, 44(2):239–248, 2005. | Cit. on p. 109.

[91] B. M. Bowen, S. Hershkop, A. D. Keromytis and S. J. Stolfo. Baiting inside attackers using decoy documents. In *Proceedings of Security and privacy in communication networks (SecureComm)*, pp. 51–70. Springer, 2009. | Cit. on p. 56.

[92] d. m. boyd and N. B. Ellison. Social network sites: Definition, history, and scholarship. *Journal of computer-mediated communication*, 13(1):210–230, 2007. | Cit. on p. 201.

[93] L. Brandimarte, A. Acquisti and G. Loewenstein. Misplaced confidences: Privacy and the control paradox. *Social psychological and personality science*, 4(3):340–347, 2013. | Cit. on p. 210.

[94] M. Brennan, S. Afroz and R. Greenstadt. Adversarial stylometry: Circumventing authorship recognition to preserve privacy and anonymity. *ACM Transactions on Information and System Security (TISSEC)*, 15(3):12, 2012. | Cit. on p. 43.

[95] M. Brkan. Do algorithms rule the world? Algorithmic decision-making in the framework of the GDPR and beyond. *International journal of law and information technology*, 27(2):91–121, 2019. | Cit. on pp. 4, 247.

[96] J. Brooke. SUS: a 'quick and dirty' usability scale. In *Usability evaluation in industry*, pp. 189–194. Taylor & Francis, 1996. | Cit. on pp. 208, 309.

[97] C. Brubaker, A. Houmansadr and V. Shmatikov. CloudTransport: Using cloud storage for censorship-resistant networking. In *Proceedings of Privacy enhancing technologies symposium (PETS)*, pp. 1–20. Springer, 2014. | Cit. on p. 54.

[98] F. Brunton and H. Nissenbaum. Vernacular resistance to data collection and analysis: A political theory of obfuscation. *First Monday*, 16(5), 2011. | Cit. on pp. 101, 107, 166.

[99] F. Brunton and H. Nissenbaum. Political and ethical perspectives on data obfuscation. In *Privacy, due process and the computational turn. The philosophy of law meets the philosophy of technology*, pp. 171–195. Taylor and Francis, 2013. | Cit. on pp. 4, 6, 7, 8, 9, 244.

[100] F. Brunton and H. Nissenbaum. *Obfuscation: A user's guide for privacy and protest.* MIT Press, 2015. | Cit. on pp. 5, 7, 15, 16, 37, 38, 104, 107, 166, 229.

[101] F. Brunton, H. Nissenbaum and others. International workshop on obfuscation. Science, technology and theory. Workshop report, 2017. | Cit. on p. 6.

[102] D. A. Burgess and S. Kapur. Controlled targeted experimentation. Google Patents, March 2009. US Patent number: US 2009/0063250 A1. | Cit. on p. 2.

[103] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad and M. Stal. *Pattern-Oriented Software Architecture. Volume 1: A System of Patterns.* Wiley Publishing, 1996. | Cit. on p. 223.

[104] C. Cachin. *Entropy measures and unconditional security in cryptography.* PhD thesis, ETH Zurich, 1997. | Cit. on pp. 72, 74, 75.

[105] X. Cai, R. Nithyanand and R. Johnson. CS-BuFLO: A congestion sensitive website fingerprinting defense. In *Proceedings of Workshop on privacy in the electronic society (WPES)*, pp. 121–130. ACM, 2014. | Cit. on pp. 48, 55.

[106] L. J. Camp. Mental models of privacy and security. *IEEE Technology and society magazine*, 28(3):37–46, 2009. | Cit. on p. 96.

[107] R. Canetti, Y. Dodis, S. Halevi, E. Kushilevitz and A. Sahai. Exposure-resilient functions and all-or-nothing transforms. In *Proceedings of Advances in cryptology (EUROCRYPT)*, pp. 453–469. Springer, 2000. | Cit. on p. 66.

[108] N. Carlini and D. Wagner. Towards evaluating the robustness of neural networks. In *Proceedings of Symposium on security and privacy (S&P)*, pp. 39–57. IEEE, 2017. | Cit. on p. 104.

[109] J. Castellà-Roca, A. Viejo and J. Herrera-Joancomartí. Preserving user's privacy in web search engines. *Computer communications*, 32(13-14):1541–1551, 2009. | Cit. on pp. 115, 165.

[110] C. Castelluccia. Behavioural tracking on the Internet: A technical perspective. In *European data protection: In good health?*, pp. 21–33. Springer, 2012. | Cit. on p. 170.

[111] S. Castillo-Perez and J. Garcia-Alfaro. Anonymous resolution of DNS queries. In *Proceedings of On the move to meaningful Internet systems (OTM)*, pp. 987–1000. Springer, 2008. | Cit. on p. 52.

[112] S. Castillo-Perez and J. Garcia-Alfaro. Evaluation of two privacy-preserving protocols for the DNS. In *Proceedings of International conference on information technology: New generations (ITNG)*, pp. 411–416. IEEE, 2009. | Cit. on p. 52.

[113] E.-C. Chang, R. Shen and F. W. Teo. Finding the original point set hidden among chaff. In *Proceedings of Asia conference on computer and communications security (ASIACCS)*, pp. 182–188. ACM, 2006. | Cit. on p. 50.

[114] K. Chatzikokolakis, C. Palamidessi and P. Panangaden. Anonymity protocols as noisy channels. *Information and computation*, 206(2-4):378–401, 2008. | Cit. on pp. 73, 74.

[115] K. Chatzikokolakis and C. Troncoso. Editors' introduction. *Proceedings on privacy enhancing technologies (PoPETs)*, 2019(1):1–4, 2019. | Cit. on p. 3.

[116] D. L. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–90, 1981. | Cit. on p. 47.

[117] C. Chen, D. E. Asoni, A. Perrig, D. Barrera, G. Danezis and C. Troncoso. TARANET: Traffic-analysis resistant anonymity at the network layer. In *Proceedings of European symposium on security and privacy (EuroS&P)*, pp. 137–152. IEEE, 2018. | Cit. on p. 48.

[118] L. Chen, S. P. Jordan, Y.-K. Liu, D. Moody, R. C. Peralta, R. A. Perlner and D. C. Smith-Tone. Report on post-quantum cryptography. NIST Interagency/Internal Report (NISTIR) 8105, NIST, 2016. | Cit. on p. 20.

[119] L. Chen, K. Harrison, D. Soldera and N. P. Smart. Applications of multiple trust authorities in pairing based cryptosystems. In *Proceedings of Infrastructure security (InfraSec)*, volume 2437 of *LNCS*, pp. 260–275. Springer, 2002. | Cit. on p. 203.

[120] W. R. Cheswick. Johnny can obfuscate: beyond mother's maiden name. In *Workshop on hot topics in security (HotSec)*. USENIX, 2006. | Cit. on p. 177.

[121] S. Chinoy. What 7 creepy patents reveal about facebook. In *The New York Times* at `https://www.nytimes.com/interactive/2018/06/21/opinion/sunday/facebook-patents-privacy.html`, June 2018. Last accessed on 13 September 2019. | Cit. on p. 170.

[122] G. Chittaranjan, J. Blom and D. Gatica-Perez. Mining large-scale smartphone data for personality studies. *Personal and ubiquitous computing*, 17(3):433–450, 2013. | Cit. on p. 171.

[123] B. Chor, E. Kushilevitz, O. Goldreich and M. Sudan. Private information retrieval. *Journal of the ACM (JACM)*, 45:965–981, 1998. | Cit. on p. 115.

[124] E. S. Chung, J. I. Hong, J. Lin, M. K. Prabaker, J. A. Landay and A. L. Liu. Development and evaluation of emerging design patterns for ubiquitous computing. In *Proceedings of Designing interactive systems (DIS)*, pp. 233–242. ACM, 2004. | Cit. on pp. 222, 223.

[125] J. Cinnamon. Social injustice in surveillance capitalism. *Surveillance & society*, 15(5):609–625, 2017. | Cit. on p. 3.

[126] D. K. Citron and F. Pasquale. The scored society: Due process for automated predictions. *Washington Law Review*, 89(1):1–33, 2014. | Cit. on p. 2.

[127] T. Claburn. The cybercriminal's cash cow and the marketer's machine: Inside the mad sad bad web ad world. In *The Register* at `https://www.theregister.co.uk/2018/06/29/ad_fraud_bad/`, Jun 2018. Last accessed on 8 August 2019. | Cit. on p. 105.

[128] J. Clark, P. C. Van Oorschot and C. Adams. Usability of anonymous web browsing: an examination of Tor interfaces and deployability. In

Proceedings of Symposium on usable privacy and security (SOUPS), pp. 41–51. ACM, 2007. | Cit. on p. 99.

[129] J. Clark and J. Jacob. A survey of authentication protocol literature: Version 1.0. Report, University of York, 1997. | Cit. on p. 39.

[130] M. R. Clarkson, A. C. Myers and F. B. Schneider. Belief in information flow. In *Proceedings of Computer security foundations workshop (CSFW)*, pp. 31–45. IEEE, 2005. | Cit. on pp. 78, 79, 133.

[131] M. R. Clarkson, A. C. Myers and F. B. Schneider. Quantifying information flow with beliefs. *Journal of computer security*, 17(5):655–701, 2009. | Cit. on pp. 79, 80.

[132] R. Clayton and G. Danezis. Chaffinch: Confidentiality in the face of legal threats. In *Proceedings of Workshop on information hiding (IH)*, pp. 70–86. Springer-Verlag, 2002. | Cit. on p. 49.

[133] F. Cohen. The use of deception techniques: honeypots and decoys. In *Handbook of information security: Threats, vulnerabilities, prevention, detection, and management*, volume 3, pp. 646–655. John Wiley & Sons, 2006. | Cit. on pp. 56, 246.

[134] D. Cole. We kill people based on metadata. In The New York Review of Books at `https://www.nybooks.com/daily/2014/05/10/we-kill-people-based-metadata/`, May 2014. Last accessed on 8 August 2019. | Cit. on p. 171.

[135] M. Colesky, J.-H. Hoepman and C. Hillen. A critical analysis of privacy design strategies. In *Security and privacy workshops (SPW)*, pp. 33–40. IEEE, 2016. | Cit. on p. 235.

[136] S. Coll. Power, knowledge, and the subjects of privacy: understanding privacy as the ally of surveillance. *Information, communication & society*, 17(10):1250–1263, 2014. | Cit. on p. 244.

[137] C. S. Collberg and C. Thomborson. Watermarking, tamper-proofing, and obfuscation — Tools for software protection. *Transactions on software engineering*, 28(8):735–746, 2002. | Cit. on pp. 15, 17, 18, 19.

[138] C. S. Collberg, C. Thomborson and D. Low. A taxonomy of obfuscating transformations. Computer science technical report #148, University of Auckland, 1997. | Cit. on p. 15.

[139] J.-S. Coron, M. Joye, D. Naccache and P. Paillier. New attacks on PKCS#1 v1.5 encryption. In *Proceedings of Advances in cryptology (EUROCRYPT)*, pp. 369–381. Springer, 2000. | Cit. on p. 66.

[140] T. M. Cover and J. A. Thomas. *Elements of information theory*. Wiley-Interscience, 1991. | Cit. on p. 151.

[141] R. Cramer, I. B. Damgård and J. B. Nielsen. *Secure multiparty computation and secret sharing*. Cambridge University Press, 2015. | Cit. on p. 39.

[142] R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *Proceedings of Advances in cryptology (CRYPTO)*, pp. 13–25. Springer, 1998. | Cit. on p. 66.

[143] L. F. Cranor and N. Sadeh. A shortage of privacy engineers. *IEEE Security & privacy*, 11(2):77–79, 2013. | Cit. on pp. 220, 221.

[144] K. Crawford and J. Schultz. Big data and due process: toward a framework to redress predictive privacy harms. *Boston College Law Review*, 55(1):93–128, 2014. | Cit. on pp. 2, 3.

[145] P. Cuff and L. Yu. Differential privacy as a mutual information constraint. In *Proceedings of Computer and communications security (CCS)*, pp. 43–54. ACM, 2016. | Cit. on pp. 73, 77.

[146] E. Cutrell, D. Robbins, S. Dumais and R. Sarin. Fast, flexible filtering with *Phlat* — Personal search and organization made easy. In *Proceedings of the SIGCHI conference on human factors in computing systems (CHI)*, pp. 261–270. ACM, 2006. | Cit. on p. 166.

[147] J. Daemen and V. Rijmen. *The design of Rijndael: AES – The advanced encryption standard*. Springer, 2013. | Cit. on p. 176.

[148] I. Damgård, M. Keller, E. Larraia, V. Pastro, P. Scholl and N. P. Smart. Practical covertly secure MPC for dishonest majority – or: Breaking the SPDZ limits. In *Proceedings of European symposium on research in computer security (ESORICS)*, pp. 1–18. Springer, 2013. | Cit. on pp. 33, 39.

[149] I. B. Damgård, T. P. Pedersen and B. Pfitzmann. Statistical secrecy and multibit commitments. *IEEE Transactions on information theory*, 44(3):1143–1151, 1998. | Cit. on pp. 71, 73.

[150] E. Damiani, S. De Capitani Di Vimercati, S. Paraboschi and P. Samarati. Computing range queries on obfuscated data. In *Proceedings of Information processing and management of uncertainty in knowledge-based systems (IPMU)*. Springer, 2004. | Cit. on p. 20.

[151] G. Danezis. Statistical disclosure attacks. In *Proceedings of Security and privacy in the age of uncertainty (SEC)*, pp. 421–426. Springer, 2003. | Cit. on p. 133.

[152] G. Danezis and C. Diaz. A survey of anonymous communication channels. Technical Report MSR-TR-2008-35, Microsoft Research, 2008. | Cit. on p. 47.

[153] G. Danezis, C. Diaz, C. Troncoso and B. Laurie. Drac: An architecture for anonymous low-volume communications. In *Proceedings of Privacy enhancing technologies symposium (PETS)*, pp. 202–219. Springer, 2010. | Cit. on pp. 89, 95, 217.

[154] G. Danezis, R. Dingledine and N. Mathewson. Mixminion: Design of a type III anonymous remailer protocol. In *Proceedings of Symposium on security and privacy (S&P)*, pp. 2–15. IEEE, 2003. | Cit. on p. 43.

[155] G. Danezis, J. Domingo-Ferrer, M. Hansen, J.-H. Hoepman, D. Le Métayer, R. Tirtea and S. Schiffner. Privacy and data protection by design – from policy to engineering. Technical report, ENISA, 2014. | Cit. on p. 221.

[156] G. Danezis and S. Gürses. A critical review of 10 years of privacy technology. In *Proceedings of Surveillance cultures: A global surveillance society?*, pp. 1–16, 2010. | Cit. on p. 47.

[157] G. Danezis and P. Mittal. SybilInfer: Detecting Sybil nodes using social networks. In *The network and distributed system security symposium (NDSS)*, pp. 1–15. Internet Society, 2009. | Cit. on p. 217.

[158] G. Danezis and C. Troncoso. Vida: How to use Bayesian inference to de-anonymize persistent communications. In *Privacy enhancing technologies symposium (PETS)*, pp. 56–72. Springer, 2009. | Cit. on p. 218.

