CyBOK

PRIVACY & ONLINE RIGHTS KNOWLEDGE AREA

(DRAFT FOR COMMENT)

AUTHOR: Carmela Troncoso – École Polytechnique Fédérale de Lausanne

EDITOR: George Danezis – University College London

REVIEWERS:

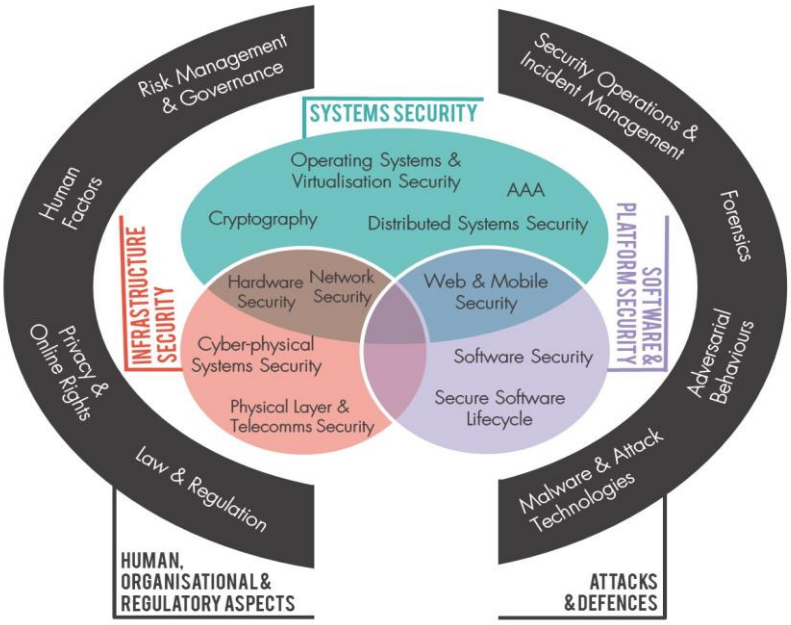
Emiliano De Cristofaro – University College London

Ian Goldberg – University of Waterloo

Theresa Stadler – Privitar

© Crown Copyright, The National Cyber Security Centre 2019.

Following wide community consultation with both academia and industry, 19 Knowledge Areas (KAs) have been identified to form the scope of the CyBOK (see diagram below). The Scope document provides an overview of these top-level KAs and the sub-topics that should be covered under each and can be found on the project website: https://www.cybok.org/.



We are seeking comments within the scope of the individual KA; readers should note that important related subjects such as risk or human factors have their own knowledge areas.

It should be noted that a fully-collated CyBOK document which includes issue 1.0 of all 19 Knowledge Areas is anticipated to be released by the end of July 2019. This will likely include updated page layout and formatting of the individual Knowledge Areas.

Privacy and Online Rights

Carmela Troncoso

April 2019

INTRODUCTION

The pervasiveness of data collection, processing, and dissemination raises severe privacy concerns regarding individual and societal harms. Information leaks may cause physical or psychological damage to individuals, e.g., when published information can be used by thieves to infer when users are not home, by enemies to find out weak points to launch attacks on users or by advertising companies to build profiles and influence users. On a large scale, the use of this information can be used to influence society as a whole, causing irreversible harm to democracy. The extent of the harms that privacy loss causes highlights that privacy cannot simply be tackled as a confidentiality issue. Beyond keeping information private, it is important to ensure that the systems we build support freedom of speech and individuals' autonomy of decision and self-determination.

The goal of this knowledge area is to introduce system designers to the concepts and technologies that are used to engineer systems that inherently protect users' privacy. We aim to provide designers with the ability to identify privacy problems, to describe them from a technical perspective, and to select adequate technologies to eliminate, or at least, mitigate these problems.

Privacy is recognised as a fundamental human right [1]. As such, it has been studied for many years from a socio-legal perspective with two goals. First, to better understand what privacy means for society and individuals. Second, to ensure that the legal frameworks that underpin our democracies support privacy as a right. The former studies proposed definitions such as privacy being 'the right to be let alone' [2], 'the right to informational self-determination' [3, 4] or 'the freedom from unreasonable constraints on the construction of one's own identity' [5]. Probably one of the best examples of the latter are the principles and rules associated with the European Data Protection Legislation [6] covered in the Law and Regulation KA. All of these conceptualisations are of great importance to define and understand the boundaries of privacy and its role for society. However, their abstract and context-free nature often makes them not actionable for system designers who need to select technologies to ensure that privacy is supported in their systems.

To address this gap, in this knowledge area, we conceptualise privacy in a similar way as security engineering conceptualises security problems [7, 8]. We consider that privacy concerns, and the solutions that can address them, are defined by the adversarial model considered by the designer, the nature of the information to be protected and the nature of the protection itself. Typical examples of adversarial models can be: third-party services with whom data are shared are not trusted, the service provider itself is not trusted with private data of the users, or users of a service should not learn private data from other users. Typical examples of private data to be protected from these adversaries can be: the content of users' communications, their service usage patterns, or the mere existence of users and/or their actions. Finally, typical examples of protection means can be techniques that enable information availability to be controlled, such as access control settings, or techniques to hide information, such as encryption.

This knowledge area is structured as follows. The first part, comprising three sections, considers three different privacy paradigms that have given rise to different classes of privacy technologies. The

first is privacy as confidentiality (Section 1), in which the privacy goal is to *hide* information from the adversary. We revise technological approaches to hide both data and metadata, and approaches to hinder the adversary's ability to perform inferences using the data that cannot be hidden. The second is privacy as informational control (Section 2), in which the goal is to provide users with the means to *decide* what information they will expose to the adversary. We revise technologies that support users in their privacy-oriented decisions and techniques that help them express their preferences when interacting with digital services. Finally, we introduce privacy as transparency (Section 3), in which the goal is to *inform* the user about what data she has exposed and who has accessed or processed these data. We revise solutions that show users their digital footprint, and solutions that support accountability through secure logging.

The second part of the knowledge area is devoted to illustrating how privacy technologies can be used to support democracy and civil liberties (Section 4). We consider two core examples: systems for secure voting and to circumvent censorship. For the former, privacy of the votes is imperative for the functionality itself. For the latter, privacy of communication partners is necessary to ensure that content cannot be blocked by a censor.

We conclude the knowledge area by outlining the steps involved in the engineering of privacy-preserving systems (5). We provide guidelines for engineers to make informed choices about architectural and privacy technologies. These guidelines can help system designers to build systems in which the users' privacy does not depend on a centralised entity that may become a single point of failure.

We note that many of the privacy technologies we revise in this knowledge area rely on the cryptographic concepts introduced in the Cryptography knowledge area [9]. Throughout this knowledge area, we assume that the reader is familiar with these basic concepts and avoid repeating cryptographic definitions and reiterating on the explanation of common primitives.

CONTENT

1 Privacy as Confidentiality

[10][11][12][13][14][15][16][17]

In a technical re-interpretation of the 'right to be let alone' [2] privacy definition, a common conceptualisation of privacy is to avoid making personal information accessible to any entity, in particular to a wider public [7]. Under this definition, the objective of privacy technologies is to enable the use of services while minimising the amount of exposed information. Here, information refers to both data exchanged explicitly with the service, as well as information made implicitly available in the metadata associated with these exchanges (e.g., identity of the users or frequency of usage).

1.1 Data Confidentiality

We now describe two approaches to minimise the amount of exposed information. We first present methods that provably prevent unauthorised access to information, typically based on the use of advanced cryptographic primitives to ensure that no data can be inferred. Second, we present disclosure control methods, which relax the confidentiality definition to ensure that the information leaked to the adversary is limited to a certain amount, or is not linkable to an individual person.