[159] D. Das, S. Meiser, E. Mohammadi and A. Kate. Anonymity trilemma: Strong anonymity, low bandwidth overhead, low latency—Choose two. In *Symposium on security and privacy (S&P)*. IEEE, 2018. | Cit. on p. 48.

[160] A. Datta, M. C. Tschantz and A. Datta. Automated experiments on Ad Privacy Settings. A tale of opacity, choice, and discrimination. *Proceedings on privacy enhancing technologies (PoPETs)*, 2015(1):92–112, 2015. | Cit. on p. 161.

[161] A. Datta, S. Buchegger, L.-H. Vu, T. Strufe and K. Rzadca. Decentralized online social networks. In *Handbook of social network technologies and applications*, pp. 349–378. Springer, 2010. | Cit. on p. 4.

[162] A. De Luca, S. Das, M. Ortlieb, I. Ion and B. Laurie. Expert and non-expert attitudes towards (secure) instant messaging. In *Symposium on usable privacy and security (SOUPS)*. USENIX, 2016. | Cit. on pp. 97, 98.

[163] Y.-A. de Montjoye, J. Quoidbach, F. Robic and A. S. Pentland. Predicting personality using novel mobile phone-based metrics. In *Proceedings of Social computing, behavioral-cultural modeling, and prediction (SBP)*, pp. 48–55. Springer, 2013. | Cit. on p. 171.

[164] Y.-A. De Montjoye, L. Radaelli, V. K. Singh and A. S. Pentland. Unique in the shopping mall: On the reidentifiability of credit card metadata. *Science*, 347(6221):536–539, 2015. | Cit. on p. 43.

[165] J. Dean, G. Harik and P. Buchheit. Serving advertisements using information associated with e-mail. Google Patents, March 2004. US Patent number: US 2004/0059712 A1. | Cit. on p. 170.

[166] J. Dean. Affective networks. *MediaTropes*, 2(2):19–44, 2010. | Cit. on p. 170.

[167] M. Degeling and T. Herrmann. Your interests according to Google – A profile-centered analysis for obfuscation of online tracking profiles. *Computing research repository (CoRR)*, arXiv:1601.06371, 2016. | Cit. on p. 158.

[168] L. Dencik and J. Cable. The advent of surveillance realism: Public opinion and activist responses to the Snowden leaks. *International journal of communication*, 11:763–781, 2017. | Cit. on p. 4.

[169] L. Dencik, A. Hintz and J. Cable. Towards data justice? The ambiguity of anti-surveillance resistance in political activism. *Big data & society*, 3(2):1–12, 2016. | Cit. on p. 221.

[170] C. Devet and I. Goldberg. The best of both worlds: Combining information-theoretic and computational PIR for communication efficiency. In *Privacy enhancing technologies symposium (PETS)*, pp. 63–82. Springer, 2014. | Cit. on pp. 4, 115.

[171] C. Diaz and B. Preneel. Reasoning about the anonymity provided by pool mixes that generate dummy traffic. In *Proceedings of Information hiding (IH)*, pp. 309–325. Springer, 2004. | Cit. on p. 48.

[172] C. Diaz, O. Tene and S. Gürses. Hero or villain: The data controller in privacy law and technologies. *Ohio state law journal*, 74(6):923–964, 2013. | Cit. on p. 233.

[173] C. Diaz, C. Troncoso and G. Danezis. Does additional information always reduce anonymity? In *Workshop on privacy in the electronic society (WPES)*, pp. 72–75. ACM, 2007. | Cit. on p. 133.

[174] C. Diaz, C. Troncoso and A. Serjantov. On the impact of social network profiling on anonymity. In *Privacy enhancing technologies symposium (PETS)*, pp. 44–62. Springer, 2008. | Cit. on p. 186.

[175] R. Dingledine and N. Mathewson. Anonymity loves company: Usability and the network effect. In *Workshop on the economics of information security (WEIS)*, 2006. | Cit. on pp. 96, 100.

[176] R. Dingledine, N. Mathewson and P. Syverson. Tor: The second-generation onion router. In *Proceedings of USENIX Security symposium*, pp. 1–17. USENIX, 2004. | Cit. on pp. 4, 28, 43, 114, 224, 240.

[177] R. Dingledine (*arma*). Did the FBI pay a university to attack Tor users? In *The Tor Project Blog*. Online at `https://blog.torproject.org/did-fbi-pay-university-attack-tor-users`, November 2015. Last accessed on 15 August 2019. | Cit. on p. 44.

[178] P. Dixon. Consumer tips: Search engine privacy. In *World Privacy Forum* at `https://www.worldprivacyforum.org/2013/10/consumer-tips-search-engine-privacy-2/`, October 2013. Last accessed on 15 August 2019. | Cit. on p. 114.

[179] S. Dodier-Lazaro, R. Abu-Salma, I. Becker and M. A. Sasse. From paternalistic to user-centred security: Putting users first with value-sensitive design. In *CHI Workshop on Values in Computing (ViC)*, pp. 1–7. ACM, 2017. | Cit. on pp. 96, 98, 215.

[180] Y. Dodis, R. Ostrovsky, L. Reyzin and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM Journal on computing*, 38(1):97–139, 2008. | Cit. on p. 75.

[181] J. Domingo-Ferrer, M. Bras-Amorós, Q. Wu and J. Manjón. User-private information retrieval based on a peer-to-peer community. *Data & knowledge engineering*, 68(11):1237–1252, 2009. | Cit. on p. 165.

[182] J. Domingo-Ferrer, A. Solanas and J. Castellà-Roca. *h(k)*-private information retrieval from privacy-uncooperative queryable databases. *Online information review*, 33(4):720–744, 2009. | Cit. on pp. 52, 116, 123, 132, 133, 229.

[183] N. Doty and M. Gupta. Privacy design patterns and anti-patterns. Patterns misapplied and unintended consequences. In *Symposium on*

usable privacy and security (SOUPS). A turn for the worse: Trustbusters for user interfaces workshop, 2013. | Cit. on pp. 222, 233.

[184] A. Drutsa, G. Gusev and P. Serdyukov. Future user engagement prediction and its application to improve the sensitivity of online experiments. In *Proceedings of World wide web conference (WWW)*, pp. 256–266. International world wide web conferences steering committee, 2015. | Cit. on p. 2.

[185] J. C. Duchi, M. I. Jordan and M. J. Wainwright. Local privacy and statistical minimax rates. In *Foundations of computer science symposium (FOCS)*, pp. 429–438. IEEE, 2013. | Cit. on p. 67.

[186] M. Duckham and L. Kulik. A formal model of obfuscation and negotiation for location privacy. In *Proceedings of International conference on pervasive computing*, pp. 152–170. Springer, 2005. | Cit. on pp. 15, 16, 21, 22, 36.

[187] C. Dwork. Differential privacy. In *International colloquium on automata, languages, and programming (ICALP)*, pp. 1–12. Springer, 2006. | Cit. on pp. 33, 34.

[188] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov and M. Naor. Our data, ourselves: Privacy via distributed noise generation. In *Proceedings of Advances in cryptology (EUROCRYPT)*, pp. 486–503. Springer, 2006. | Cit. on p. 68.

[189] C. Dwork, F. McSherry, K. Nissim and A. Smith. Calibrating noise to sensitivity in private data analysis. *Journal of privacy and confidentiality*, 7(3):17–51, 2017. | Cit. on pp. 34, 67, 68, 77.

[190] C. Dwork and M. Naor. On the difficulties of disclosure prevention in statistical databases or the case for differential privacy. *Journal of privacy and confidentiality*, 2(1):93–107, 2010. | Cit. on pp. 39, 83.

[191] C. Dwork and A. Roth. The algorithmic foundations of differential privacy. *Foundations and trends in theoretical computer science*, 9(3–4):211–407, 2014. | Cit. on pp. 34, 77.

[192] K. P. Dyer, S. E. Coull, T. Ristenpart and T. Shrimpton. Peek-a-boo, I still see you: Why efficient traffic analysis countermeasures fail. In *Symposium on security and privacy (S&P)*, pp. 332–346. IEEE, 2012. | Cit. on pp. 48, 55, 232.

[193] Do Not Track. Electronic Frontier Foundation. Online at `https://www.eff.org/issues/do-not-track`. Last accessed on 15 August 2019. | Cit. on pp. 224, 225, 229.

[194] M. Egorov and M. Wilkison. ZeroDB white paper. *Computing research repository (CoRR)*, arXiv:1602.07168, 2016. | Cit. on p. 42.

[195] Y. Elovici, C. Glezer and B. Shapira. Enhancing customer privacy while searching for products and services on the world wide web. *Internet research*, 15(4):378–399, 2005. | Cit. on pp. 145, 149.

[196] Y. Elovici, B. Shapira and A. Maschiach. A new privacy model for hiding group interests while accessing the web. In *Workshop on privacy in the electronic society (WPES)*, pp. 63–70. ACM, 2002. | Cit. on p. 145.

[197] Y. Elovici, B. Shapira and A. Maschiach. A new privacy model for web surfing. In *Proceedings of Next generation information technologies and systems (NGITS)*, pp. 45–57. Springer, 2002. | Cit. on pp. 52, 145.

[198] Y. Elovici, B. Shapira and A. Meshiach. Cluster-analysis attack against a PRivAte Web solution (PRAW). *Online information review*, 30(6):624–643, 2006. | Cit. on pp. 116, 123, 145, 146, 147, 148, 229.

[199] S. Englehardt and A. Narayanan. Online tracking: A 1-million-site measurement and analysis. In *Proceedings of Computer and communications security (CCS)*, pp. 1388–1401. ACM, 2016. | Cit. on pp. 171, 224.

[200] E. p. i. c. (EPIC). Search engine privacy. Online at `https://epic.org/privacy/search-engine/`. Last accessed on 15 August 2019. | Cit. on p. 113.

[201] R. Epstein and R. E. Robertson. The search engine manipulation effect (SEME) and its possible impact on the outcomes of elections. *Proceedings of the National Academy of Sciences*, 112(33):E4512–E4521, 2015. | Cit. on p. 114.

[202] K. Ermoshina, F. Musiani and H. Halpin. End-to-end encrypted messaging protocols: An overview. In *Proceedings of International conference on Internet science (INSCI)*, pp. 244–254. Springer, 2016. | Cit. on p. 41.

[203] A. Erola, J. Castellà-Roca, A. Viejo and J. M. Mateo-Sanz. Exploiting social networks to provide privacy in personalized web search. *Journal of systems and software*, 84(10):1734–1745, 2011. | Cit. on pp. 73, 115.

[204] Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data,

and repealing Directive 95/46/EC (General Data Protection Regulation). *Official Journal of the European Union (OJ)*, 59(1-88):294, 2016. | Cit. on pp. 41, 221, 246.

[205] A. B. Evnine, Z. Rosenstein, H. G. Lee and A. Dhesi. Methods and systems for making recommendations based on relationships, November 2017. US Patent number: US 2017/0344553 A1. | Cit. on p. 170.

[206] J. Farrell and P. Klemperer. Coordination and lock-in: Competition with switching costs and network effects. UC Berkeley: Competition policy center, 2006. | Cit. on p. 172.

[207] H. Federrath, K.-P. Fuchs, D. Herrmann and C. Piosecny. Privacy-preserving DNS: analysis of broadcast, range queries and mix-based protection methods. In *European symposium on research in computer security (ESORICS)*, pp. 665–683. Springer, 2011. | Cit. on p. 52.

[208] E. W. Felten. Declaration of Professor Edward W. Felten. ACLU v. Clapper, Case No. 13-cv-03994, Southern District Court of New York (26 August 2013). | Cit. on p. 171.

[209] A. T. Fernando, J. T. Du and H. Ashman. Personalisation of web search: Exploring search query parameters and user information privacy implications – The case of Google. In *Workshop on privacy-preserving IR: When information retrieval meets privacy and security (PIR)*, pp. 31–36. ACM SIGIR, 2014. | Cit. on p. 166.

[210] E. Ferrara, O. Varol, C. Davis, F. Menczer and A. Flammini. The rise of social bots. *Communications of the ACM*, 59(7):96–104, 2016. | Cit. on p. 217.

[211] D. Fifield, C. Lan, R. Hynes, P. Wegmann and V. Paxson. Blocking-resistant communication through domain fronting. *Proceedings on privacy enhancing technologies (PoPETs)*, 2015(2):46–64, 2015. | Cit. on p. 54.

[212] S. Fischer-Hübner, C. Köffel, J.-S. Pettersson, P. Wolkerstorfer, C. Graf, L. E. Holtz, U. König, H. Hedbom and B. Kellermann. HCI pattern collection – Version 2. PrimeLife Deliverable D4.1.3, 2010. | Cit. on p. 233.

[213] R. J. Fisher. Social desirability bias and the validity of indirect questioning. *Journal of consumer research*, 20(2):303–315, 1993. | Cit. on p. 212.

[214] D. Florêncio and C. Herley. A large-scale study of web password habits. In *World wide web conference (WWW)*, pp. 657–666. ACM, 2007. | Cit. on p. 205.

[215] A. Forget, S. Chiasson and R. Biddle. Persuasion as education for computer security. In *World conference on e-learning in corporate, government, healthcare, and higher education (E-Learn)*, pp. 822–829. Association for the advancement of computing in education (AACE), 2007. | Cit. on p. 97.

[216] M. Franz, B. Meyer and A. Pashalidis. Attacking unlinkability: The importance of context. In *Workshop on privacy enhancing technologies (PET)*, pp. 1–16. Springer, 2007. | Cit. on p. 79.

[217] D. Froomkin and J. McLaughlin. FBI vs. Apple establishes a new phase of the Crypto Wars. In *The Intercept* at `https://theintercept.com/2016/02/26/fbi-vs-apple-post-crypto-wars/`, February 2016. Last accessed on 19 August 2019. | Cit. on p. 49.

[218] X. Fu, B. Graham, R. Bettati and W. Zhao. On effectiveness of link padding for statistical traffic analysis attacks. In *Proceedings of Distributed computing systems*, pp. 340–347. IEEE, 2003. | Cit. on pp. 47, 232.

[219] C. Fuchs. Information and communication technologies and society: A contribution to the critique of the political economy of the Internet. *European journal of communication*, 24(1):69–87, 2009. | Cit. on p. 170.

[220] C. Fuchs and S. Sevignani. What is digital labour? What is digital work? What's their difference? And why do these questions matter for understanding social media? *tripleC. Communication, capitalism & critique*, 11(2):237–293, 2013. | Cit. on p. 161.

[221] A. Fuggetta. Open source software—an evaluation. *The journal of systems and software*, 66(1):77–90, 2003. | Cit. on p. 109.

[222] S. Furnell. End-user security culture: A lesson that will never be learnt? *Computer fraud & security*, 2008(4):6–9, 2008. | Cit. on p. 97.

[223] S. M. Furnell, P. Bryant and A. D. Phippen. Assessing the security perceptions of personal Internet users. *Computers & security*, 26(5):410–417, 2007. | Cit. on p. 97.

[224] G. G. Fuster. Inaccuracy as a privacy-enhancing tool. *Ethics and information technology*, 12(1):87–95, 2010. | Cit. on pp. 5, 6, 243, 246.

[225] R. G. Gallager. *Information theory and reliable communication*. John Wiley & Sons, Inc., New York, NY, USA, 1968. | Cit. on p. 152.

[226] K. Gallagher. Ad fraud estimates doubled. In *Business Insider* at `https://www.businessinsider.com/ad-fraud-estimates-doubled-2017-3`, March 2017. Last accessed on 20 August 2019. | Cit. on p. 105.

[227] K. Gallagher, S. Patil, B. Dolan-Gavitt, D. McCoy and N. Memon. Peeling the Onion's user experience layer: Examining naturalistic use of the Tor Browser. In *Proceedings of Computer and communications security (CCS)*, pp. 1290–1305. ACM, 2018. | Cit. on p. 114.

[228] K. Gallagher, S. Patil and N. Memon. New me: Understanding expert and non-expert perceptions and usage of the Tor anonymity network. In *Symposium on usable privacy and security (SOUPS)*, pp. 385–398. USENIX, 2017. | Cit. on pp. 96, 114.

[229] E. Gamma, R. Helm, R. Johnson and J. Vlissides. *Design patterns: Elements of reusable object-oriented software*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1995. | Cit. on pp. 222, 223, 236.

[230] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *Proceedings of Foundations of computer science symposium (FOCS)*, pp. 40–49. IEEE, 2013. | Cit. on p. 42.

[231] W. Gasarch. A survey on private information retrieval. *Bulletin of the European Association for Theoretical Computer Science*, 82(1):72–107, 2004. | Cit. on p. 39.

[232] B. Gedik and L. Liu. Location privacy in mobile systems: A personalized anonymization model. In *Proceedings of International conference on distributed computing systems (ICDCS)*, pp. 620–629. IEEE, 2005. | Cit. on p. 36.

[233] R. Gellman and P. Dixon. Many failures: A brief history of privacy self-regulation in the United States. Report, World Privacy Forum, October 2011. | Cit. on p. 3.

[234] C. Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009. | Cit. on p. 39.

[235] A. Gervais, R. Shokri, A. Singla, S. Capkun and V. Lenders. Quantifying web-search privacy. In *Proceedings of Computer and communications security (CCS)*, pp. 966–977. ACM, 2014. | Cit. on pp. 22, 82.

[236] R. C. Geyer, T. Klein and M. Nabi. Differentially private federated learning: A client level perspective. *Computing research repository (CoRR)*, arXiv:1712.07557, 2017. | Cit. on p. 244.

[237] B. Gierlichs, L. Batina, P. Tuyls and B. Preneel. Mutual information analysis. In *Proceedings of Cryptographic hardware and embedded systems (CHES)*, pp. 426–442. Springer, 2008. | Cit. on p. 192.

[238] M. Goddard. The EU General Data Protection Regulation (GDPR): European regulation that has a global impact. *International journal of market research*, 59(6):703–705, 2017. | Cit. on p. 4.

[239] K.-I. Goh, E. Oh, B. Kahng and D. Kim. Betweenness centrality correlation in social networks. *Physical Review E*, 67(017101):1–4, 2003. | Cit. on p. 181.

[240] I. Goldberg. Privacy-enhancing technologies for the Internet III: Ten years later. In *Digital privacy: Theory, technologies, and practices*, pp. 25–40. Auerbach Publications, 2007. | Cit. on pp. 3, 96.

[241] S. A. Golder, D. M. Wilkinson and B. A. Huberman. Rhythms of social interaction: Messaging within a massive online network. In *Proceedings of Communities and technologies*. Springer, 2007. | Cit. on p. 182.

[242] O. Goldreich. Extension to the multi-party case. In *Foundations of cryptography. Volume 2. Basic applications*, pp. 693–740. Cambridge University Press, 2004. | Cit. on p. 39.

[243] D. Goldschlag, M. Reed and P. Syverson. Onion routing. *Communications of the ACM*, 42(2):39–41, 1999. | Cit. on p. 28.

[244] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of computer and system sciences*, 28(2):270–299, 1984. | Cit. on pp. 28, 66.

[245] P. Golle and K. Partridge. On the anonymity of home/work location pairs. In *Proceedings of Pervasive computing*, pp. 390–397. Springer, 2009. | Cit. on p. 43.

[246] How search organizes information. In *Google Search Blog* at `https://www.google.com/search/howsearchworks/crawling-indexing/`. Last accessed on 30 July 2019. | Cit. on p. 32.

[247] M. Green. Cryptographic obfuscation and 'unhackable' software. In *A few thoughts on cryptographic engineering* at `https://blog.cryptographyengineering.com/2014/02/21/`

cryptographic-obfuscation-and/, February 2014. Last accessed on 21 August 2019. | Cit. on p. 19.

[248] M. Green. What's the matter with PGP? In *A few thoughts on cryptographic engineering* at https://blog.cryptographyengineering.com/2014/08/13/whats-matter-with-pgp/, August 2014. Last accessed on 21 August 2019. | Cit. on p. 205.

[249] A. Greenberg. Whatsapp just switched on end-to-end encryption for hundreds of millions of users. In *Wired* at https://www.wired.com/2014/11/whatsapp-encrypted-messaging/, November 2014. Last accessed on 21 August 2019. | Cit. on p. 177.

[250] A. Greenberg. After 3 years, why Gmail's end-to-end encryption is still vapor. In *Wired* at https://www.wired.com/2017/02/3-years-gmails-end-end-encryption-still-vapor/, February 2017. Last accessed on 21 August 2019. | Cit. on p. 201.

[251] A. Greiner. Reality check: Is ad fraud up or down? In *Forbes* at https://www.forbes.com/sites/forbesagencycouncil/2018/07/26/reality-check-is-ad-fraud-up-or-down/, July 2018. Last accessed on 21 August 2019. | Cit. on p. 105.

[252] B. Greschbach, G. Kreitz and S. Buchegger. The devil is in the metadata – New privacy challenges in decentralised online social networks. In *Pervasive computing and communications workshops (PerCom)*, pp. 333–339. IEEE, 2012. | Cit. on p. 172.

[253] M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *Proceedings of Mobile systems, applications and services (MobiSys)*, pp. 31–42. ACM, 2003. | Cit. on p. 36.

[254] Q. Gu, X. Chen, Z. Jiang and J. Wu. Sink-anonymity mobility control in wireless sensor networks. In *Proceedings of Wireless and mobile computing, networking and communications (WIMOB)*, pp. 36–41. IEEE, 2009. | Cit. on p. 48.

[255] P. Guarda and N. Zannone. Towards the development of privacy-aware systems. *Information and software technology*, 51(2):337–350, 2009. | Cit. on p. 221.

[256] S. Guha, K. Tang and P. Francis. NOYB: Privacy in online social networks. In *Workshop on online social networks (WOSN)*, pp. 49–54. ACM, 2008. | Cit. on pp. 201, 204.

[257] S. Gupta, L. Ulanova, S. Bhardwaj, P. Dmitriev, P. Raff and A. Fabijan. The anatomy of a large-scale experimentation platform. In *Proceedings of International conference on software architecture (ICSA)*, pp. 1–10. IEEE, 2018. | Cit. on p. 2.

[258] T. Gupta, N. Crooks, W. Mulhern, S. T. Setty, L. Alvisi and M. Walfish. Scalable and private media consumption with Popcorn. In *Proceedings of Networked systems design and implementation (NSDI)*, pp. 91–107. USENIX, 2016. | Cit. on pp. 31, 32.

[259] S. Gürses. *Multilateral privacy requirements analysis in online social network services*. PhD thesis, KU Leuven, 2010. | Cit. on pp. 16, 46.

[260] S. Gürses and C. Diaz. Two tales of privacy in online social networks. *IEEE Security & privacy*, 11(3):29–37, 2013. | Cit. on p. 211.

[261] S. Gürses, R. Overdorf and E. Balsa. POTs: the revolution will not be optimized? Presented at *Workshop on hot topics in privacy enhancing technologies (HotPETs)*, 2018. | Cit. on pp. 12, 53, 104, 108, 244, 245.

[262] S. Gürses, C. Troncoso and C. Diaz. Engineering privacy by design reloaded. Paper presented at *Amsterdam privacy conference (APC)*, 2015. | Cit. on pp. 220, 222, 230, 235, 236, 244.

[263] S. Gürses and J. Van Hoboken. Privacy after the agile turn. In *The Cambridge handbook of consumer privacy*, pp. 579–601. Cambridge University Press, 2018. | Cit. on p. 31.

[264] P. Gutmann. Lessons learned in implementing and deploying crypto software. In *Proceedings of USENIX Security symposium*, pp. 315–325. USENIX, 2002. | Cit. on p. 28.

[265] S. Hada. Zero-knowledge and code obfuscation. In *Proceedings of Advances in cryptology (ASIACRYPT)*, pp. 443–457. Springer, 2000. | Cit. on p. 19.

[266] M. Hafiz. A collection of privacy design patterns. In *Proceedings of Pattern languages of programs (PLoP)*, article no. 7. ACM, 2006. | Cit. on pp. 223, 230, 232, 233.

[267] M. Hafiz. A pattern language for developing privacy enhancing technologies. *Software: Practice and experience*, 43(7):769–787, 2013. | Cit. on pp. 222, 230, 233.

[268] M. Hafiz, P. Adamczyk and R. E. Johnson. Organizing security patterns. *IEEE Software*, 24(4):52–60, 2007. | Cit. on p. 235.

[269] Y. Y. Haimes. On the definition of resilience in systems. *Risk Analysis*, 29(4):498–501, 2009. | Cit. on p. 44.

[270] S. Hamadou, C. Palamidessi and V. Sassone. Quantifying leakage in the presence of unreliable sources of information. *Journal of computer and system sciences*, 88:27–52, 2017. | Cit. on pp. 78, 80.

[271] S. Hamadou, V. Sassone and C. Palamidessi. Reconciling belief and vulnerability in information flow. In *Symposium on security and privacy (S&P)*, pp. 79–92. IEEE, 2010. | Cit. on p. 79.

[272] A. Hannak, P. Sapiezynski, A. Molavi Kakhki, B. Krishnamurthy, D. Lazer, A. Mislove and C. Wilson. Measuring personalization of web search. In *Proceedings of World Wide Web conference (WWW)*, pp. 527–538. ACM, 2013. | Cit. on p. 166.

[273] A. Hannak, G. Soeller, D. Lazer, A. Mislove and C. Wilson. Measuring price discrimination and steering on e-commerce web sites. In *Proceedings of Internet measurement conference (IMC)*, pp. 305–318. ACM, 2014. | Cit. on p. 2.

[274] T. Hara, A. Suzuki, M. Iwata, Y. Arase and X. Xie. Dummy-based user location anonymization under real-world constraints. *IEEE Access*, 4:673–687, 2016. | Cit. on p. 229.

[275] W. Hartzog and F. Stutzman. Obscurity by design. *Washington Law Review*, 88(2):385–418, 2013. | Cit. on p. 202.

[276] T. He, E. N. Ciftcioglu, S. Wang and K. S. Chan. Location privacy in mobile edge clouds. In *Proceedings of International conference on distributed computing systems (ICDCS)*, pp. 2264–2269. IEEE, 2017. | Cit. on p. 51.

[277] C. Herley and D. Florêncio. How to login from an Internet café without worrying about keyloggers. Paper presented at *Symposium on usable privacy and security*, 2006. | Cit. on p. 50.

[278] J. Hermans, A. Pashalidis, F. Vercauteren and B. Preneel. A new RFID privacy model. In *Proceedings of European symposium on research in computer security (ESORICS)*, pp. 568–587. Springer, 2011. | Cit. on p. 67.

[279] J. Herrera-Joancomartí and C. Pérez-Solà. Online social honeynets: Trapping web crawlers in OSN. In *Proceedings of Modeling decisions for artificial intelligence (MDAI)*, pp. 1–16. Springer, 2011. | Cit. on p. 57.

[280] S. C. Herring. Slouching toward the ordinary: Current trends in computer-mediated communication. *New media & society*, 6(1):26–36, 2004. | Cit. on p. 169.

[281] A. Herzberg and H. Leibowitz. Can Johnny finally encrypt? Evaluating E2E-encryption in popular IM applications. In *Proceedings of Socio-technical aspects in security and trust (STAST)*. ACM, 2016. | Cit. on pp. 41, 205, 214.

[282] D. D. Hirsch. The law and policy of online privacy: Regulation, self-regulation, or co-regulation. *Seattle University Law Review*, 34(2):439–480, 2011. | Cit. on p. 3.

[283] M. Hirt. *Multi party computation: Efficient protocols, general adversaries, and voting*. PhD thesis, ETH Zurich, 2001. | Cit. on p. 40.

[284] L. Hix. A wonderful life: How postwar Christmas embraced spaceships, nukes, cellophane. In *Collectors Weekly* at `https://www.collectorsweekly.com/articles/postwar-christmas/`, Dec 2016. Last accessed on 5 October 2019. | Cit. on p. 46.

[285] J.-H. Hoepman. Privacy design strategies. In *Proceedings of ICT systems security and privacy protection (IFIP SEC)*, pp. 446–459. Springer, 2014. | Cit. on pp. 222, 233, 235.

[286] J.-H. Hoepman and B. Jacobs. Increased security through open source. *Communications of the ACM*, 50(1):79–83, 2007. | Cit. on pp. 214, 218.

[287] C. J. Hoofnagle. Privacy self regulation: A decade of disappointment. In *Consumer protection in the age of the 'information economy'*, pp. 379–402. Rouledge, 2005. | Cit. on p. 3.

[288] C. J. Hoofnagle, B. van der Sloot and F. Z. Borgesius. The European Union general data protection regulation: what it is and what it means. *Information & communications technology law*, 28(1):65–98, 2019. | Cit. on pp. 3, 4, 221, 246.

[289] Johns Hopkins EpiWatch: App and research study. Online at `https://www.hopkinsmedicine.org/epiwatch/`. Last accessed on 15 August 2019. | Cit. on p. 30.

[290] A. Houmansadr, C. Brubaker and V. Shmatikov. The parrot is dead: Observing unobservable network communications. In *Proceedings of Symposium on security and privacy (S&P)*, pp. 65–79. IEEE, 2013. | Cit. on p. 55.

[291] A. Houmansadr, T. J. Riedl, N. Borisov and A. C. Singer. I want my voice to be heard: IP over Voice-over-IP for unobservable censorship circumvention. In *Proceedings of Network & distributed system security symposium (NDSS)*. Internet Society, 2013. | Cit. on p. 54.

[292] A. E. Howe, I. Ray, M. Roberts, M. Urbanska and Z. Byrne. The psychology of security for the home computer user. In *Proceedings of Symposium on security and privacy (S&P)*, pp. 209–223. IEEE, 2012. | Cit. on p. 97.

[293] D. C. Howe. Surveillance countermeasures: Expressive privacy via obfuscation. *Datafied Research*, 4(1), 2015. | Cit. on pp. 6, 8, 107, 246.

[294] D. C. Howe and H. Nissenbaum. TrackMeNot: Resisting surveillance in web search. In *Lessons from the identity trail: Anonymity, privacy, and identity in a networked society*, chapter 23, pp. 417–436. Oxford University Press, 2009. | Cit. on pp. 6, 52, 98, 116, 123, 142, 160, 164, 166, 229.

[295] D. C. Howe and H. Nissenbaum. Engineering privacy and protest: a case study of AdNauseam. In *Proceedings of International workshop on privacy engineering (IWPE)*, pp. 57–64. IEEE, 2017. | Cit. on pp. 52, 98, 99, 105, 163, 166, 229.