1.1.1 Cryptography-based access control

A first flavour of confidentiality-oriented privacy technologies focus on protecting the *data* through the use of cryptography. These technologies mainly consider two adversary models: one where the

recipient is considered trusted and the data have to be protected while in transit, and one in which the recipient is not trusted and the data must be kept private even when it is processed.

Protecting data in transit. The protection of data in transit is typically known as *end-to-end encryption (E2EE)*. Here, an end refers to the origin and destination of the communication. For instance, the sender and receiver of an email, or the client and server of a service. E2EE ensures that the confidentiality of data is ensured between both ends. That is, no third party, from the routers in the communication infrastructure, to the application (e.g., email, messaging) servers that enable the communication, can access the communication. Additionally, E2EE typically provides integrity, impeding any intermediary from modifying the data exchanged, and authentication, ensuring that the communication parties can be sure of each others' identity.

From a technical perspective, in E2EE the devices at the end of the communication hold the encryption key used to protect the data. Usually, these are symmetric encryption keys and can be agreed using key transport, or can be established using any modality of the Diffie-Hellman exchange. The use of Diffie-Hellman to agree one key per session additionally provides forward secrecy, but one must be careful when implementing the exchange [18]. Typically, Digital Signatures and Message Authentication Codes are used to provide integrity and authentication. Canonical examples of E2EE encryption are the TLS protocol [19], widely used in client-server scenarios; or the PGP protocol, a common encryption mechanism for email communications [20].

A special type of E2EE is Off-the-Record Messaging (OTR) [11].[1] OTR seeks to provide stronger privacy properties than the above protocols. It considers an adversary that can not only observe the communication, but also eventually compromise one of the devices participating in the communication. This compromise gives the adversary the chance to get the long-term keys of the participants. In such a demanding scenario the two main goals of OTR are to provide i) perfect forward secrecy and ii) repudiable authentication, which permits a user to deny having sent a message in the past. The protocol derives the cryptographic keys used for the conversation using an *unauthenticated* Diffie-Hellman key exchange. Then, the participants carry out a mutual authentication inside the protected channel, which guarantees the future repudiability. Encryption keys are rotated, and old keys are deleted, so as to maintain forward secrecy. Current OTR protocols also include strong protection against man-in-the-middle attacks, even if the participants do not pre-share secrets [21, 22].

Finally, we can remark that E2EE is nowadays prevalent in instant messaging applications such as Signal, WhatsApp, Facebook Messenger, or Viber. All of these applications are based on the so-called Signal Protocol (previously known as Axolotl or TextSecure) [23]. Similar to OTR, this protocol provides authenticated messaging between users with end-to-end confidentiality, and messages are kept secret even if the messaging server is compromised, and even if the user's long-term keys are compromised. These properties rely on an authenticated key exchange protocol that mixes multiple Diffie-Hellman shared secrets, and on a protocol to refresh the keys called double ratcheting [15]. Cohn-Gordon et al. provided in [24] a detailed description of this protocol, including a formal analysis.

Note that all of the above protocols only offer strong guarantees as long as the mechanisms to authenticate the communication parties work as expected. For instance, the confidentiality provided by TLS relies on services keeping their keys secret and the Public Key Infrastructure operating reliably, so that the communication parties can be authenticated. Similarly, WhatsApp's confidentiality relies on the fact that phone numbers are hard to spoof and, thus, users are sure that the recipient of their message is their intended interlocutor.

Protection of data during processing. The previous protocols focus on protecting data in transit from third parties other than the communication participants. We now consider situations in which the recipient needs to perform some computation on the data, even though she is considered adversarial.

[1] `https://otr.cypherpunks.ca/`

We distinguish two scenarios: one in which computation is completely outsourced and one in which the sender participates in the computation.

In the first scenario, commonly known as *outsourcing*, the data belong to the sender and the recipient acts as the data processor. Typical examples are the use of cloud services to compute on big data, e.g., privacy-preserving training and classification using machine learning [25], or to hold a database in which the sender wants to perform searches [26]. The solutions to this problem are based on advanced cryptographic protocols. We now illustrate the use of these protocols in a couple of examples, and we refer the reader to the Cryptography knowledge area [9] for more details on the technical details of the underlying primitives.

A common problem when outsourcing services is that accessing particular pieces of outsourced data may reveal information about the user to the entity holding the data. For instance, accessing a given entry on a patent database reveals business intentions; and accessing a particular entry in a messaging directory reveals relationships between users. This problem can be mitigated by using Private Information Retrieval (see Section 9.2 in the Cryptography knowledge area [9]), which allows a database to be queried without revealing whick record is being accessed. An example use case for information retrieval is the creation of privacy-preserving directories for social networks [27, 28, 29].

Another example where remote processing is needed comprises digital shops or digital banking, where a server returns information to a user depending on the inputs. The shop needs to process payments and then ship the digital item; and the bank provides money, or makes a payment, upon authentication. Users' shopping patterns, however, may reveal a lot about their profiles. In this case, Oblivious Transfer (see Section 9.1 in the Cryptography knowledge area [9]), in which a service can transfer an item without knowing which item is being transferred, can be used to support privacy-preserving interactions [30, 31].

The previous techniques are useful for particular operations: search an item on a database, transfer that item. Ideally, we would like to be able to perform any operation on outsourced data. A very relevant technology for this is homomorphic encryption (see Section 11 in the Cryptography knowledge area [9]). This type of encryption allows any operation on encrypted data to be performed. Such flexibility, however, comes at a high cost in terms of computation time, and for some implementations also in terms of bandwidth, thus making it far from practical at this point. Less general versions such as somewhat homomorphic encryption or partially homomorphic encryption, which only permit limited operations (sums, multiplications or evaluating a given function) provide better trade-offs and can already be used for simple concrete tasks.

We note that in recent years, the privacy-preserving cryptographic primitives above have been combined with new secure hardware [32, 33] in order to improve performance. While this combination indeed brings the performance of privacy-preserving cryptography closer to the benchmarks needed for deployment, it is important to highlight that such an improvement comes at the expense of trusting the manufacturer of the secure hardware not to leak the information (or the key) to unintended parties.

In the case of database outsourcing, it is worth mentioning tailored solutions that combine different types of privacy-preserving cryptography in order to increase efficiency [34]. These databases rely on techniques such as homomorphic encryption, order-preserving encryption [35, 36], or deterministic encryption, among others. These schemes indeed provide great performance. However, it has been demonstrated that choosing weaker cryptographic primitives to favour efficiency may have a significant impact on privacy [37, 38, 39, 40]. Therefore, they are only recommended to support compliance, and should only be deployed in a trusted environment where attacks are unlikely. It is not recommended to use them in scenarios where data privacy is of critical importance and the entity that holds the database is not trusted.

The second scenario is collaborative computation, i.e., the entities involved in the communication collaborate to perform the computation. The result of this computation may be of interest for the sender, for the receiver, for both, or for third parties. Yet, if the participants do not trust each other, i.e.,

for a given entity, any of the other participants may be considered an adversary. Typical applications are comparing databases or computing statistics across datasets [41, 42]. Such applications can be supported by Multi Party Computation (see Section 9.4 in the Cryptography knowledge area [9]), as described by Archer et al. in [43]. When the goal of the application is to find similarities between two databases (e.g., contacts [44, 45], malicious activities [46], or genetic information [47]), one can also use lighter protocols such as Private Set Intersection [48, 49, 50]. These protocols allow two parties to compute the intersection of datasets without revealing anything except the intersection, or the cardinality of the intersection.