[296] C. Huang, P. Kairouz, X. Chen, L. Sankar and R. Rajagopal. Context-aware generative adversarial privacy. *Entropy*, 19(12):656, 2017. | Cit. on p. 45.

[297] J. H. Huh, S. Verma, S. S. V. Rayala, R. B. Bobba, K. Beznosov and H. Kim. I don't use Apple Pay because it's less secure...: Perception of security and usability in mobile tap-and-pay. In *Proceedings of Workshop on usable security (USEC)*, pp. 1–12. Internet Society, 2017. | Cit. on p. 96.

[298] M. Isaac. Facebook's Mark Zuckerberg says he'll shift focus to users' privacy. In *The New York Times* at `https://www.nytimes.com/2019/03/06/technology/mark-zuckerberg-facebook-privacy.html`, March 2019. | Cit. on p. 201.

[299] ISO/IEC PDTR 27550. Information technology – Security techniques – Privacy engineering. ISO standard, International Organization for Standardization, December 2018. | Cit. on p. 238.

[300] P. R. Iyer, R. Sundar, M. Jha, K. Raman and M. Katzenellenbogen. Context enhanced marketing of content and targeted advertising to mobile device users, March 2011. US Patent number: US 2011/0066507 A1. | Cit. on p. 170.

[301] S. Jahid, P. Mittal and N. Borisov. EASiER: Encryption-based access control in social networks with efficient revocation. In *Proceedings of Symposium on information, computer and communications security (ASIACCS)*, pp. 411–415. ACM, 2011. | Cit. on p. 201.

[302] B. Jansen, M. Zhang, D. Booth, D. Park, Y. Zhang, A. Kathuria and P. Bonner. To what degree can log data profile a web searcher? *Proceedings of the American society for information science and technology*, 46(1):1–19, 2009. | Cit. on p. 113.

[303] B. J. Jansen and D. Booth. Classifying web queries by topic and user intent. In *Extended abstracts on Human factors in computing systems (CHI)*, pp. 4285–4290. ACM, 2010. | Cit. on p. 117.

[304] J. Jaworska and M. Sydow. Behavioural targeting in on-line advertising: An empirical study. In *Proceedings of Web information systems engineering (WISE)*, pp. 62–76. Springer, 2008. | Cit. on p. 105.

[305] T. Jiang and M. Li. On the approximation of shortest common supersequences and longest common subsequences. *SIAM Journal on computing*, 24(5):1122–1139, 1995. | Cit. on p. 94.

[306] N. F. Johnson and S. Jajodia. Exploring steganography: Seeing the unseen. *Computer*, 31(2):26–34, 1998. | Cit. on p. 55.

[307] R. Jones, R. Kumar, B. Pang and A. Tomkins. "I know what you did last summer" — Query logs and user privacy. In *Proceedings of Conference on information and knowledge management (CIKM)*, pp. 909–914. ACM, 2007. | Cit. on p. 113.

[308] M. Joye and F. Olivier. Side-channel analysis. In *Encyclopedia of cryptography and security*, pp. 1198–1204. Springer, 2011. | Cit. on p. 49.

[309] M. Juarez, M. Imani, M. Perry, C. Diaz and M. Wright. Toward an efficient website fingerprinting defense. In *Proceedings of European symposium on research in computer security (ESORICS)*, pp. 27–46. Springer, 2016. | Cit. on pp. 48, 55.

[310] M. Juarez and V. Torra. Dispa: An intelligent agent for private web search. In *Advanced research in data privacy*, pp. 389–405. Springer, 2015. | Cit. on p. 114.

[311] A. Juels. Targeted advertising... and privacy too. In *Proceedings of The cryptographer's track at RSA (CT-RSA)*, pp. 408–424. Springer, 2001. | Cit. on p. 244.

[312] A. Juels and M. Sudan. A fuzzy vault scheme. *Designs, codes and cryptography*, 38(2):237–257, 2006. | Cit. on p. 50.

[313] B. Kang, S. C. Goh and M. Kim. Private web search with constant round efficiency. In *Proceedings of International conference on information systems security and privacy (ICISSP)*, pp. 205–212. IEEE, 2015. | Cit. on p. 115.

[314] S. P. Kasiviswanathan and A. Smith. On the 'semantics' of differential privacy: A bayesian formulation. *Journal of privacy and confidentiality*, 6(1):1–16, 2014. | Cit. on p. 69.

[315] S. P. Kasiviswanathan and A. Smith. A note on differential privacy: Defining resistance to arbitrary side information. *Computing research repository (CoRR)*, arXiv:0803.3946, 2008. | Cit. on p. 69.

[316] K. Kenan. *Cryptography in the database: The last line of defense.* Addison-Wesley, 2006. | Cit. on p. 20.

[317] H. Kennedy, D. Elgesem and C. Miguel. On fairness: User perspectives on social media data mining. *Convergence*, 23(3):270–288, 2017. | Cit. on p. 3.

[318] S. T. Kent. Encryption-based protection for interactive user/computer communication. In *Proceedings of Symposium on data communications (SIGCOMM)*, pp. 5.7–5.13. ACM, 1977. | Cit. on pp. 47, 172.

[319] S. Khattak, T. Elahi, L. Simon, C. M. Swanson, S. J. Murdoch and I. Goldberg. SoK: Making sense of censorship resistance systems. *Proceedings on privacy enhancing technologies (PoPETs)*, 2016(4):37–61, 2016. | Cit. on pp. 54, 240.

[320] H. Kido, Y. Yanagisawa and T. Satoh. An anonymous communication technique using dummies for location-based services. In *Proceedings of International conference on pervasive services (ICPS)*, pp. 88–97. IEEE, 2005. | Cit. on p. 36.

[321] H. Kido, Y. Yanagisawa and T. Satoh. Protection of location privacy using dummies for location-based services. In *Proceedings of International conference on data engineering workshops (ICDEW)*, pp. 1248–1248. IEEE, 2005. | Cit. on pp. 36, 51, 89, 229.

[322] D. Kifer and A. Machanavajjhala. No free lunch in data privacy. In *Proceedings of International conference on management of data (SIGMOD)*, pp. 193–204. ACM, 2011. | Cit. on pp. 33, 67, 77.

[323] I. Kirlappos and M. A. Sasse. Security education against phishing: A modest proposal for a major rethink. *IEEE Security & privacy*, 10(2):24–32, 2012. | Cit. on p. 97.

[324] I. Kirlappos and M. A. Sasse. What usable security really means: Trusting and engaging users. In *Proceedings of Human aspects of information security, privacy, and trust (HAS)*, pp. 69–78. Springer, 2014. | Cit. on p. 98.

[325] Á. Kiss, J. Liu, T. Schneider, N. Asokan and B. Pinkas. Private set intersection for unequal set sizes with mobile applications. *Proceedings on privacy enhancing technologies (PoPETs)*, 2017(4):177–197, 2017. | Cit. on p. 42.

[326] N. Kobeissi and A. Breault. Cryptocat: Adopting accessibility and ease of use as security properties. *Computing research repository (CoRR)*, arXiv:1306.5156, 2013. | Cit. on p. 204.

[327] P. Kocher, J. Jaffe and B. Jun. Differential power analysis. In *Proceedings of Advances in cryptology (CRYPTO)*, pp. 388–397. Springer, 1999. | Cit. on pp. 7, 28, 49.

[328] K. G. Kogos, K. S. Filippova and A. V. Epishkina. Fully homomorphic encryption schemes: The state of the art. In *Proceedings of Conference of Russian young researchers in electrical and electronic engineering (EIConRus)*, pp. 463–466. IEEE, 2017. | Cit. on p. 40.

[329] J. Kokott and C. Sobotta. The distinction between privacy and data protection in the jurisprudence of the CJEU and the ECtHR. *International data privacy law*, 3(4):222–228, 2013. | Cit. on p. 41.

[330] B. Köpf and G. Smith. Vulnerability bounds and leakage resilience of blinded cryptography under timing attacks. In *Proceedings of Computer security foundations symposium (CSF)*, pp. 44–56. IEEE, 2010. | Cit. on pp. 75, 76.

[331] A. Korolova. Privacy violations using microtargeted ads: A case study. In *Proceedings of International conference on data mining workshops (ICDMW)*, pp. 474–482. IEEE, 2010. | Cit. on p. 244.

[332] A. Korolova, K. Kenthapadi, N. Mishra and A. Ntoulas. Releasing search queries and clicks privately. In *Proceedings of World wide web (WWW)*, pp. 171–180. ACM, 2009. | Cit. on p. 68.

[333] A. Korolova, R. Motwani, S. U. Nabar and Y. Xu. Link privacy in social networks. In *Proceedings of Conference on information and*

knowledge management (CIKM), pp. 289–298. ACM, 2008. | Cit. on p. 171.

[334] M. Kosinski, D. Stillwell and T. Graepel. Private traits and attributes are predictable from digital records of human behavior. *Proceedings of the National Academy of Sciences*, 110(15):5802–5805, 2013. | Cit. on p. 2.

[335] T. Kuflik, B. Shapira, Y. Elovici and A. Maschiach. Privacy preservation improvement by learning optimal profile generation rate. In *Proceedings of User Modeling (UM)*, pp. 168–177. Springer, 2003. | Cit. on p. 145.

[336] G. Kumparak. Apple explains exactly how secure iMessage really is. In *TechCrunch* at `https://techcrunch.com/2014/02/27/apple-explains-exactly-how-secure-imessage-really-is/`, February 2014. Last accessed on 27 August 2019. | Cit. on p. 203.

[337] E. Kushilevitz and R. Ostrovsky. Replication is not needed: Single database, computationally-private information retrieval. In *Proceedings of Foundations of computer science (FOCS)*, pp. 364–373. IEEE, 1997. | Cit. on p. 115.

[338] N. Kuzurin, A. Shokurov, N. Varnovsky and V. Zakharov. On the concept of software obfuscation in computer security. In *Proceedings of Information security conference (ISC)*, pp. 281–298. Springer, 2007. | Cit. on pp. 17, 19.

[339] S. Lahlou, M. Langheinrich and C. Röcker. Privacy and trust issues with invisible computers. *Communications of the ACM*, 48(3):59–60, 2005. | Cit. on p. 221.

[340] S. Landau. Making sense from Snowden: What's significant in the NSA surveillance revelations. *IEEE Security & privacy*, 11(4):54–63, 2013. | Cit. on pp. 170, 171.

[341] S. Landau. Control use of data to protect privacy. *Science*, 347(6221):504–506, 2015. | Cit. on p. 3.

[342] B. Laurie, A. Langley and E. Kasper. Certificate transparency. RFC 6962, IETF, 2013. | Cit. on p. 203.

[343] J. Lazar, J. H. Feng and H. Hochheiser. *Research methods in human-computer interaction*. Wiley, 2010. | Cit. on p. 208.

[344] C. S. Leberknight, M. Chiang, H. V. Poor and F. Wong. A taxonomy of Internet censorship and anti-censorship. Draft version, December 2010. | Cit. on p. 54.

[345] K. Lee, J. Caverlee and S. Webb. The Social Honeypot Project: Protecting online communities from spammers. In *Proceedings of World wide web (WWW)*, pp. 1139–1140. ACM, 2010. | Cit. on p. 57.

[346] L. Lee, D. Fifield, N. Malkin, G. Iyer, S. Egelman and D. Wagner. A usability evaluation of Tor Launcher. *Proceedings on privacy enhancing technologies (PoPETs)*, 2017(3):90–109, 2017. | Cit. on p. 99.

[347] T. Lee. Google purges the Play Store of click fraud apps. In *übergizmo* at `https://www.ubergizmo.com/2018/12/google-purge-play-store-click-fraud-apps/`, October 2018. Last accessed on 27 August 2019. | Cit. on p. 105.

[348] J. Lenhard, L. Fritsch and S. Herold. A literature study on privacy patterns research. In *Proceedings of Software engineering and advanced applications (SEAA)*, pp. 194–201. IEEE, 2017. | Cit. on pp. 222, 236, 237.

[349] B. N. Levine, M. K. Reiter, C. Wang and M. Wright. Timing attacks in low-latency mix systems. In *Proceedings of Financial cryptography (FC)*, pp. 251–265. Springer, 2004. | Cit. on p. 48.

[350] S. Li, H. Guo and N. Hopper. Measuring information leakage in website fingerprinting attacks and defenses. In *Proceedings of Computer and communications security (CCS)*, pp. 1977–1992. ACM, 2018. | Cit. on p. 84.

[351] T. Li and N. Li. On the tradeoff between privacy and utility in data publishing. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 517–526. ACM, 2009. | Cit. on p. 22.

[352] Y. Li and M. Chen. Privacy preserving joins. In *Proceedings of International conference on data engineering (ICDE)*, pp. 1352–1354. IEEE, 2008. | Cit. on pp. 48, 49.

[353] M. Liberatore and B. N. Levine. Inferring the source of encrypted HTTP connections. In *Proceedings of Computer and communications security (CCS)*, pp. 255–263. ACM, 2006. | Cit. on p. 55.

[354] L. Lilien, Z. H. Kamal, V. Bhuse and A. Gupta. The concept of opportunistic networks and their research challenges in privacy and security. In *Mobile and wireless network security and privacy*, chapter 5, pp. 85–117. Springer, 2006. | Cit. on p. 56.

[355] Y. Lindell and E. Waisbard. Private web search with malicious adversaries. In *Proceedings of Privacy enhancing technologies symposium (PETS)*, pp. 220–235. Springer, 2010. | Cit. on p. 115.

[356] H. Liu, X. Li, H. Li, J. Ma and X. Ma. Spatiotemporal correlation-aware dummy-based privacy protection scheme for location-based services. In *Proceedings of IEEE Conference on computer communications (INFOCOM)*, pp. 1–9. IEEE, 2017. | Cit. on p. 229.

[357] D. Lowd and C. Meek. Adversarial learning. In *Proceedings of Knowledge discovery in data mining (KDD)*, pp. 641–647. ACM, 2005. | Cit. on pp. 53, 104.

[358] H. Lu, C. S. Jensen and M. L. Yiu. Pad: Privacy-area aware, dummy-based location privacy in mobile services. In *Proceedings of Workshop on data engineering for wireless and mobile access (MobiDE)*, pp. 16–23. ACM, 2008. | Cit. on pp. 36, 51, 229.

[359] Y. Lu and G. Tsudik. Towards plugging privacy leaks in the domain name system. In *Proceedings of Peer-to-peer computing (P2P)*, pp. 1–10. IEEE, 2010. | Cit. on p. 52.

[360] N. B. Lucena, J. Pease, P. Yadollahpour and S. J. Chapin. Syntax and semantics-preserving application-layer protocol steganography. In *Proceedings of Information hiding (IH)*, pp. 164–179. Springer, 2005. | Cit. on pp. 27, 38.

[361] W. Luo, Q. Xie and U. Hengartner. FaceCloak: An architecture for user privacy on social networking sites. In *Proceedings of Computational science and engineering (CSE)*, volume 3, pp. 26–33. IEEE, 2009. | Cit. on pp. 201, 204.

[362] D. Lyon. *Surveillance society: Monitoring everyday life*. McGraw-Hill Education, 2001. | Cit. on p. 2.

[363] D. Lyon. Surveillance, Snowden, and big data: Capacities, consequences, critique. *Big data & society*, 1(2):1–13, 2014. | Cit. on p. 170.

[364] A. Machanavajjhala, J. Gehrke, D. Kifer and M. Venkitasubramaniam. ℓ-diversity: Privacy beyond k-anonymity. *ACM Transactions on knowledge discovery from data (TKDD)*, 1(1):3, 2007. | Cit. on pp. 22, 159.

[365] L. Magid. How (and why) to turn off Google's personalized search results. In *Forbes* at `https://www.forbes.com/sites/larrymagid/2012/01/13/how-and-why-to-turn-off-googles-personalized-search-results/`. Last accessed on 5 October 2019. | Cit. on p. 166.

[366] E. Magkos, P. Kotzanikolaou, M. Magioladitis, S. Sioutas and V. S. Verykios. Towards secure and practical location privacy through private equality testing. In *Proceedings of Privacy in statistical databases (PSD)*, pp. 312–325. Springer, 2014. | Cit. on p. 36.

[367] E. Marin, D. Singelée, F. D. Garcia, T. Chothia, R. Willems and B. Preneel. On the (in)security of the latest generation implantable cardiac defibrillators and how to secure them. In *Proceedings of Annual conference on computer security applications (ACSAC)*, pp. 226–236. ACM, 2016. | Cit. on p. 20.

[368] R. Masood, D. Vatsalan, M. Ikram and M. A. Kaafar. Incognito: A method for obfuscating web data. In *Proceedings of World Wide Web conference (WWW)*, pp. 267–276. International World Wide Web conferences steering committee, 2018. | Cit. on p. 115.

[369] A. Mathur, J. Vitak, A. Narayanan and M. Chetty. Characterizing the use of browser-based blocking extensions to prevent online tracking. In *Proceedings of Symposium on usable privacy and security (SOUPS)*, pp. 103–116. USENIX, 2018. | Cit. on pp. 5, 96, 97, 99, 105, 214, 224, 225.

[370] L. Matsakis and I. Lapowsky. Everything we know about Facebook's massive security breach. In *Wired* at `https://www.wired.com/story/facebook-security-breach-50-million-accounts/`, September 2018. Last accessed on 5 October 2019. | Cit. on p. 201.

[371] N. Mavrogiannopoulos, N. Kisserli and B. Preneel. A taxonomy of self-modifying code for obfuscation. *Computers & Security*, 30(8):679 – 691, 2011. | Cit. on pp. 17, 18.

[372] J. Mayer, P. Mutchler and J. C. Mitchell. Evaluating the privacy properties of telephone metadata. *Proceedings of the National Academy of Sciences (PNAS)*, 113(20):5536–5541, 2016. | Cit. on p. 171.

[373] A. Mazzia, K. LeFevre and E. Adar. The PViz comprehension tool for social network privacy settings. In *Proceedings of Symposium on usable privacy and security (SOUPS)*, pp. 1–13. ACM, 2012. | Cit. on p. 202.

[374] A. W. E. McDonald, S. Afroz, A. Caliskan, A. Stolerman and R. Greenstadt. Use fewer instances of the letter "i": Toward writing style anonymization. In *Proceedings of Privacy enhancing technologies symposium (PETS)*, pp. 299–318. Springer, 2012. | Cit. on p. 43.

[375] A. McIver, C. Morgan, G. Smith, B. Espinoza and L. Meinicke. Abstract channels and their robust information-leakage ordering. In *Proceedings of Principles of security and trust (POST)*, pp. 83–102. Springer, 2014. | Cit. on p. 75.

[376] B. McMahan and D. Ramage. Federated learning: Collaborative machine learning without centralized training data. In *Google AI Blog* at `https://ai.googleblog.com/2017/04/federated-learning-collaborative.html`, April 2017. Last accessed on 5 October 2019. | Cit. on p. 32.

[377] F. McSherry and I. Mironov. Differentially private recommender systems: Building privacy into the Netflix Prize contenders. In *Proceedings of Knowledge discovery and data mining (KDD)*, pp. 627–636. ACM, 2009. | Cit. on p. 68.

[378] A. McStay. *The mood of information: a critique of online behavioural advertising*. Continuum, 2011. | Cit. on p. 103.

[379] M. S. Melara, A. Blankstein, J. Bonneau, E. W. Felten and M. J. Freedman. CONIKS: Bringing key transparency to end users. In *Proceedings of USENIX Security symposium*, pp. 383–398. USENIX, 2015. | Cit. on p. 205.

[380] W. Melicher, M. Sharif, J. Tan, L. Bauer, M. Christodorescu and P. G. Leon. (Do not) track me sometimes: Users' contextual preferences for web tracking. *Proceedings on privacy enhancing technologies (PoPETs)*, 2016(2):135–154, 2016. | Cit. on pp. 97, 214.

[381] I. Mendoza and L. A. Bygrave. The right not to be subject to automated decisions based on profiling. In *EU Internet law. Regulation and enforcement*, pp. 77–98. Springer, 2017. | Cit. on p. 4.

[382] Microsoft. Migration and interoperability guidance for organizations using teams together with skype for business. Online at `https://docs.microsoft.com/en-us/microsoftteams/migration-interop-guidance-for-teams-with-skype`, August 2019. Last accessed on 5 October 2019. | Cit. on p. 170.

[383] J. Mikians, L. Gyarmati, V. Erramilli and N. Laoutaris. Detecting price and search discrimination on the Internet. In *Proceedings of Hot topics in networks (Hotnets)*, pp. 79–84. ACM, 2012. | Cit. on p. 114.

[384] Z. Miners. End-to-end encryption needs to be easier for users before Facebook embraces it. In *PCWorld* at `https://www.pcworld.com/article/2109582/end-to-end-encryption-needs-to-be-easier-for-users-before-facebook-embraces-it.html`, March 2014. Last accessed on 5 October 2019. | Cit. on p. 201.

[385] J. Mitchell. How Google Search really works. In *readwrite* at `https://readwrite.com/2012/02/29/interview_changing_engines_mid-`

flight_qa_with_goog/, February 2012. Last accessed on 5 October 2019. | Cit. on p. 88.

[386] H. Mohajeri Moghaddam, B. Li, M. Derakhshani and I. Goldberg. SkypeMorph: Protocol obfuscation for Tor bridges. In *Proceedings of Computer and communications security (CCS)*, pp. 97–108. ACM, 2012. | Cit. on p. 54.

[387] A. Mohammed. *The computational complexity of program obfuscation.* PhD thesis, University of Virginia, 2018. | Cit. on p. 20.

[388] A. Mols and S. Janssen. Not interesting enough to be followed by the NSA: An analysis of Dutch privacy attitudes. *Digital journalism*, 5(3):277–298, 2017. | Cit. on pp. 3, 4.

[389] E. Morozov. *To save everything, click here: The folly of technological solutionism.* Public Affairs, 2013. | Cit. on p. 246.

[390] mPower: Mobile Parkinson disease study. Online at https://parkinsonmpower.org/. Last accessed on 5 October 2019. | Cit. on p. 30.

[391] T. G. Muir and D. L. Bradley. Underwater acoustics: A brief historical overview through World War II. *Acoustics today*, 12(3):40–48, 2016. | Cit. on p. 46.

[392] A. Munster. Data undermining: The work of networked art in an age of imperceptibility. In *Networked: A networked book about networked art.* Turbulence.org, 2009. | Cit. on p. 244.

[393] D. Murakami Wood. What is global surveillance? Towards a relational political economy of the global surveillant assemblage. *Geoforum*, 49:317–326, 2013. | Cit. on p. 3.

[394] S. J. Murdoch and P. Zieliński. Sampled traffic analysis by Internet-exchange-level adversaries. In *Proceedings of Workshop on privacy enhancing technologies (PET)*, pp. 167–183. Springer, 2007. | Cit. on p. 43.

[395] M. Murphy. Millions of Facebook user records exposed in data breach. In *The Telegraph* at https://www.telegraph.co.uk/technology/2019/04/03/millions-facebook-user-records-exposed-data-breach/, April 2019. Last accessed on 5 October 2019. | Cit. on p. 201.

[396] M. Murugesan and C. Clifton. Providing privacy through plausibly deniable search. In *Proceedings of SIAM International conference on data mining (SDM)*, pp. 768–779. SIAM, 2009. | Cit. on pp. 123, 138.

[397] M. Murugesan and C. W. Clifton. Plausibly deniable search. In *Proceedings of Workshop on secure knowledge management (SKM)*, November 2008. | Cit. on pp. 52, 116, 123, 138.

[398] M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *Proceedings of Symposium on theory of computing (STOC)*, pp. 427–437. ACM, 1990. | Cit. on p. 66.

[399] A. Narayanan. What happened to the crypto dream?, Part 1. *IEEE security & privacy*, 11(2):75–76, 2013. | Cit. on p. 40.

[400] A. Narayanan. What happened to the crypto dream?, Part 2. *IEEE Security & privacy*, 11(3):68–71, 2013. | Cit. on pp. 8, 40, 41, 42, 60, 109, 221.

[401] A. Narayanan and V. Shmatikov. Obfuscated databases and group privacy. In *Proceedings of Computer and communications security (CCS)*, pp. 102–111. ACM, 2005. | Cit. on pp. 15, 17, 19.

[402] A. Narayanan and V. Shmatikov. De-anonymizing social networks. In *Proceedings of Symposium on security and privacy (S&P)*, pp. 173–187. IEEE, 2009. | Cit. on pp. 43, 171.

[403] J. Naughton. Facebook's new encrypted network will give criminals the privacy they crave. In *The Guardian* at `https://www.theguardian.com/commentisfree/2019/mar/17/facebook-encrypted-network-gives-criminals-privacy-they-crave-john-naughton`, March 2019. Last accessed on 5 October 2019. | Cit. on p. 201.

[404] J. P. Nehf. Recognizing the societal value in information privacy. *Washington law review*, 78(1):1–92, 2003. | Cit. on pp. 2, 3, 4.

[405] R. E. Newman-Wolfe and B. R. Venkatraman. High level prevention of traffic analysis. In *Proceedings of Annual computer security applications conference (ACSAC)*, pp. 102–109. IEEE, 1991. | Cit. on pp. 47, 172.

[406] E. J. Neystadt, R. Karidi, Y. T. Weisfeild, R. Varshavsky, A. Oron and K. Radinsky. Social network based contextual ranking, August 2012. US Patent number: US 2012/0209832 A1. | Cit. on p. 170.

[407] T. T. Nguyen, P.-M. Hui, F. M. Harper, L. Terveen and J. A. Konstan. Exploring the filter bubble: the effect of using recommender systems on content diversity. In *Proceedings of World Wide Web conference (WWW)*, pp. 677–686. ACM, 2014. | Cit. on pp. 2, 114.

[408] S. Nichols. 3ve offline: Countless Windows PCs using 1.7m IP addresses hacked to 'view' up to 12 billion adverts a day. In *The Register* at `https://www.theregister.co.uk/2018/11/28/3ve_ad_fraud_men_charged/`, November 2018. Last accessed on 5 October 2019. | Cit. on p. 105.

[409] N. Nikiforakis, A. Kapravelos, W. Joosen, C. Kruegel, F. Piessens and G. Vigna. Cookieless monster: Exploring the ecosystem of web-based device fingerprinting. In *Proceedings of Symposium on security and privacy (S&P)*, pp. 541–555. IEEE, 2013. | Cit. on p. 114.

[410] H. Nissenbaum. From preemption to circumvention: If technology regulates, why do we need regulation (and vice versa)? *Berkeley technology law journal*, 26(3):1367–1386, 2011. | Cit. on p. 246.

[411] B. Niu, Z. Zhang, X. Li and H. Li. Privacy-area aware dummy generation algorithms for location-based services. In *Proceedings of International conference on communications (ICC)*, pp. 957–962. IEEE, 2014. | Cit. on p. 229.

[412] G. Norcie, J. Blythe, K. Caine and L. J. Camp. Why Johnny can't blow the whistle: Identifying and reducing usability issues in anonymity systems. In *Proceedings of Workshop on usable security (USEC)*. Internet Society, 2014. | Cit. on p. 96.

[413] G. Norcie, K. Caine and L. J. Camp. Eliminating stop-points in the installation and use of anonymity systems: A usability evaluation of the Tor Browser Bundle. Presented at *Workshop on hot topics in privacy enhancing technologies (HotPETs)*, 2012. | Cit. on p. 99.

[414] M. Oates, Y. Ahmadullah, A. Marsh, C. Swoopes, S. Zhang, R. Balebako and L. F. Cranor. Turtles, locks, and bathrooms: Understanding mental models of privacy through illustration. *Proceedings on privacy enhancing technologies (PoPETs)*, 2018(4):5–32, 2018. | Cit. on p. 96.

[415] S. E. Oh, S. Li and N. Hopper. Fingerprinting keywords in search queries over Tor. *Proceedings on privacy enhancing technologies (PoPETs)*, 2017(4):251–270, 2017. | Cit. on p. 114.

[416] P. Olson. Facebook is committed to WhatsApp encryption, but could bypass it too. In *Forbes* at `https://www.forbes.com/sites/parmyolson/2018/09/27/facebook-is-committed-to-whatsapp-encryption-but-could-bypass-it-too/`, September 2018. Last accessed on 5 October 2019. | Cit. on p. 203.

[417] F. Olumofin, P. K. Tysowski, I. Goldberg and U. Hengartner. Achieving efficient query privacy for location based services. In *Proceedings of Privacy enhancing technologies symposium (PETS)*, pp. 93–110. Springer, 2010. | Cit. on p. 36.

[418] Y. Orito, Y. Fukuta and K. Murata. I will continue to use this nonetheless: Social media survive users' privacy concerns. *International journal of virtual worlds and human computer interaction*, 2:92–107, 2014. | Cit. on p. 4.

[419] R. Ostrovsky and W. E. Skeith. A survey of single-database private information retrieval: Techniques and applications. In *Proceedings of Workshop on Public key cryptography (PKC)*, pp. 393–411. Springer, 2007. | Cit. on p. 39.

[420] S. Oya, C. Troncoso and F. Pérez-González. Do dummies pay off? Limits of dummy traffic protection in anonymous communications. In *Proceedings of Privacy enhancing technologies symposium (PETS)*, pp. 204–223. Springer, 2014. | Cit. on p. 172.

[421] S. Oya, C. Troncoso and F. Pérez-González. Is geo-indistinguishability what you are looking for? In *Proceedings of Workshop on privacy in the electronic society (WPES)*, pp. 137–140. ACM, 2017. | Cit. on pp. 77, 78.

[422] X. Page, B. P. Knijnenburg and A. Kobsa. What a tangled web we weave: Lying backfires in location-sharing social media. In *Proceedings of Computer supported cooperative work and social computing (CSCW)*, pp. 273–284. ACM, 2013. | Cit. on p. 5.

[423] A. Panchenko, F. Lanze, A. Zinnen, M. Henze, J. Pennekamp, K. Wehrle and T. Engel. Website fingerprinting at Internet scale. In *Proceedings of The network and distributed system security symposium (NDSS)*, pp. 1–15. Internet Society, 2016. | Cit. on p. 94.

[424] A. Panchenko, L. Niessen, A. Zinnen and T. Engel. Website fingerprinting in onion routing based anonymization networks. In *Proceedings of Workshop on privacy in the electronic society (WPES)*, pp. 103–114. ACM, 2011. | Cit. on pp. 48, 55.