Verification in the encrypted domain. When data are processed in the encrypted domain, it is hard for the entities performing the computation to run any check on the adequacy of the inputs. To solve this problem, many primitives build on Zero-Knowledge Proofs (see Section 9.1 in the Cryptography knowledge area [9]) to prove to the entity performing the computation that the inputs comply with a certain format or with certain constraints. We now describe three cases in which verification in the encrypted domain is key to enabling the use of privacy-preserving cryptographic protocols.

Private computation - input verification. Zero knowledge proofs are very well suited to ensuring that the input to a privacy-preserving protocol is of a particular form or is not malicious. For instance, they have been used, among others, to prove the adequacy of inputs in billing applications, e.g., that they belong to a set of valid inputs [51], or are within particular ranges [52], to prove that there are no malicious inputs when requesting information from a messaging system [53], or when running a private intersection protocol [54].

Private authentication. To maintain confidentiality, entities participating in protocols may want to authenticate their communication partners. However, typical authentication systems are based on revealing the identity of the authenticating party. Revealing one's identity, in and of itself, may result in a privacy breach (e.g., when the authentication is against a sensitive service, such as a medical service). A solution to avoid this problem is the use of Anonymous Credentials, also known as Attribute-Based Credentials (ABCs) [55, 12, 56].

Instead of authenticating an entity with respect to an identity in order to grant authorisation, ABCs enable the entity to prove possession of a combination of different attributes to obtain the same authorisation. This proof does *not* reveal any additional information about the entity authenticating, nor does it reveal the concrete values of the attributes. Furthermore, ABCs are unlinkable between contexts. In other words, credentials look different every time they are shown, such that different showings cannot be linked to each other.

While from the point of view of privacy, ABCs bring many advantages, they also introduce new challenges. Anonymity may open the door to misbehaviour. Unfortunately, the strong anonymity and unlinkability properties provided by original ABCs do not allow an authority to limit or revoke authorisation for misbehaving users. In response, several schemes have appeared that provide capabilities to limit the amount of times that a credential can be used before the user is identifiable [57]; capabilities to blacklist credentials so that access can be temporarily revoked [58, 59]; or capabilities to completely revoke the credentials [60].

There exist several implementations of ABCs [61, 62] available under diverse licenses. These implementations offer different subsets of the functionalities mentioned above.

Private payments. Verification of encrypted data is also key to enabling privacy-preserving payments, in which the payer may have to prove to the buyer, for instance, that he has enough funds without revealing the exact amount. Early digital cash systems relied on Blind Signatures (see Section 10 in the Cryptography knowledge area [9]) to enable banks to sign e-coins [13]. In a nutshell, to extend an e-coin to a client, a bank would blindly sign a random value. To spend the e-coin, the client would give this number to the seller, who could redeem it at the bank. By storing the random number, banks can detect double spending, but not identify the double spender.

More recent privacy-preserving payment schemes, like the blockchain-based Zerocash [63, 64] system, include more information in the transactions to provide better guarantees. In each transaction, the user proves, in zero knowledge, that she owns the e-coins input to the transaction; that each one of the input e-coins was either recently mined (minted in Zerocash terms) or was the output of a previous transaction; and that the input and output values of the transaction are the same, i.e., no money would be lost. For the sake of efficiency, Zerocash relies on particularly efficient zero-knowledge proofs called zero-knowledge Succinct Non-interactive ARguments of Knowledge (zk-SNARK) systems [65]. These proofs are shorter (in the order of hundreds of bytes) and relatively fast to verify.

1.1.2 Obfuscation-based inference control

The protocols discussed in the previous section provide strong (cryptographic) guarantees regarding the confidentiality of data. Such strong protection, however, comes at the cost of efficiency and flexibility. On one hand, privacy-preserving cryptographic primitives require significant resources in terms of computation and/or bandwidth. On the other hand, they narrow down the type of processing that can be done on data. This is inherent to cryptographic constructions that fix inputs and outputs, and strictly define what information will be available after the protocol is executed.

In this section, we describe approaches to protect data confidentiality based on obfuscating the data exposed to an adversary. These techniques provide a more relaxed definition of confidentiality than cryptography, in the sense that they cannot completely conceal information. Instead, their goal is to provide a way to *control* the extent to which an adversary can make inferences about users' sensitive information. In fact, for most of these techniques, the level of protection depends on the concrete data and adversarial knowledge. Thus, it is important to run an ad-hoc analysis for the inference capability, as explained in Section 5. Also, we note that the privacy gained from these techniques is based on limiting the information available to one's adversary. This limitation implicitly reduces the amount of information available for anyone and, hence, may have an impact on utility if the purpose of the application is based on sensitive information, e.g., finding matches on dating applications. However, we note that when the sensitive information is not crucial for the purpose of the application in equation, e.g., a weather application that can operate using very rudimentary location data, these techniques may be deployed without affecting utility.

Obfuscation-based inference control techniques are not suitable for protecting data in transit, but can be used to support privacy-preserving outsourcing, privacy-preserving collaborative computations, and privacy-preserving publishing. There are four main techniques to obfuscate data, as described below. We note that these techniques are also typically used to reduce the information leaked by metadata. We discuss this aspect in Section 1.2.

For the sake of illustration, let us take the following microdata file as a current example. This is a very simple example, and we stress that the techniques introduced below can be applied to many types of data formats and domains.

An example database

Name	Age	Gender	ZIP	Salary
Alice	21	Female	21345	51300
Bob	32	Male	25669	67400
Carla	25	Female	18934	51500
Diana	64	Female	21223	60200
Eve	34	Female	18022	73400
Frank	37	Male	25321	55800
Gerald	19	Female	18235	68900

Anonymisation. A common technique used to permit data processing without risk for individuals is **data anonymisation**. Anonymisation, as its name indicates, seeks to decouple identity from informa-

tion. The idea is that removing identifying information from data points makes them unlinkable (i.e., they cannot be grouped as belonging to the same entity), thus hindering the ability of the adversary to perform inferences from the data.

However, achieving full anonymity is extremely difficult. In fact, when a dataset can be declared anonymous remains unclear. Data in and on themselves contains enough information to correlate different attributes and/or records on a database. Given these groups, there are many techniques to re-identify individuals behind the data release. A key insight into understanding the difficulty of anonymisation is the uniqueness of individual's data patterns [66, 67]. There may be many combinations of the information released in a dataset that are unique to an individual. These are called quasi-identifiers. Finding quasi-identifiers enables the re-identified data by mapping them to identifying information in other data sources [68, 16]. Thus, anonymisation is commonly combined with the obfuscation techniques described below to limit the risk of re-identification.

At this point in the knowledge area, it is worth referring to the notion of k-anonymity, which advocates combining generalisation and suppression in order to ensure that records on a database are anonymous among (i.e., indistinguishable from) at least other k entries in the same dataset [69]. For instance, in the example above, one can generalise the ZIP code to achieve two-anonymity:

Anonymization: *A two-anonymous database through generalisation*

Name	Age	Gender	ZIP	Salary
*	21	Female	21*	51300
*	32	Male	25*	67400
*	25	Female	18*	51500
*	64	Female	21*	60200
*	34	Female	18*	73400
*	37	Male	25*	55800
*	19	Female	18*	68900

While this notion is promising, there are several factors that make it unappealing and hard to use in practice. First, due to the uniqueness of the problem mentioned above, obtaining k-anonymity may require an unacceptable amount of generalisation in the database. Second, depending on the application, k-anonymity may not actually prevent inference of sensitive attributes. This is illustrated in our running example in the Gender column. Even though the generalisation in the ZIP code ensures two-anonymity, the adversary knows with a 100% probability the gender of the users in each ZIP area, e.g., all users living in 21* are women. Similarly, the adversary learns that females in their 20s earn approximately 51000.

To address this issue, researchers argue that privacy not only requires k-anonymity, but also l-diversity, which ensures that for each k anonymous individual, there are at least l possible values for the sensitive attribute [70]. Researchers have also shown that l-diversity can be broken and so-called t-closeness, where the set of sensitive attributes is not only diverse but follows the general distribution for this attribute across a population, is needed [71].

The k-anonymity notion is very popular in health-related applications [72]. It has also been adapted to fields other than databases [73, 74].

Generalisation. This technique consists in reducing the precision with which data are shared, with the goal of reducing the accuracy of the adversary's inferences. Generalisation can be achieved via a direct precision reduction of the shared values, or a bucketisation (i.e., a mapping from values to ranges) before data are released. This technique has been applied, among others, for database anonymisation [69], reducing the precision of the values in the different cells; or in private web searches [75], where words are mapped to the closest word of a pre-defined set.

Generalisation: *Reducing the precision of the ZIP code to the first two digits; reducing the precision of the Age column via bucketisation.*

Name	Age	Gender	ZIP	Salary
Alice	10-30	Female	21***	51300
Bob	30-40	Male	25***	67400
Carla	20-30	Female	18***	51500
Diana	60-70	Female	21***	60200
Eve	30-40	Female	18***	73400
Frank	30-40	Male	25***	55800
Gerald	10-20	Female	18***	68900

Suppression. This technique consists in suppressing part of the information before it is made available to the adversary. The rationale behind suppression is that the fewer the data are provided to the adversary, the more difficult is for her to make inferences. The suppression strategy, which decides which information to hide, is key for the level of privacy protection that such a scheme may provide. For instance, suppressing information at random is unlikely to destroy the patterns in the data that allow for inferences. Thus, unless most of the data are deleted, this strategy seldom provides good protection. A common strategy is small count suppression, where aggregated values below a threshold are not reported. The level of protection of this strategy depends on the type of access to the data and the knowledge of the adversary [76]. Other suppression strategies, tailored to the nature of the data under consideration and their characteristics [77, 78] provide better privacy results. This technique has been applied, among others, for database anonymisation [69], to hide some of the values in the different cells; or in location data publishing [77], to hide location samples that provide too much information.

Suppression: *Suppression of the Gender attribute for 50% of the records.*

Name	Age	Gender	ZIP	Salary
Alice	21	Female	21345	51300
Bob	32	Male	25669	67400
Carla	25	*	18934	51500
Diana	64	*	21223	60200
Eve	34	Female	18022	73400
Frank	37	*	25321	55800
Gerald	19	Female	18235	68900

Dummy addition. This technique consists in adding fake data points, so-called *dummies*, to the data made available to the adversary in order to hide which are the real samples. The idea is that, as the adversary considers fake points when running the attack, her inference will have errors. For this defense to be effective, fake points have to be indistinguishable from real points. Ideally, from the point of the view of the adversary, any sample should look like a real or dummy one with equal probability. However, creating such indistinguishable points tends to be difficult [79], and the adversary can easily filter them out. Thus, this technique is useful in very few domains. Dummy addition techniques have been used to increase privacy in web searches [75, 80] or to protect databases from inferences [81].

Dummy addition: *Adding 50% of fake records (in blue).*

Name	Age	Gender	ZIP	Salary
Alice	21	Female	21345	51300
Bob	32	Male	25669	67400
Carla	25	Female	18934	51500
Donald	54	Male	25669	53500
Diana	64	Female	21223	60200
Eve	34	Female	18022	73400
Frank	37	Male	25321	55800
Goofy	61	Male	21346	41500
Gerald	19	Female	18235	68900
Minnie	23	Female	18456	62900

Perturbation. Perturbation techniques inject noise into the data made available to the adversary. The noise is aimed at reducing the adversary's inference performance. Similar to suppression techniques, the strategy used to introduce noise plays a crucial role in the level of privacy provided. Initial schemes drew noise from many kinds of random distributions [82, 83] and added them to the data. This approach was not really effective, as an adversary with knowledge of the noise distribution could infer the original data values with reasonable accuracy and thus risked leaking more information than intended.

Perturbation: *Obfuscating salary with noise drawn from a normal distribution N(0,1000).*

Name	Age	Gender	ZIP	Salary
Alice	21	Female	21345	51345
Bob	32	Male	25669	67863
Carla	25	Female	18934	51053
Diana	64	Female	21223	60302
Eve	34	Female	18022	74558
Frank	37	Male	25321	55005
Gerald	19	Female	18235	69425

Currently, the gold standard in perturbation-based techniques is to add noise to achieve so-called ***differential privacy***. The main goal of this technique is to address the limitations of data anonymisation techniques for publishing such as the aforementioned k-anonymity.

Differential privacy, introduced by Dwork [84], is a privacy definition originally intended to enable the design of techniques that permit maximising accuracy when querying statistical information (mean, variance, median etc.) about users on a database while minimising the risk of unintended inferences. Rather than a property of a dataset, differential privacy is a property of a mechanism used to output the answers to queries against a dataset. An algorithm is differentially private if, by looking at the result of the query, the adversary cannot distinguish whether an individual's data were included in the analysis or not. More formally, an algorithm $\mathcal{A}$ provides ϵ-differential privacy if, for all datasets D_1 and D_2 that differ on a single element (i.e., the data of one individual), and all possible outputs S of the algorithm:

$$\Pr[\mathcal{A}(D_1) \in S] \leq e^{\epsilon} \times \Pr[\mathcal{A}(D_2) \in S],.$$

Differential privacy ensures that, given a perturbed data sample, the adversary gains a negligible amount of new information about the original data sample with respect to the her prior knowledge, regardless of what this prior knowledge was. There exist a number of algorithms to ensure that differential privacy is met for a variety of queries [14].

Differential privacy is an extremely useful definition because it gives a formal framework to reason about the amount of information a powerful adversary might be able to infer about individuals in the data, regardless of the adversary's prior knowledge. However, it must be noted that:

- Differential privacy provides a *relative* guarantee, as opposed to an absolute privacy protection. This means that the protection provided is regarding the prior knowledge of the adversary. If the adversary already has full knowledge, differential privacy will not improve privacy. In other words, differential privacy ensures that the release of data does not worsen the privacy loss of a user or population by more than a set threshold. However, this does not automatically ensure that a user's privacy is preserved overall. Therefore, to claim privacy, it is important to not only ensure that a scheme provides a given guarantee, but also computes the adversarial error on the inferences so as to ensure that users' sensitive information is actually protected (see Section 5).
- One of the current practical challenges of differential privacy is to determine what values of ϵ provide an acceptable level of privacy. The level of protection of crucially depends on the value of this parameter. This means that merely fulfilling the differential privacy definition with arbitrary parameter values does not directly guarantee that the adversary does not learn too much new information from the data. It is important to ensure that the value of ϵ is such that the probabilities for different inferences are actually indistinguishable. For example, if $\epsilon = 3$, this only ensures that the ratio between the probability of observing a result when an individual is, or is not, in the dataset is: $\Pr[\mathcal{A}(D_1) \in S] / \Pr[\mathcal{A}(D_2) \in S] \leq e^3 = 20.08$. This probabilistic difference can typically be detected by classical statistical detectors, or any modern machine-learning classifier. In general, ϵ values greater than one deserve a closer look to verify that the algorithms provide the sought level of protection.
- The amount of noise required to hinder inferences on the data depends on the so-called *sensitivity* of the algorithm. Sensitivity measures how much a change in the input will change the output of the algorithm $\mathcal{A}$. When the input is a database and the output a statistical function, small input changes have little influence on the output and, thus, a small amount of noise is enough to make the algorithm differentially private. We note, however, that when differentially private algorithms are applied to protect the privacy of a single sample instead of to the result of a statistical query on a database, the sensitivity may be much higher. Thus, only a large amount of noise may ensure protection. For instance, when differential privacy is applied to obfuscate a user's reported position to obtain location privacy, protection may be less than expected if the parameters are not carefully chosen [85].
- Differential privacy provides a worst-case guarantee, which means that the amount of noise introduced is tailored to bound the leakage given by the data point in the dataset that provides the most information to the adversary with the best knowledge. This means that in an average case the amount of noise is larger than needed. Recent studies have been working towards tighter bounds that permit reduction in the noise required to provide a desired protection level [86].