[425] H. H. Pang, X. Xiao and J. Shen. Obfuscating the topical intention in enterprise text search. In *Proceedings of International conference on data engineering (ICDE)*, pp. 1168–1179. IEEE, 2012. | Cit. on pp. 157, 163.

[426] H. Pang, X. Ding and X. Xiao. Embellishing text search queries to protect user privacy. *Proceedings of the Very Large Data Bases Endowment (PVLDB)*, 3(1-2):598–607, 2010. | Cit. on pp. 52, 156.

[427] R. Parameswaran and D. M. Blough. Privacy preserving collaborative filtering using data obfuscation. In *Proceedings of Granular Computing (GRC)*, pp. 380–380. IEEE, 2007. | Cit. on p. 22.

[428] E. Pariser. *The filter bubble: How the new personalized web is changing what we read and how we think*. Penguin, 2011. | Cit. on p. 2.

[429] A. Pashalidis and B. Preneel. Evaluating tag-based preference obfuscation systems. *IEEE Transactions on knowledge and data engineering (TKDE)*, 24(9):1613–1623, 2012. | Cit. on p. 22.

[430] S. T. Peddinti and N. Saxena. On the privacy of web search based on query obfuscation: A case study of TrackMeNot. In *Proceedings of Privacy enhancing technologies symposium (PETS)*, pp. 19–37. Springer, 2010. | Cit. on pp. 142, 149.

[431] W. L. Perry. *Predictive policing: The role of crime forecasting in law enforcement operations*. Rand Corporation, 2013. | Cit. on p. 103.

[432] A. Petit, T. Cerqueus, S. B. Mokhtar, L. Brunie and H. Kosch. PEAS: Private, efficient and accurate web search. In *Proceedings of Trust, security and privacy in computing and communications (Trustcom)*, volume 1, pp. 571–580. IEEE, 2015. | Cit. on p. 165.

[433] H. Petrie and N. Bevan. The evaluation of accessibility, usability, and user experience. In C. Stephanidis, editor, *The Universal Access Handbook*, chapter 20. CRC Press, 2009. | Cit. on p. 101.

[434] A. Pfitzmann and M. Hansen. A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management. Online, TU Dresden, August 2010. v0.34. | Cit. on pp. 43, 47.

[435] A. M. Piotrowska, J. Hayes, T. Elahi, S. Meiser and G. Danezis. The Loopix anonymity system. In *Proceedings of USENIX Security symposium*, pp. 1199–1216. USENIX, 2017. | Cit. on p. 95.

[436] J. O. Pliam. On the incomparability of entropy and marginal guesswork in brute-force attacks. In *Proceedings of International conference on cryptology in India (Indocrypt)*, pp. 67–79. Springer, 2000. | Cit. on p. 74.

[437] H. Polat and W. Du. Privacy-preserving collaborative filtering. *International journal of electronic commerce*, 9(4):9–35, 2005. | Cit. on p. 16.

[438] privacypatterns.eu - collecting patterns for better privacy. At `https://privacypatterns.eu`. Last accessed on 5 October 2019. | Cit. on pp. 223, 230, 232, 233, 234.

[439] privacypatterns.org. At `https://privacypatterns.org`. Last accessed on 5 October 2019. | Cit. on pp. 223, 230, 232, 233, 234, 237.

[440] C. Rackoff and D. R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *Proceedings of Advances in cryptology (CRYPTO)*, pp. 433–444. Springer, 1991. | Cit. on p. 66.

[441] E. Rader. Awareness of behavioral tracking and information privacy concern in Facebook and Google. In *Proceedings of Symposium on usable privacy and security (SOUPS)*, pp. 51–67. USENIX, 2014. | Cit. on p. 97.

[442] K.-J. Räihä and E. Ukkonen. The shortest common supersequence problem over binary alphabet is NP-complete. *Theoretical computer science*, 16(2):187–198, 1981. | Cit. on p. 94.

[443] P. Rajivan and L. J. Camp. Influence of privacy attitude and privacy cue framing on Android app choices. Presented at Workshop on privacy indicators (WPI) at the Symposium on usable security and privacy (SOUPS), 2016. | Cit. on pp. 97, 215.

[444] A. Rao, F. Schaub, N. Sadeh, A. Acquisti and R. Kang. Expecting the unexpected: Understanding mismatched privacy expectations online. In *Proceedings of Symposium on usable privacy and security (SOUPS)*, pp. 77–96. USENIX, 2016. | Cit. on p. 96.

[445] J. R. Rao and P. Rohatgi. Can pseudonymity really guarantee privacy? In *Proceedings of USENIX Security symposium*, pp. 85–96. USENIX, 2000. | Cit. on p. 43.

[446] A. Rayaprolu. Click fraud: A bane of contention in the digital ad ecosystem. In *Forbes India* at `http://www.forbesindia.com/blog/technology/click-fraud-a-bane-of-contention-in-the-digital-ad-ecosystem/`, July 2018. Last accessed on 5 October 2019. | Cit. on p. 105.

[447] J.-F. Raymond. Traffic analysis: Protocols, attacks, design issues and open problems. In *Proceedings of Workshop on design issues in anonymity and unobservability (PET)*, pp. 10–29. Springer, 2000. | Cit. on p. 176.

[448] D. Rebollo-Monedero and J. Forné. Optimized query forgery for private information retrieval. *IEEE Transactions on information theory*, 56(9):4631–4642, 2010. | Cit. on pp. 52, 116, 123, 151, 163.

[449] M. K. Reiter and A. D. Rubin. Anonymous web transactions with Crowds. *Communications of the ACM*, 42(2):32–38, 1999. | Cit. on p. 114.

[450] K. Renaud, M. Volkamer and A. Renkema-Padmos. Why doesn't Jane protect her privacy? In *Proceedings of Privacy enhancing technologies symposium (PETS)*, pp. 244–262. Springer, 2014. | Cit. on p. 96.

[451] A. Rial and G. Danezis. Privacy-preserving smart metering. In *Proceedings of Workshop on privacy in the electronic society (WPES)*, pp. 49–60. ACM, 2011. | Cit. on p. 233.

[452] M. R. Rieback, B. Crispo and A. S. Tanenbaum. The evolution of RFID security. *IEEE Pervasive computing*, 5(1):62–69, 2006. | Cit. on p. 46.

[453] R. L. Rivest. Chaffing and winnowing: Confidentiality without encryption. *CryptoBytes – The technical newsletter of RSA Laboratories*, 4(1):12–17, Summer 1998. | Cit. on pp. 49, 90, 177.

[454] M. Rogers and S. Bhatti. How to disappear completely: A survey of private peer-to-peer networks. Research note RN/07/13, Department of Computer Science, University College London, May 2007. | Cit. on p. 172.

[455] S. Romanosky, A. Acquisti, J. Hong, L. F. Cranor and B. Friedman. Privacy patterns for online interactions. In *Proceedings of Pattern languages of programs (PLoP)*, article 12. ACM, 2006. | Cit. on p. 233.

[456] S. Rosenberg. Russia inflates its military with blow-up weapons. In *BBC News* at `http://www.bbc.com/news/world-europe-11511886`, October 2010. Last accessed on 6 October 2019. | Cit. on p. 46.

[457] I. S. Rubinstein, R. D. Lee and P. M. Schwartz. Data mining and Internet profiling: Emerging regulatory and technological approaches. *The University of Chicago law review*, 75(1):261–285, 2008. | Cit. on p. 3.

[458] S. Ruoti, N. Kim, B. Burgon, T. Van Der Horst and K. Seamons. Confused Johnny: When automatic encryption leads to confusion and mistakes. In *Proceedings of Symposium on usable privacy and security (SOUPS)*, article 5. ACM, 2013. | Cit. on pp. 97, 98, 215.

[459] B. A. Safari. Intangible privacy rights: How Europe's GDPR will set a new global standard for personal data protection. *Seton Hall law review (SHLR)*, 47(3):809–848, 2016. | Cit. on pp. 3, 4, 41.

[460] P. Saint-Andre. Extensible messaging and presence protocol (XMPP): Core. RFC 6120, IETF, 2011. | Cit. on p. 206.

[461] F. Saint-Jean, A. Johnson, D. Boneh and J. Feigenbaum. Private web search. In *Proceedings of Workshop on privacy in the electronic society (WPES)*, pp. 84–90. ACM, 2007. | Cit. on p. 114.

[462] M. B. Salem and S. J. Stolfo. Decoy document deployment for effective masquerade attack detection. In *Proceedings of Detection of intrusions and malware, and vulnerability assessment (DIMVA)*, pp. 35–54. Springer, 2011. | Cit. on p. 57.

[463] D. Sánchez, J. Castellà-Roca and A. Viejo. Knowledge-based scheme to create privacy-preserving but semantically-related queries for web search engines. *Information sciences*, 218:17–30, January 2013. | Cit. on p. 165.

[464] R. Sandhu and P. Samarati. Authentication, access control, and audit. *ACM Computing surveys (CSUR)*, 28(1):241–243, 1996. | Cit. on p. 39.

[465] L. Sankar, S. R. Rajagopalan and H. V. Poor. Utility-privacy tradeoffs in databases: An information-theoretic approach. *IEEE Transactions on information forensics and security (TIFS)*, 8(6):838–852, 2013. | Cit. on p. 22.

[466] S. Sannon, N. N. Bazarova and D. Cosley. Privacy lies: Understanding how, when, and why people lie to protect their privacy in multiple online contexts. In *Proceedings of Conference on human factors in computing systems (CHI)*, paper 52. ACM, 2018. | Cit. on p. 5.

[467] M. A. Sasse. Usability and trust in information systems. In *Trust and crime in information societies*, chapter 10, pp. 319–348. Edward Elgar, 2005. | Cit. on pp. 95, 96, 98, 109.

[468] M. A. Sasse. Scaring and bullying people into security won't work. *IEEE Security & privacy*, 13(3):80–83, 2015. | Cit. on p. 98.

[469] M. A. Sasse and I. Flechais. Usable security: Why do we need it? How do we get it? In *Security and usability: Designing secure systems that people can use*, chapter 2, pp. 13–30. O'Reilly, 2005. | Cit. on p. 97.

[470] M. A. Sasse, M. Smith, C. Herley, H. Lipford and K. Vaniea. Debunking security-usability tradeoff myths. *IEEE Security & privacy*, 14(5):33–39, 2016. | Cit. on p. 98.

[471] F. Schaub, A. Marella, P. Kalvani, B. Ur, C. Pan, E. Forney and L. F. Cranor. Watching them watching me: Browser extensions' impact on user privacy awareness and concern. In *Proceedings of Workshop on usable security (USEC)*, paper 17. Internet Society, 2016. | Cit. on pp. 5, 97, 109, 214.

[472] B. Schermer. Risks of profiling and the limits of data protection law. In *Discrimination and privacy in the information society*, chapter 7, pp. 137–152. Springer, 2013. | Cit. on p. 41.

[473] B. Schneier. TrackMeNot. In *Schneier on Security* at `https://www.schneier.com/blog/archives/2006/08/trackmenot_1.html`. Last accessed on 6 October 2019. | Cit. on p. 142.

[474] S. Schrittwieser, S. Katzenbeisser, J. Kinder, G. Merzdovnik and E. Weippl. Protecting software through obfuscation: Can it keep pace with progress in code analysis? *ACM Computing surveys (CSUR)*, 49(1):4, 2016. | Cit. on pp. 17, 18.

[475] S. Schröder, M. Huber, D. Wind and C. Rottermanner. When Signal hits the fan: On the usability and security of state-of-the-art secure mobile messaging. In *Proceedings of European workshop on usable security (EuroUSEC)*, paper 8. Internet Society, 2016. | Cit. on pp. 41, 205, 214.

[476] G. Schryen. Is open source security a myth? *Communications of the ACM*, 54(5):130–140, 2011. | Cit. on p. 109.

[477] M. Schumacher. Security patterns and security standards. In *Proceedings of European conference on pattern languages of programs (EuroPLoP)*, pp. 289–300. Hillside Europe, 2002. | Cit. on p. 220.

[478] J. C. Scott. Everyday forms of resistance. *The Copenhagen journal of Asian studies*, 4(89):33–62, 1989. | Cit. on pp. 6, 244, 245.

[479] B. Scottberg, W. Yurcik and D. Doss. Internet honeypots: Protection or entrapment? In *Proceedings of International symposium on technology and society (ISTAS)*, pp. 387–391. IEEE, 2002. | Cit. on p. 56.

[480] secushare. 15 reasons not to start using PGP. Online at `http://secushare.org/PGP`. Last accessed on 6 October 2019. | Cit. on pp. 5, 205.

[481] J. P. Semitsu. From Facebook to mug shot: How the dearth of social networking privacy rights revolutionized online government surveillance. *Pace law review*, 31(1):291–381, 2011. | Cit. on p. 170.

[482] P. Shankar, V. Ganapathy and L. Iftode. Privately querying location-based services with SybilQuery. In *Proceedings of Ubiquitous computing (UbiComp)*, pp. 31–40. ACM, 2009. | Cit. on p. 51.

[483] C. E. Shannon. Communication theory of secrecy systems. *Bell Labs Technical journal*, 28(4):656–715, 1949. | Cit. on pp. 47, 73.

[484] C. E. Shannon. The bandwagon. *IRE Transactions on information theory*, 2(1):3, 1956. | Cit. on p. 74.

[485] B. Shapira, Y. Elovici, A. Meshiach and T. Kuflik. PRAW - A PRivAcy model for the Web. *Journal of the American society for information science and technology*, 56(2):159–172, 2005. | Cit. on p. 145.

[486] X. Shen, B. Tan and C. Zhai. Privacy protection in personalized search. *ACM SIGIR Forum*, 41(1):4–17, 2007. | Cit. on pp. 114, 166.

[487] V. Shmatikov and M.-H. Wang. Timing analysis in low-latency mix networks: Attacks and defenses. In *Proceedings of European symposium on research in computer security (ESORICS)*, pp. 18–33. Springer, 2006. | Cit. on p. 55.

[488] R. Shokri. Privacy games: Optimal user-centric data obfuscation. *Proceedings on privacy enhancing technologies (PoPETs)*, 2015(2):299–315, 2015. | Cit. on pp. 15, 16, 78, 81, 82.

[489] R. Shokri, M. Stronati, C. Song and V. Shmatikov. Membership inference attacks against machine learning models. In *Proceedings of Symposium on security and privacy (S&P)*, pp. 3–18. IEEE, 2017. | Cit. on p. 33.

[490] R. Shokri, G. Theodorakopoulos, J.-Y. Le Boudec and J.-P. Hubaux. Quantifying location privacy. In *Proceedings of Symposium on security and privacy (S&P)*, pp. 247–262. IEEE, 2011. | Cit. on pp. 22, 78, 80, 81.

[491] N. P. Smart and F. Vercauteren. Fully homomorphic SIMD operations. *Designs, codes and cryptography*, 71(1):57–81, April 2014. | Cit. on p. 42.

[492] G. Smith. On the foundations of quantitative information flow. In *Proceedings of Foundations of software science and computational structures (FoSSaCS)*, pp. 288–302. Springer, 2009. | Cit. on pp. 71, 75.

[493] G. Smith. Quantifying information flow using min-entropy. In *Proceedings of Quantitative evaluation of systems (QEST)*, pp. 159–167. IEEE, 2011. | Cit. on pp. 74, 75, 76.

[494] M. Sobolewski, J. Mazur and M. Paliński. GDPR: A step towards a user-centric Internet? *Intereconomics*, 52(4):207–213, 2017. | Cit. on p. 4.