The differential privacy notion has been extended to account for metrics other than the Hamming distance (i.e., distinguishing whether one individual is in a database or not) [87]. Perturbations with differential privacy guarantees have been used to protect, among others, privacy in collaborative learning [88], or locations when querying location-based services [89]. It has also been recently adopted by the US to protect census data [90].

Finally, it is important to remark that for many real cases, one of these inference controls cannot provide enough privacy on its own. Therefore, typically one needs to combine several of these techniques to limit the numbers of inferences that can be made.

1.2 Metadata Confidentiality

In the previous section, we discussed means to protect the confidentiality of the contents of messages, databases, queries, etc. These techniques are essential to ensure privacy. Yet, they do not protect against an adversary that uses the metadata to infer sensitive information about individuals. Concretely, there are three types of metadata that have been demonstrated to be extremely vulnerable to privacy attacks: traffic metadata, associated to the communication infrastructure; device metadata, associated with the platform generating the data, and location metadata, associated with the physical location from which data is generated.

In this section, we discuss the privacy risks associated with these types of metadata and the relevant controls to address these risks.

Traffic metadata. Network-layer information, such as the identities of the participants in the communication (IP addresses), the amount and timing of the data transferred, or the duration of the connection, is accessible to observers even if communications are encrypted or obfuscated. This information, commonly known as traffic data, can be exploited to deduce potentially sensitive private information about the communication. For instance, in an e-health contex,t messages are generally encrypted to preserve patients' privacy. However, the mere fact that a patient is seen communicating with a specialised doctor can reveal highly sensitive information even when the messages themselves cannot be decrypted. Confidential communications are not only desirable for personal reasons, but they also play an important role in corporate environments. The browsing habits of a company's employees (e.g., accessing a given patent from a patent database) can be used to infer the company's future lines of investment, thus giving an advantage to their competitors.

A technique to protect traffic data is the use of ***anonymous communications networks***. These networks are typically formed by a series of *relays* such that communications do not travel directly from origin to destination, but are sent from relay to relay. These relays also change the appearance of a message through means of encryption to provide *bitwise unlinkability*, i.e., to ensure that packets cannot be linked just by looking at their bit content; they can also change traffic patterns by introducing delays, re-packaging messages, or introducing dummy traffic.

Anonymous communications networks follow different designs regarding how the infrastructure is built (by users or dedicated relays), how they consider communications (message-based vs. flow-based), or how they reroute messages (deciding a route at the source, or letting relays decide on routing), among others. In the following, we focus on the two most known anonymous communications network types which have real-world deployment. We refer readers to the surveys by Danezis et al. [91] for a historical overview of anonymous communications and by Shirazi et al. [17] for a comprehensive overview of more recent anonymous communication systems.

The most popular anonymous communication network is Tor[2] [92]. The core element of the Tor Network are Onion Routers (ORs), which are essentially routers that forward encrypted data. ORs encrypt, respectively, decrypt packets along the way to achieve bitwise unlinkability, as detailed below. When a user who wants to anonymously access an Internet service through the Tor network, she installs a Tor client in her device. This software builds a *circuit* of connections over three ORs, called entry, middle and exit nodes, and the client routes encrypted traffic to the destination server through this circuit. Tor uses so-called *onion encryption*, in which the client establishes a secret key with each of the ORs in the circuit using an adapted version of authenticated Diffie-Helmann [9]. Every packet routed through the circuit gets encrypted with these three keys, first with the exit OR's key, then the middle OR's key, and finally that of the entry OR. When the message travels through the circuit, the nodes 'peel' each layer of encryption until the original packet is sent to the destination. The server sends data to the client using the same circuit, but in the inverse order; i.e., the server encrypts the message in layers that are decrypted by exit, middle, and entry ORs.

[2] `https://www.torproject.org/`

In order to enable low-latency applications, Onion Routers do not impose delays on the messages they receive and resend. Thus, traffic patterns are conserved while packets travel through the network. This enables an adversary with the capability to observe both ends of the communication (i.e., the entry and exit nodes) to correlate incoming and outgoing flows in order to link the origin and destination of communications [93].

In order to destroy traffic patterns and protect correlation attack relays in an anonymous communication, networks need to delay packets or add new ones. This is the principle behind the design of *mix networks* [94]. As opposed to onion routing, where all the packets from a communication are routed through a circuit, in mix-based communications routes are selected for every message. Then, when a mix relay receives a packet, instead of immediately decrypting and send it to the next hop on the path, the message is delayed. How many messages are delayed is determined by a *firing condition*, which is an event such as the arrival of a message or the expiration of a timeout that causes the mix to forward some of the messages it has stored. Which messages are fired depends on the *batching strategy*, which can select all of the messages or a fraction according to a probabilistic function. Both mixes and users can send dummy traffic, which may be absorbed by other mixes or by the recipient.

A mix network designed to provide low latency is Loopix[3] [95]. As opposed to Tor, where users' clients communicate directly with the Tor nodes, Loopix assumes that users communicate with providers that in turn send messages to each other through the Loopix anonymous communication network. Providers choose a random route composed of Loopix routers and send the message to the first node. Similar to Tor, messages get encrypted with the keys of each of these routers using the Sphinx packet format [96]. In addition to the encryption, messages are assigned a delay for every relay they visit according to an exponential distribution. Finally, providers inject dummy traffic into the network by sending packets to themselves via a Loopix path, so as to provide cover for real messages. The combination of providers that hide mix messages from users sending (respectively receiving) messages at the same time, delays and cover traffic enable Loopix to provide provable guarantees regarding the unlinkability of the senders and receivers of messages.

Device metadata. In today's optimised Internet services, the concrete characteristics of users' devices are frequently sent along with their data requests in order to optimise the service providers' responses. Even if users are anonymous on the network layer, these characteristics may become a quasi-identifier that enables service providers to track users across the web [97, 98]. This is because combinations of features such as the User Agent (the browser software vendor, software revision, etc.), its Language, or the Plugins it has installed, or the platform are mostly unique.[4].

Device or browser fingerprinting is the systematic collection of this information for identification and tracking purposes. A large number of attributes, such as browser and operating system type and version, screen resolution, architecture type, and installed fonts, can be collected directly, using client-side scripting and result in unique fingerprints. When this information is not directly available, other techniques can be used to learn this information, as explained below.

As an illustrative example, let us consider the list of fonts installed on a particular user's web browser as an identifier that enables tracking. There are two techniques to obtain the list of installed fonts, which is known to provide a good level of uniqueness. This is because browsers install fonts on demand depending on the sites visited. Since users have different browsing patters, their lists of installed fonts become different as well. Font fingerprinting techniques exploit the fact that if a font is installed, browsers will render it, but if not. browsers will revert to monospace font. Thus, depending on whether a font is installed or not, sentences will be rendered differently. In the first technique, the tracking web sends a sentence to the browser to be printed with a series of fonts. Then, the client-side script checks the size of each sentence. When the size is equal to the sentence printed in monospace, the tracker learns that the font is not installed. A similar technique is called canvas

[3] `https://katzenpost.mixnetworks.org/`

[4] `https://amiunique.org/`, `https://panopticlick.eff.org/`

fingerprinting. In this case, the tracker exploits the HTML5 Canvas feature, which renders pixels on the fly. As before, different font size result in different pixel footprints. Measuring the result of the canvas rendering the tracker can ascertain which fonts are installed in a browser.

Defending against device metadata attacks while retaining utility is extremely difficult. On the hand, hiding these metadata from service providers has an impact on the performance of the services, since it limits personalisation and deteriorates the rendering of information. On the other hand, it is hard to establish which combination of features would actually make users indistinguishable from other users. This is because we have no knowledge of the distribution of fingerprints in order to imitate one of them, and trying combinations at random runs the risk of being as unique as the original fingerprint [99]. Therefore, mechanisms need to be carefully crafted and evaluated [100].

We note that besides tracking based on metadata, trackers also use a series of techniques based on the use of cookies. For instance, web pages can include cookies from third parties, which allows these parties to detect when users revisit a page [101]. Third parties can also use cookie syncing, whereby, besides adding their own tracking, webs redirect cookies to other trackers to inform them of where the users are going [102]. Finally, there exist pervasive mechanisms to install cookie-like information that cannot be removed by cleaning the browser's cache [10].

Location metadata. Finally, a user's geographical location revealed to online services can be used to infer sensitive information. This information can be revealed explicitly, for example, when the user makes queries to location-based services to find nearby points of interest or friends; or implicitly, for example, when GPS coordinates are associated with photos or content published on social networks, or inferred from the access point used to access the Internet.

Clustering techniques to find groups of nearby points where the user spends not significant amounts of time can be used to infer users' points of interest such as where they live, where they work or their favourite leisure places. In many cases, points of interest can be used as quasi-identifiers for users. For instance, the three most commonly visited locations are unique to a user in a majority of cases, even when the locations are provided on a more general level (e.g., US counties) [103]. Similarly, the types and patterns of locations visited can be used to infer demographic data about users such as age or gender [104]. Furthermore, once movement patterns have been characterised, they can be used to predict individuals' future movements [105].

There are two kinds of defence for protecting location metadata in the literature. The first relies on cryptographic techniques to process location-based services' queries (see Section 1.1.1). For instance, users can privately learn whether a friend is nearby. This service can be realised by homomorphic encryption [106], private equality testing [107] or private threshold set intersection [108]. The second kind of defence is based on the obfuscation techniques described in Section 1.1.2 in order to control the inferences that an adversary draws from the location data. For instance, user's location can be hidden [109], i.e., not reported to the provider; perturbed [110], i.e., reporting a location different from the user's actual position; generalised [111], i.e., reported with less precision; or accompanied by dummy locations so that the user's real movement patterns cannot be identified [112].

2 Privacy as Control

[113][114][115]

In the previous section, we discussed privacy technologies that keep data confidential, by minimising the collection of data and/or minimising the amount of information that can be inferred from any released data. A wider notion of privacy, which is usually referenced in regulations, broadens privacy from the notion of concealment of personal information, to the ability to control what happens with the information that is revealed [4, 6].

The idea behind the shift from technologies that minimise disclosure to technologies that provide

the means to control information use, is that in many cases, revealing data may be unavoidable or perceived as beneficial to the data subject. Thus, it is advisable to consider the use of technologies that address two major concerns: i) enable users to express how they expect that data disclosed to the service provider are used, so as to prevent undesirable processing of these data; and ii) enable organisations to define and enforce policies that prevent the misuse of information, as defined by the users.

In this section, we revise techniques that have been designed under the privacy as control paradigm. We focus on techniques for the creation and configuration of good privacy settings that help users express their preferences with respect to data disclosure and processing; and techniques that support the automated negotiation of privacy policies across services. Because much of the protection relies on trust, privacy technologies that enhance privacy in a system through improved control are less numerous and varied than those designed to achieve confidentiality.

It is important to highlight that these techniques inherently trust the service provider that collects the data to correctly enforce the policies established by the user with respect to third parties, as well as not to abuse the collected data itself. Also, as noted by Acquisti et al. [113], providing users with tools to control information flows can reduce risk perception and increase risk-taking, effectively reducing the overall privacy of users.

2.1 Support for privacy settings configuration

Privacy settings are those controls in a web service that allow users to express their preferences regarding how data should be revealed to other users, shared with third parties, and processed by the service providers. Madejski et al. have shown that the complexity of these privacy settings makes them barely usable by individuals [114]. This lack of usability causes users to misconfigure their privacy settings, i.e., establish configurations that do not match their expectations. This in turn results in unintended disclosure of data. We refer readers to the Human Factors knowledge area for further information about the impact of usability of systems on security and privacy.

To counter this problem, researchers have proposed a number of techniques whose goal is to identify groups of individuals that share certain characteristics, and then establish the most adequate settings for each user group. One area of research suggests letting security and privacy experts define what are the best policies are. [116]. This approach, however, is difficult to generalise from targeted groups to the general population, and in many cases may result in strategies that overestimate the need for protection. This in turn limits too much the sharing and processing of data, thus rendering systems unusable. Other proposals advocate for using machine-learning techniques to infer adequate settings for a user based on the social graph of a user's friends and acquaintances [96]. This technique, however, requires users, or a centralised recommender system, to know a user's social graph in order to perform the inferences, which raises privacy concerns in itself. A third approach does not require knowledge of a user's social graph, but tries to find adequate privacy settings by looking at a larger set of users. [117]. As opposed to the previous technique, a user's suggested settings preference is derived from generic data. These techniques have been shown to be prone to produce policies that are valid for the majority of users, but often discriminate against user groups with specific privacy requirements such as activists or persons of public interest. Furthermore, ML-based techniques often augment and perpetuate biases present in the data from which the initial policies are inferred. A final research areas suggests crowdsourcing the optimum composition of these policies [118]. These techniques are more flexible in the sense that users have more leeway to influence the policies. However, they are still influenced by majority votes and may not be ideal for users who do not follow mainstream practices.

2.2 Support for privacy policy negotiation

The previous technologies support users at the time of configuring their privacy settings in an online service. An orthogonal line of work is dedicated to automating the communication of user preferences to the service, or between services.

Technologies such as the W3C's Platform for Privacy Preferences Project (P3P) [119], which facilitate the communication of setting preferences between user and service provider. P3P is an industry standard that allows websites to encode their privacy policies (what information is collected, how it is used etc.) in a pre-defined format. These policies can be read and interpreted by browsers equipped to do so. The browser can then compare the site's policy with a user's specified privacy preferences written in a machine readable language such as P3P Preference Exchange Language (APPEL) [120]. P3P, however, does not have any means to enforce that the service provider actually follows the practices described in the policy.

Other technologies such as purpose-based access control (PBAC) [121] or sticky policies [122] provide the means to specify allowed uses of collected information, and to verify that the purpose of a data access is compliant with the policy. These technologies can be supported by cryptographic mechanisms that guarantee that the service providers must comply with the preferences established by users.

2.3 Support for privacy policy interpretability

In order to configure the privacy settings according to their expectations of how data should be handled, users need to understand the privacy policies that describe the meanings of these settings. These policies are often long, verbose, and contain a lot of legal terms; and they often evolve over time. Thus, users find them difficult to understand. Researchers have developed technologies that enhance users'ability to interpret privacy policies.