[495] C. Soghoian. The problem of anonymous vanity searches. *I/S: A journal of law and policy for the information society*, 3(2):299–318, 2007. | Cit. on p. 234.

[496] C. Soghoian. An end to privacy theater: Exposing and discouraging corporate disclosure of user data to the government. *Minnesota journal of law, science & technology*, 12(1):191–237, 2011. | Cit. on p. 203.

[497] D. J. Solove. A taxonomy of privacy. *University of Pennsylvania law review*, 154(3):477–560, 2005. | Cit. on p. 2.

[498] D. J. Solove. *Nothing to hide: The false tradeoff between privacy and security*. Yale University Press, 2011. | Cit. on pp. 2, 3.

[499] D. J. Solove. Privacy self-management and the consent dilemma. *Harvard law review*, 126(7):1880–1903, 2013. | Cit. on p. 3.

[500] S. Spiekermann. The challenges of privacy by design. *Communications of the ACM*, 55(7):38–40, 2012. | Cit. on p. 221.

[501] S. Spiekermann and L. F. Cranor. Engineering privacy. *IEEE Transactions on software engineering*, 35(1):67–82, 2009. | Cit. on pp. 233, 237.

[502] L. Spitzner. Honeypots: Catching the insider threat. In *Proceedings of Annual computer security applications conference (ACSAC)*, pp. 170–179. IEEE, 2003. | Cit. on p. 57.

[503] N. Srnicek. We need to nationalise Google, Facebook and Amazon. Here's why. In *The Guardian* at `https://www.theguardian.com/commentisfree/2017/aug/30/nationalise-google-facebook-amazon-data-monopoly-platform-public-interest`, August 2017. Last accessed on 6 October 2019. | Cit. on pp. 170, 172.

[504] F. Stalder. The failure of privacy enhancing technologies (PETs) and the voiding of privacy. *Sociological research online*, 7(2):1–15, 2002. | Cit. on p. 3.

[505] F.-X. Standaert, T. G. Malkin and M. Yung. A formal practice-oriented model for the analysis of side-channel attacks. Cryptology ePrint archive report 2006/139, IACR, 2006. | Cit. on p. 192.

[506] R. Stedman, K. Yoshida and I. Goldberg. A user study of Off-the-Record Messaging. In *Proceedings of Symposium on usable privacy and security (SOUPS)*, pp. 95–104. ACM, 2008. | Cit. on pp. 41, 205.

[507] S. J. Stolfo, M. B. Salem and A. D. Keromytis. Fog computing: Mitigating insider data theft attacks in the cloud. In *Proceedings of Symposium on security and privacy workshops (SPW)*, pp. 125–128. IEEE, 2012. | Cit. on p. 56.

[508] T. Strufe. Safebook: A privacy-preserving online social network leveraging on real-life trust. *IEEE Communications magazine*, 47(12):94–101, December 2009. | Cit. on p. 4.

[509] N. Summers. Facebook trials tweaked single-column Timeline design and new 'Like Page' button in New Zealand. In *The Next Web* at `https://thenextweb.com/facebook/2013/02/28/facebook-trials-tweaked-single-column-timeline-and-new-like-page-button-on-posted-links/`, February 2013. Last accessed on 6 October 2019. | Cit. on p. 205.

[510] J. Sun, K. Sun and Q. Li. CyberMoat: Camouflaging critical server infrastructures with large scale decoy farms. In *Proceedings of Communications and network security (CNS)*, pp. 1–9. IEEE, 2017. | Cit. on p. 56.

[511] D. Susser, B. Roessler and H. Nissenbaum. Online manipulation: Hidden influences in a digital world. *Georgetown law technology review (forthcoming)*, September 2019. Available at SSRN: `https://ssrn.com/abstract=3306006`. | Cit. on p. 2.

[512] L. Sweeney. k-anonymity: a model for protecting privacy. *International journal of uncertainty, fuzziness and knowledge-based systems*, 10(5):557–570, 2002. | Cit. on p. 22.

[513] L. Sweeney. Discrimination in online ad delivery. *ACM Queue - Storage*, 11(4):1–19, March 2013. | Cit. on p. 2.

[514] B. Tancer. *Click: What millions of people are doing online and why it matters*. Hyperion, 2008. | Cit. on p. 113.

[515] N. J. Taylor, A. R. Dennis and J. W. Cummings. Situation normality and the shape of search: The effects of time delays and information presentation on search behavior. *Journal of the Association for information science and technology*, 64(5):909–928, 2013. | Cit. on p. 32.

[516] J. Teevan, S. T. Dumais and E. Horvitz. Personalizing search via automated analysis of interests and activities. In *Proceedings of the Research and development in information retrieval (SIGIR)*, pp. 449–456. ACM, 2005. | Cit. on p. 166.

[517] O. Tene and J. Polonetsky. To track or do not track: Advancing transparency and individual control in online behavioral advertising. *Minnesota journal of law, science & technology*, 13(1):281–357, 2012. | Cit. on p. 224.

[518] L. C. Thomas. A survey of credit and behavioural scoring: Forecasting financial risk of lending to consumers. *International journal of forecasting*, 16(2):149–172, 2000. | Cit. on p. 103.

[519] R. R. Toledo, G. Danezis and I. Goldberg. Lower-cost ϵ-private information retrieval. *Proceedings on privacy enhancing technologies (PoPETs)*, 2016(4):184–201, 2016. | Cit. on pp. 4, 115.

[520] Tor Browser user manual. In *The Tor Project* at `https://tb-manual.torproject.org/`. Last accessed on 5 October 2019. | Cit. on pp. 4, 99.

[521] Tor FAQ. In *The Tor Project* at `https://2019.www.torproject.org/docs/faq.html.en`. Last accessed on 6 October 2019. | Cit. on p. 225.

[522] V. Toubiana, A. Narayanan, D. Boneh, H. Nissenbaum and S. Barocas. Adnostic: Privacy preserving targeted advertising. In *Proceedings of The network and distributed systems symposium (NDSS)*, pp. 1–16, 2010. | Cit. on p. 244.

[523] V. Toubiana, L. Subramanian and H. Nissenbaum. TrackMeNot: Enhancing the privacy of web search. *Computing research repository (CoRR)*, arXiv:1109.4677, 2011. | Cit. on pp. 5, 142, 144, 145.

[524] F. Tramèr, F. Zhang, A. Juels, M. K. Reiter and T. Ristenpart. Stealing machine learning models via prediction apis. In *Proceedings of USENIX Security symposium*, pp. 601–618. USENIX, 2016. | Cit. on p. 45.

[525] C. Troncoso, C. Diaz, O. Dunkelman and B. Preneel. Traffic analysis attacks on a continuously-observable steganographic file system. In *Proceedings of Information hiding (IH)*, pp. 220–236. Springer, 2007. | Cit. on p. 48.

[526] S. Vadhan. The complexity of differential privacy. In *Tutorials on the foundations of cryptography*, chapter 7, pp. 347–450. Springer, 2017. | Cit. on p. 68.

[527] M. Van Dijk, C. Gentry, S. Halevi and V. Vaikuntanathan. Fully homomorphic encryption over the integers. In *Proceedings of Advances in cryptology (EUROCRYPT)*, pp. 24–43. Springer, 2010. | Cit. on p. 39.

[528] J. van Rest, D. Boonstra, M. Everts, M. van Rijn and R. van Paassen. Designing privacy-by-design. In *Proceedings of Annual privacy forum (APF) 2012: Privacy technologies and policy*, pp. 55–72. Springer, 2014. | Cit. on p. 222.

[529] S. J. Vaughan-Nichols. The great instant-messaging foul-up. In *ZDNet* at `https://www.zdnet.com/article/the-great-instant-messaging-foul-up/`, March 2017. Last accessed on 6 October 2019. | Cit. on p. 206.

[530] E. Vaziripour, M. O'Neill, J. Wu, S. Heidbrink, K. Seamons and D. Zappala. Social authentication for end-to-end encryption. In *Proceedings of Symposium on usable privacy and security (SOUPS)*, 2016. | Cit. on p. 201.

[531] K. Vemou and M. Karyda. A classification of factors influencing low adoption of PETs among SNS users. In *Proceedings of Trust, privacy and security in digital business (TrustBus)*, pp. 74–84. Springer, 2013. | Cit. on pp. 201, 202.

[532] A. Viejo and J. Castellà-Roca. Using social networks to distort users' profiles generated by web search engines. *Computer networks*, 54(9):1343–1357, 2010. | Cit. on pp. 115, 165.

[533] B. Viswanath, A. Mislove, M. Cha and K. P. Gummadi. On the evolution of user interaction in Facebook. In *Proceedings of Workshop on online social networks (WOSN)*, pp. 37–42. ACM, 2009. | Cit. on p. 182.

[534] L. Von Ahn, M. Blum and J. Langford. Telling humans and computers apart automatically. *Communications of the ACM*, 47(2):56–60, 2004. | Cit. on p. 106.

[535] G. von Krogh and S. Spaeth. The open source software phenomenon: Characteristics that promote research. *The journal of strategic information systems*, 16(3):236–253, 2007. | Cit. on p. 109.

[536] V. L. Voydock and S. T. Kent. Security mechanisms in high-level network protocols. *ACM Computing surveys (CSUR)*, 15(2):135–171, 1983. | Cit. on pp. 47, 172.

[537] R. Walker Reczek, C. A. Summers and R. W. Smith. Online ads know who you are, but can they change you too? In *The Conversation* at `https://theconversation.com/online-ads-know-who-you-are-but-can-they-change-you-too-54983`, March 2016. Last accessed on 6 October 2019. | Cit. on p. 44.

[538] S. W. Waller. Antitrust and social networking. *North Carolina law review*, 90(5):1771–1806, June 2012. | Cit. on p. 170.

[539] D. Wang, D. Pedreschi, C. Song, F. Giannotti and A.-L. Barabasi. Human mobility, social ties, and link prediction. In *Proceedings of*

Knowledge discovery and data mining (KDD), pp. 1100–1108. ACM, 2011. | Cit. on p. 193.

[540] G. Wang, T. Konolige, C. Wilson, X. Wang, H. Zheng and B. Y. Zhao. You are how you click: Clickstream analysis for Sybil detection. In *Proceedings of USENIX Security symposium*, pp. 241–255. USENIX, 2013. | Cit. on p. 217.

[541] J. Wang, W. Zhang and S. Yuan. Display advertising with real-time bidding (RTB) and behavioural targeting. *Foundations and trends in information retrieval*, 11(4-5):297–435, 2017. | Cit. on p. 105.

[542] P. Wang and C. V. Ravishankar. On masking topical intent in keyword search. In *Proceedings of International conference on data engineering (ICDE)*, pp. 256–267. IEEE, 2014. | Cit. on p. 158.

[543] Q. Wang, X. Gong, G. T. Nguyen, A. Houmansadr and N. Borisov. CensorSpoofer: Asymmetric communication using IP spoofing for censorship-resistant web browsing. In *Proceedings of Computer and communications security (CCS)*, pp. 121–132. ACM, 2012. | Cit. on p. 54.

[544] S. Wang, D. Agrawal and A. E. Abbadi. Towards practical private processing of database queries over public data with homomorphic encryption. Report 2011-06, Department of computer science, University of California, Santa Barbara, 2011. | Cit. on p. 36.

[545] T. Wang, X. Cai, R. Nithyanand, R. Johnson and I. Goldberg. Effective attacks and provable defenses for website fingerprinting. In *Proceedings of USENIX Security symposium*, pp. 143–157. USENIX, 2014. | Cit. on pp. 87, 90, 93, 94, 241.

[546] T. Wang and I. Goldberg. Walkie-talkie: An efficient defense against passive website fingerprinting attacks. In *Proceedings of USENIX Security symposium*, pp. 1375–1390. USENIX, 2017. | Cit. on pp. 48, 55, 94.

[547] Y. Wang and A. Kobsa. Privacy-enhancing technologies. In *Handbook of research on social and organizational liabilities in information security*, chapter 13, pp. 203–227. IGI Global, 2009. | Cit. on p. 15.

[548] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang and H. Qi. Beyond inferring class representatives: User-level privacy leakage from federated learning. In *Proceedings of International conference on computer communications (INFOCOM)*, pp. 2512–2520. IEEE, 2019. | Cit. on p. 244.

[549] Y. Watanabe, J. Shikata and H. Imai. Equivalence between semantic security and indistinguishability against chosen ciphertext attacks. In *Proceedings of Public key cryptography (PKC)*, pp. 71–84. Springer, 2003. | Cit. on p. 66.

[550] Z. Weinberg, J. Wang, V. Yegneswaran, L. Briesemeister, S. Cheung, F. Wang and D. Boneh. StegoTorus: A camouflage proxy for the Tor anonymity system. In *Proceedings of Computer and communications security (CCS)*, pp. 109–120. ACM, 2012. | Cit. on pp. 54, 55.

[551] A. Whitten and J. D. Tygar. Why Johnny can't encrypt: A usability evaluation of PGP 5.0. In *Proceedings of USENIX Security symposium*, pp. 169–183. USENIX, 1999. | Cit. on pp. 5, 41, 205.

[552] B. Wiley. Dust: A blocking-resistant Internet transport protocol. Technical report, School of information, University of Texas at Austin, 2011. | Cit. on p. 54.

[553] L. E. Willis. Why not privacy by default? *Berkeley technology law journal*, 29(1):61–134, 2014. | Cit. on p. 100.

[554] C. Wilson, B. Boe, A. Sala, K. P. Puttaswamy and B. Y. Zhao. User interactions in social networks and their implications. In *Proceedings of European conference on computer systems (EuroSys)*, pp. 205–218. ACM, 2009. | Cit. on p. 182.

[555] R. Wishart, K. Henricksen and J. Indulska. Context obfuscation for privacy via ontological descriptions. In *Proceedings of Simposium on Location- and Context-Awareness (LoCA)*, pp. 276–288. Springer, 2005. | Cit. on p. 16.

[556] J. C. Wong. The Cambridge Analytica scandal changed the world - but it didn't change Facebook. In *The Guardian* at `https://www.theguardian.com/technology/2019/mar/17/the-cambridge-analytica-scandal-changed-the-world-but-it-didnt-change-facebook`, March 2019. Last accessed on 7 October 2019. | Cit. on pp. 44, 201, 221.

[557] J. C. Wong and O. Solon. Google to shut down Google+ after failing to disclose user data leak. In *The Guardian* at `https://www.theguardian.com/technology/2018/oct/08/google-plus-security-breach-wall-street-journal`, October 2018. Last accessed on 7 October 2019. | Cit. on p. 201.

[558] K. Wuyts, R. Scandariato, B. De Decker and W. Joosen. Linking privacy solutions to developer goals. In *Proceedings of Availability, reliability and security (ARES)*, pp. 847–852. IEEE, 2009. | Cit. on p. 222.

[559] K. Xu, T. Cao, S. Shah, C. Maung and H. Schweitzer. Cleaning the null space: A privacy mechanism for predictors. In *Proceedings of AAAI conference on artificial intelligence*, pp. 2789–2795. AAAI, 2017. | Cit. on p. 45.

[560] A. C. Yao. Protocols for secure computations. In *Proceedings of Symposium on foundations of computer science (SFCS)*, pp. 160–164. IEEE, 1982. | Cit. on p. 39.