Currently, there exist two approaches to improve users' understanding of privacy policies. One is to trust experts to label, analyse and provide reasons for existing privacy policies [123]. Another avenue is to completely automate the interpretation process. Polisis[5] [115] is a machine-learning-based framework that enables users to ask questions about natural language privacy policies. This tool offers a visual representation of the policy specifying the types of data collected, the purpose of this collection, and the sharing practices, among others.

3 Privacy as Transparency

[124][125][126]

The last privacy design paradigm we consider is privacy as transparency. As opposed to technologies that limit data disclosure or the use of disclosed data, transparency mechanisms analyse users' online activities in order to either provide them with feedback about the implications of their actions, or run audits to check that there has been no violation of privacy.

As with control-oriented technologies, transparency-based privacy cannot prevent privacy violations in and of themselves. In fact, feedback or audits happen *after* the users have already disclosed data to the provider. Thus, providers are again trusted with making sure that the collected data are not processed or shared in ways not authorised by the users.

3.1 Feedback-based transparency

We first describe mechanisms that make transparent the way in which information is collected, aggregated, analysed and used for decision making. The common factor between these technologies

[5] `https://pribot.org/polisis`

is that they provide users with feedback about how their information is processed or perceived by others.

An early effort in this direction is the concept of privacy mirrors [124], which show users their 'digital selves'; i.e., how others see their data online. This concept was adopted by popular online social networks such as Facebook, which allows users to check how different audiences (e.g., friends, friends of friends, others), or even individual users, see their profiles whenever they make changes to their privacy controls. A similar line of work provides other means of visualising how privacy settings affect data sharing in order to improve users' understanding of the set of permissions they have selected. This solution provides visual cues to users that indicate the access permissions associated with the data they shared [127]. For instance, it can highlight fields in a social network profile with a different colour depending on who has access to that particular information. Both solutions help users understand their practices and modify their actions. However, they can only do so after the information has been revealed to the provider (and possibly to other users).

A different type of user feedback comprises so-called privacy nudges [125]. Nudges assist users in making choices about their privacy and security settings. They give users *immediate* feedback whenever the user performs an online action in a way that the action could be cancelled or modified. For instance, the nudge can inform the user that the post she is currently writing is public so that the user is careful about the words she chooses to use. Nudging tools can be even more sophisticated and use modern machine learning algorithms to analyse photos or text as they are being uploaded, and provide users with more concrete feedback such as 'the post can be perceived as negative' or 'the photo is very explicit'. While immediate feedback presents evident benefits compared to mirrors, since actions can be modified before information is sent to the provider, it also has drawbacks. Experiments with users have shown that immediate feedback results in an uncomfortable feeling for users as they feel monitored, and users sometimes perceive the advice as paternalistic and out of place [126].

3.2 Audit-based transparency

As mentioned before, even with privacy policies and access control in place, there is no guarantee that user preferences will be respected. Additional measures can be put in place to enable users to verify that no abuse has taken place. To realise these audits, the system is required to log all data access and processing operations. This logging may reveal when users log into the system, and when and how their data are transmitted to others. Thus, depending on the amount and granularity of the information, logging may introduce additional privacy risks.

Therefore, logging policies must be carefully crafted. One approach to do this is to derive the auditing specifications from the policies using formal methods [128]. This guarantees that the generated logs, while being minimal, still contain enough information to audit whether the policies are being respected. The solutions, however, are limited in their expressiveness and cannot handle privacy policies in modern systems where the amount of data collected and the number of entities involved make a formal analysis extremely cumbersome.

The use of formal methods assumes that data sharing is managed by a centralised authority that must be trusted. This is problematic because the centralised authority becomes a single point of failure. Recent advances in cryptography and distributed ledgers permit the design of solutions that provide the means to create highly secure logs, while ensuring that no private information is shared with unauthorised parties. When logging is made in such a distributed manner, no individual party can modify the log on its own, reducing the need for trust and eliminating any single point of failure. For instance, systems like UnLynx [129] permit entities to share sensitive data, and perform computations on them, without entrusting any entity with protecting the data. All actions are logged in a distributed ledger for auditing, and the correctness of the operations is ensured by using verifiable cryptographic primitives and zero-knowledge proofs. Therefore, it is not necessary to publish or log the sensitive data or the operations done on them.

4 Privacy Technologies and Democratic Values

[130][131][132]

Privacy technologies are of paramount importance to ensure that our fundamental right to privacy is respected in the digital world. Privacy protection is crucial for underpinning the values that support our democratic societies. Citing Daniel Solove: 'Part of what makes a society a good place in which to live is the extent to which it allows people freedom from the intrusiveness of others. A society without privacy protection would be suffocation' [132]. While such a society seemed science fiction not so long ago, scandals such as the Facebook Cambridge Analytica case highlight the importance of securing data from being accessed by unintended parties (e.g., using confidentiality or control techniques) to protect citizens from interference and manipulation.

In this section, we provide two examples that highlight the importance of privacy technologies in supporting democracy. On one hand, we consider electronic voting systems that enable fair elections to take place using electronic infrastructure in adversarial conditions. On the other hand, we give an overview of censorship resistance technologies. These systems ensure that in a digital world, where communication infrastructure is dominated by a small number of companies and state actors can observe all communications, individuals have the means to communicate freely.

4.1 Privacy technologies as support for democratic political systems

The growing use of electronic applications to interact with governmental bodies brings great advantages to society. Providing citizens with easy means to express their opinions, comment on government initiatives, or vote in elections, increases their involvement in public decision processes. This in turn improves the power balance between those who can execute decisions and those who are affected by the outcome of the decision process.

For these improvements to be effective, citizens must be able to freely express their opinions and must be sure that their inputs cannot be modified or lost during the process. The use of common infrastructures (e.g., cloud services or unprotected communication networks) to implement these democracy-oriented applications, however, raises concerns about surveillance and manipulation. Therefore, it is important that these applications are supported by strong privacy technologies that can protect users' identities, as well as their sensitive data and inputs to the system. We describe two example applications, electronic voting and electronic petitions, whereby the technologies introduced in the previous sections are combined to enable citizens and governments to enjoy technological progress without compromising our democratic values.

Electronic voting (eVoting). Electronic voting systems have the goal of enabling fair elections to be conducted via electronic infrastructure in adversarial conditions. In particular, eVoting schemes provide:

- *Ballot secrecy*: an adversary cannot determine which candidate a user voted for.
- *Universal verifiability*: an external observer can verify that all the votes cast are counted and that the tally is correct. Some protocols provide a weaker property, *individual verifiability*, where each voter can verify that his/her vote has been correctly tallied.
- *Eligibility verifiability*: an external observer can verify that all the votes cast were made by a unique eligible voter.

In order to guarantee the first aspect, it is key to break the links between the votes putting their ballots into the system, and the ballots that come out. In traditional pen-and-paper physical elections, this is done by mixing the ballots all of which have exactly the same appearance in an urn. In eVoting,

unlinkability is typically achieved using mix networks [133, 134]. the votes are passed through a series of mixes, which must not belong to the same authority. Otherwise, this authority could trace the votes and link the voters to their voting choices. The results are published on a public bulletin board which anybody can read and verify that the election was carried out in a honest manner.

Voting mix networks are designed in a slightly different way than those mentioned in Section. 1.2. In the case of eVoting, the mixes fire when all votes are in, and the batching strategy is to take all the votes. In simple terms, it ensures that all votes are mixed together, obtaining the maximum anonymity set. This fulfills the ballot secrecy criterion as any vote could have been cast by any voter. Furthermore, to ensure universal verifiability, in eVoting mix networks every node does *verifiable shuffles* [130]. This means that the mixes prove, in zero knowledge, that they mix all the votes (all the votes at the input appear at the output) and the mixing is random. Eligibility verifiability can be obtained by requiring voters to prove in zero-knowledge that they are eligible to vote.