[561] A. C. Yao. Theory and application of trapdoor functions. In *Symposium on foundations of computer science (SFCS)*, pp. 80–91. IEEE, 1982. | Cit. on p. 73.

[562] C. Yao, L. Wang, S. X. Wang and S. Jajodia. Indistinguishability: The other aspect of privacy. In *Proceedings of Workshop on Secure data management (SDM)*, pp. 1–17. Springer, 2006. | Cit. on p. 67.

[563] S. Ye, S. F. Wu, R. Pandey and H. Chen. Noise injection for search privacy protection. In *Proceedings of Computational science and engineering (CSE)*, pp. 1–8. IEEE, 2009. | Cit. on pp. 52, 163.

[564] S. Yekhanin. Private information retrieval. *Communications of the ACM*, 53(4):68–73, 2010. | Cit. on p. 39.

[565] T.-H. You, W.-C. Peng and W.-C. Lee. Protecting moving trajectories with dummies. In *Proceedings of Mobile data management (MDM)*, pp. 278–282. IEEE, 2007. | Cit. on pp. 36, 51.

[566] F. Yu, Y. Xie and Q. Ke. SBotMiner: Large scale search bot detection. In *Proceedings of Conference on Web search and data mining (WSDM)*, pp. 421–430. ACM, 2010. | Cit. on p. 144.

[567] H. Yu, P. B. Gibbons, M. Kaminsky and F. Xiao. SybilLimit: A near-optimal social network defense against Sybil attacks. In *Proceedings of Symposium on security and privacy (S&P)*, pp. 3–17. IEEE, 2008. | Cit. on p. 217.

[568] S. Yuan, A. Z. Abidin, M. Sloan and J. Wang. Internet advertising: An interplay among advertisers, online publishers, ad exchanges and web users. *Computing research repository (CoRR)*, arXiv:1206.1754, 2012. | Cit. on p. 105.

[569] J. Yuill, M. Zappe, D. Denning and F. Feer. Honeyfiles: Deceptive files for intrusion detection. In *Proceedings of Information assurance workshop (IAW)*, pp. 116–122. IEEE, 2004. | Cit. on p. 57.

[570] B. Zantout and R. Haraty. I2P data communication system. In *Proceedings of International conference on networks (ICN)*, pp. 401–409. IARIA, 2011. | Cit. on p. 43.

[571] T. Z. Zarsky. Incompatible: The GDPR in the age of big data. *Seton Hall law review*, 47(4):995–1020, 2017. | Cit. on p. 4.

[572] F. Zhao, Y. Hori and K. Sakurai. Analysis of privacy disclosure in DNS query. In *Proceedings of Multimedia and ubiquitous engineering (MUE)*, pp. 952–957. IEEE, 2007. | Cit. on pp. 52, 101.

[573] F. Zhao, Y. Hori and K. Sakurai. Two-servers PIR based DNS query scheme with privacy-preserving. In *Proceedings of Intelligent pervasive computing (IPC)*, pp. 299–302. IEEE, 2007. | Cit. on p. 52.

[574] E. Zhong, B. Tan, K. Mo and Q. Yang. User demographics prediction based on mobile data. *Pervasive and mobile computing*, 9(6):823–837, 2013. | Cit. on p. 171.

[575] M. Zhong. A faster single-term divisible electronic cash: ZCash. *Electronic commerce research and applications*, 1(3-4):331–338, 2002. | Cit. on p. 42.

[576] X. Zhou, H. Pang and K.-L. Tan. Hiding data accesses in steganographic file system. In *Proceedings of International conference on data engineering (ICDE)*, pp. 572–583. IEEE, 2004. | Cit. on p. 48.

[577] K. Zhu, K. L. Kraemer, V. Gurbaxani and S. Xu. Migration to open-standard interorganizational systems: Network effects, switching costs and path dependency. *MIS Quarterly – Special issue on standard making*, 30:515–539, August 2005. | Cit. on p. 172.

[578] Y. Zhu and R. Bettati. Anonymity vs. information leakage in anonymity systems. In *Distributed Computing Systems, 2005. ICDCS 2005. Proceedings. 25th IEEE International Conference on*, pp. 514–524. IEEE, 2005. | Cit. on p. 73.

[579] M. Zimmer. The externalities of search 2.0: The emerging privacy threats when the drive for the perfect search engine meets web 2.0. *First Monday*, 13(3), March 2008. | Cit. on p. 2.

[580] S. Zuboff. Big other: Surveillance capitalism and the prospects of an information civilization. *Journal of information technology*, 30(1):75–89, 2015. | Cit. on p. 3.

[581] S. Zuboff. *The age of surveillance capitalism: The fight for the future at the new frontier of power*. Profile Books, 2019. | Cit. on pp. 2, 3, 4, 243.

[582] M. Zuckerberg. A privacy-focused vision for social networking. Online at `https://www.facebook.com/notes/mark-zuckerberg/a-privacy-focused-vision-for-social-networking/10156700570096634/`, March 2019. Last accessed on 7 October 2019. | Cit. on pp. 170, 201.

[583] F. J. Zuiderveen Borgesius, S. Kruikemeier, S. C. Boerman and N. Helberger. Tracking walls, take-it-or-leave-it choices, the GDPR, and the ePrivacy regulation. *European data protection law review*, 3(3):353–368, 2017. | Cit. on p. 4.

[584] F. J. Zuiderveen Borgesius and J. Poort. Online price discrimination and EU data privacy law. *Journal of consumer policy*, 40(3):347–366, 2017. | Cit. on p. 2.

Appendix A

Scramble! user study documentation

This appendix includes the following documentation to our Scramble! user study: an excerpt of the entry questionnaire that we asked user participants to fill in upon the start of the session, the introduction to Facebook's privacy settings' limitations that we asked user participants to read as an introduction to the guided tour to Scramble! and, lastly, the exit questionnaire that we asked user participants to fill in at the end of the session.

A.1 Entry questionnaire (excerpt)

We denote single choice questions as [SC], multiple choice questions as [MC] and open questionas as [OQ].

Q7. Who should make the following decisions?
(You may select multiple choices, including "Others". For example, you may check both "You" and "Facebook" and add "My friends" to "Others")

"Who should decide...

- who is able to see the photos you are tagged in?"
- which data in your account the police can access in case of an investigation?"

- who is able to see your personal details (age, phone number, hometown, etc)?"
- who is able to see what you post on the site?"
- when you can delete your account?"
- what kind of information (i.e., news, links to youtube videos, etc) you post?"
- who is able to see your list of friends?"
- who is able to know whose profiles' you have visited?"
- the configuration of your privacy settings?"
- the number of private messages you can keep in your Facebook mailbox?"
- where your data goes if Facebook stops its service?"
- who is able to read the private messages you send?"
- decide which ads show up on Facebook?"
- who will have access to your data if Facebook stops its service"
- what information is used from your profile to deliver ads to you?"
- who can send you a friend request?"
- who is able to see and access everything you do/see/post on Facebook?"
- who is able to know which photos (of your friends or other Facebook users) you have seen?"
- for how long your data is kept available on Facebook?"
- who can send you private messages?"

You / Facebook / Others (who?) / Don't know

Q8. Who should be responsible for the following? [MC]
(Regardless of whether or not that is the actual situation on Facebook).
"Who should be responsible for...

- posting photos or messages you may later regret your friends had seen/read?"
- keeping your password confidential?"
- making sure your privacy settings work?"
- making sure that photos you do not like are deleted and not available on the site anymore?"
- the grammar of your posts and messages?"

- preventing strangers from logging in to your account?"
- what your friends can see in your profile (this is, which of your photos, posts, etc., they can see?"
- making sure your friends can see your profile at any time of the day?"
- making sure strangers are not able to see your posts and photos?"
- setting the proper privacy settings on your profile?"
- making sure private companies do not have access to the data you post to the site without your permission?"
- making sure your friends do not post photos of you that you do not like?"
- preventing people other than your friends from reading your messages and seeing your photos?"
- allowing your friends to post comments to your status updates and photos?"
- friends being upset because of what you post?"

You / Facebook / Others (who?) / Don't know

Q11. On Facebook, what do you feel responsible for with respect to your own privacy? [OQ]

Q13. Which privacy problems, if any, do you encounter using Facebook? [OQ]

Q17. What, if anything, would you add to, modify or delete from the Facebook privacy settings? [OQ]

Q18. What privacy issues you have, if any, that you are not able to solve with Facebook's privacy settings?
None / The following: [OQ] */ Don't know*

- (If *The following:...*) Which strategies do you use to solve those privacy issues?
  None (⇒ Why none?)/ The following: [OQ]

Q20. Are you aware of any strategies or mechanisms, currently not provided by Facebook, that can help you better protect your privacy? [SC]
Yes (⇒Which ones?) */ No*

Q24. When sharing information online, such as sending a message, posting a link to a website, etc., the term intended recipients refers all the people you would like to be able to have access to that message or piece of

information. When you send a message or post something on Facebook, do you think anyone other than the intended recipients is able to access it?
Yes / No / Don't know

- (If *Yes*) Who?

Q25. Regardless of whether or not they are able to, who, if anybody, do you think wants to have access to your data on Facebook?
Nobody / The following: [OQ] */ Don't know.*

Q26. Are you interested in tools or technologies that would prevent unintended recipients from having access to your Facebook data? [SC]
Yes / No / Don't know

Q27. Which strategies or mechanisms do you know, even if you do not use them, to prevent unintended recipients from having access to your messages and information you send or post on Facebook? *None / The following:* [OQ]

- (If any) Which ones do you use? *None / The following:*
  - (If *None*) Why would you, or would you not, use such a tool? [SC] *I would install one of those tools because...* [OQ] */ I would* ***not*** *install one of those tools because...* [OQ]

A.2 Introduction to limitation of privacy settings

Below we reproduce in its entirety the text we asked participants to read as a preparation to the guided tour to *Scramble!*.

A brief introduction to the limitations of Facebook's Privacy Settings.

As a Facebook user, you can use Facebook's privacy settings to control certain aspects of how you share your stuff on Facebook. Currently, Facebook's privacy settings allow you to control, among other things, who can see your posts. For instance, you can choose to make your posts public (this is, visible to all Facebook users), visible only to your Facebook friends or visible only to you. You can also use Facebook's friend lists to select a certain group of friends amongst your full list of friends.

Facebook's privacy settings have however certain limitations, such as the following: Facebook's privacy settings do not prevent Facebook itself from having access to all the information (private messages, photos, posts, etc).

Facebook's privacy settings may afford little protection against anybody breaking into Facebook. For example, if Facebook suffers from a security breach and an intruder is able to access all the information of all the users on Facebook, the posts of the people that carefully set their privacy settings will be as available to this intruder as the posts of the people that had posted all their information as "Public". Facebook's privacy settings enforcement solely relies on Facebook. Users may change their privacy settings, but it is Facebook who enforces the settings and any changes thereafter. This means that users have to rely on Facebook to properly enforce those settings, and to do no mistakes by revealing information to people who are not supposed to have access to it. On the other hand, nothing stops Facebook from changing these settings at any time so that information becomes more or less available to a wider or smaller audience. Users have no control over how Facebook manages their information beyond setting their preferences through the privacy settings. Facebook's privacy settings do not protect against Facebook revealing information to third parties. For instance, it has been revealed that so far in 2013 government agencies have demanded access to the information of over 38 000 *Facebook users. Facebook's privacy settings cannot prevent Facebook users against Facebook revealing information to governments or any other third parties, such as private companies.*

All in all, Facebook's privacy settings suffer from the fact that the user has no control over them beyond signalling a preference. It is Facebook who has ultimate control over these settings and has the power to enforce them.

A.3 Exit questionnaire

In addition to the SUS questionnaire (See [96]):

Q36. Can you describe, in a few words, your experience using *Scramble!*? [OQ]

Q37. What would be the advantages, if any, of using a tool like *Scramble!* over, or in combination with, other privacy controls? [OQ]

Q38. *Scramble!* encrypts messages before you send or post them on Facebook. Do you think this is a secure way to prevent unintended recipients from having access to them? [OQ]

Q39. What do you think are the differences, if any, between what *Scramble!* does and the privacy settings of Facebook?

Q40. How would you grade the following tools: Facebook's privacy settings / *Scramble!* with respect to

- safety?
- reliability?
- trustworthiness?

Q41. Overall, explain in a few words why you find a tool like *Scramble!* to be useful or not useful. [SC]+ [OQ]
I find tools like Scramble! useful because... / I find tools like Scramble! ***not*** *useful because...*

Q42. Do you see yourself using a tool such as *Scramble!*? [SC]
Never / Rarely / Sometimes / Often / All of the Time / Don't know

Q43. Which cases, purposes or people do you think a tool like Scramble! could be useful for? [OQ]

Q44. What, if anything, did you like about *Scramble!*? (You will be asked about what you did not like in the question below) [OQ]

Q45. What, if anything, did you dislike about *Scramble!*? [OQ]

Q46. What features did you miss in *Scramble!* or you think that such a tool should have? [OQ]

Curriculum vitae

Ero Balsa was born in Ferrol, Spain in 1986. He completed a *enxenharia técnica de telecomunicações* (B.S. in telecommunication engineering) in 2007 and a *enxenharia de telecomunicações* (master's degree in telecommunications engineering) in 2010, both at the University of Vigo, Spain. He completed his master's thesis at KU Leuven in 2010 as an Erasmus student under the supervision of Prof. Claudia Diaz and Prof. Carmela Troncoso. His master's thesis focused on the design and evaluation of chaff-based obfuscation strategies to achieve communication profile confidentiality in social networking sites.

He joined KU Leuven's research group COSIC as a doctoral student under the supervision of Prof. Claudia Diaz in December 2010. During his studies he visited Prof. Zhiguo Wan at Tsinghua University, China, in June 2013 and Prof. Alessandro Acquisti at Carnegie Mellon University, Pennsylvania, USA, in August–September, 2013.

List of publications

International journals

E. Balsa, C. Pérez-Solà and C. Diaz. Towards inferring communication patterns in online social networks. *ACM Transactions on internet technology (TOIT)*, 17(3), 2017.

E. Balsa, C. Troncoso and C. Diaz. A metric to evaluate interaction obfuscation in online social networks. *International journal of uncertainty, fuzziness and knowledge-based systems*, 20(06):877–892, 2012.

International conferences and workshops

E. Balsa, F. Beato, and S. Gürses. Why can't online social networks encrypt? In *Workshop on privacy and user-centric controls*. W3C, 2014.

E. Balsa, L. Brandimarte, A. Acquisti, C. Diaz and S. Gürses. Spiny CACTOS: OSN users' attitudes and perceptions towards cryptographic access control tools. In *Workshop on Usable Security (USEC)*. Internet Society, 2014.

E. Balsa, C. Troncoso and C. Diaz. OB-PWS: Obfuscation-based private web search. In *IEEE Symposium on Security and Privacy (S&P)*, pages 491–505, 2012.

S. Gürses, R. Overdorf and E. Balsa. POTs: the revolution will not be optimized? Paper presented at *Hot topics in privacy enhancing technologies (HotPETs)*, 2018.

FACULTY OF ENGINEERING SCIENCE
DEPARTMENT OF ELECTRICAL ENGINEERING (ESAT)
COSIC
Kasteelpark Arenberg 10, bus 2452
3001 Heverlee
ero.balsa@esat.kuleuven.be