Other voting protocols provide ballot secrecy through the use of blind signatures: an authorised entity verifies the eligibility of a user and blindly signs her vote (i.e., without seeing the vote content) [135]. The user provides a zero-knowledge proof along with the vote that the vote has been correctly constructed. Then, users submit the signed votes to the tally server using an anonymous communication channel. This way no entity in the system can link voter to votes.

A third strategy is based on homomorphic encryption. In these schemes, the tally server creates a bulletin board with encrypted zero entries for every candidate [136]. Then, every user adds his vote to the desired candidate, and randomises the rest of the encryptions (so that encryptions of the same number never look the same). As before, zero-knowledge proofs can be used to ensure that sums and randomisation have been performed in the correct way.

Besides the above three properties, some voting protocols additionally aim to provide *coercion resistance*, whereby a user cannot be forced to vote for a particular candidate against her will. One strategy to implement such a system is to provide users with fake credentials [137]. Then, when users are under coercion they follow the instructions of the coercer, but provide their fake credentials to the system. This enables the tally server to ignore any votes produced under coercion. Related approaches prevent coercion via re-voting, i.e., the schemes permit users to recast a vote so as to cancel their coerced choice [138]. These schemes define policies to establish how to count votes whenever a given credential has cast more than one vote. (e.g., count the last one, or add a pointer to the cancelled vote).

Anonymous petitions. We define a petition as a formal request to a higher authority, e.g., parliament or another authority, signed by one or more citizens. Signing a petition publicly, however, might raise concerns or conflicts in terms of relationships between friends, colleagues, and neighbours, discouraging citizens from participation [139]. Privacy technologies, in particular, anonymous credentials, can help in creating secure and privacy-preserving petition systems.

In petition systems based on anonymous credentials, citizens can register with the authority managing the petition system to obtain an anonymous signing key associated with some attributes relevant for the petitions. Then, at the time of signing a particular petition, they can prove they are eligible (e.g., they are inhabitants of the municipality referred to) but do not need to reveal their identity. Advanced credential properties such as double signing detection enable the creation of this system while avoiding abuse from misbehaving citizens [140].

More modern approaches rely on homomorphic encryption and threshold decryption to remove the need for a central trusted party that registers users. Instead, they make use of a distributed ledger to permit a decentralised authority to issue the credential, increasing the level of privacy in the system, while at the same time reducing the need to trust one single party [141].

4.2 Censorship resistance and freedom of speech

Censorship systems attempt to impose a particular distribution of content across a system. They may prevent users from publishing particular content that is considered controversial or dangerous for the censorship regime; or they may prevent users from accessing content that may undermine the societal equilibrium that the censor wishes to impose on their society.

In this section, we show how privacy-preserving technologies can act as a cornerstone to support freedom of speech and freedom of access to information. We will elaborate on some examples for each of these goals in order to illustrate the fundamental principles that make censorship resistance possible. We refer the interested reader to the surveys by Khattak et al. [131] and Tschantz et al. [142] for a comprehensive revision of censorship resistance systems.

Data publishing censorship resistance. Motivated by the 'Church of Scientology' court order, which caused the closure of the Penet remailer at the end of the 1990s [143], Anderson proposed the Eternity Service. This was the first system to use privacy technologies to protect the publishing of content on the Internet [144]. Anderson's scheme proposed to distribute copies of files across servers in different jurisdictions, so that those servers cannot be subpoenaed at the same time. In this scheme, privacy technologies have fundamental roles for resistance: encryption not only provides privacy for users, but also prevents selective denial of service at retrieval time; and anonymous authentication not only protects users from the service, it also protects the service from being coerced into revealing the identities of users, e.g., by law enforcement, since it cannot know these users' identities.

Anderson's proposal inspired later designs such as Freenet, a peer-to-peer system to publish, replicate, and retrieve data while protecting the anonymity of both the authors and readers [145]. Additionally, the system provides deniability for the entities storing the information; i.e., the servers cannot know the content of the files they store and thus, can always claim to be unaware of what they are serving. In Freenet, files are located according to a key that is typically the hash of the file, but can also include a file description, or be a simple string. To retrieve a file, a user obtains or computes the keys and asks Freenet nodes to find it. If a node does not contain the file, it asks a neighbour. When the file is found, it is sent back along the same path that the request followed in the network. This ensures that the node holding the data does not know the recipient. To store a file, if the key does not already exist, the publisher sends the file along a path and every node on the path stores the file. To protect the anonymity of the publisher, nodes that store the file also decide at random whether to also claim ownership. Such random claims also provide nodes with deniability as to which of the files they are storing are actually theirs.

The design of Freenet is based on strong cryptography, which protects the content of messages. However, in the early days, the routes and timing of messages allowed attacks to break the system's anonymity. Tian et al. [146] show that a passive attacker deploying a number of nodes in the network that can monitor requests can re-identify the requester by recursively asking other nodes if they have seen the request. Freenet also allows for the collection of privacy-preserving statistics. However, the statistic obfuscation method is vulnerable to inference attacks where the adversarial node combines several queries in order to learn information about other Freenet nodes' properties (e.g., bandwidth) [147]. These issues are now addressed by Freenet, but others remain such as the attack by Levine et al. [148], which enables a single node to distinguish whether a neighbouring peer is the actual requester of a file or just forwarding the requests for other peers. The attack only requires passive observation of traffic, and exploits the fact that the Freenet protocol determines the average number of requests for a file observable by a node depending on how far this node is from the requester. Thus, a simple Bayesian inference suffices to detect whether a neighbour is the request initiator.

A different approach to censorship is followed in Tangler [149]. The system also provides publisher

and reader anonymity, but achieves censorship resistance in a different way. Instead of simply storing the file replicated in many nodes in an anonymous way, Tangler files are split into small blocks that are stored on different servers. In order to recover a file, one must thus contact a number of servers to retrieve enough of these blocks. In order to avoid a server being compelled to delete a block belonging to a file, Tangler builds blocks in such a way that blocks contain parts of many documents. To 'tangle' files into blocks, Tangler uses secret sharing (See the Cryptography knowledge area [9]). Tangling improves availability in two ways. First, a censor can only delete a target file by causing collateral damage to other files that may be allowed. Second, whenever one wants to replicate a file, the files entangled in the replicated file blocks are also replicated.

Data access censorship resistance. To enable censorship-free access to data, systems must be able to conceal that users are accessing these data. This can be done in a number of ways [131]. A first approach is *mimicking*, where censorship resistance is obtained by attempting to make accessing to censored data look like accessing allowed data (e.g., as a Skype call [150] or as a visit to an innocuous web page [151]). These approaches are effective, but have been shown to be vulnerable to active attacks in which the adversary probes the suspicious connection to find out if any of the expected functions of the application being mimicked are missing [152].

A second approach is tunnelling. In this case, the censored communication is directly tunnelled through an uncensored service, instead of pretending to be that service. In particular, these systems use widely used services as tunnels, e.g., cloud services [153, 154], so that blocking communications imposes a high cost for the censor. A third approach is to embed the communication inside some content (e.g., hidden in a photo or video [155]). This approach not only makes communications unobservable, but also deniable for all senders, recipients and applications hosting the content.

Finally, some censorship resistance systems rely on hiding the destination of the communication to prevent censors from blocking connections. This is achieved by relaying censored traffic through one or more intermediate nodes. These nodes can be proxies, such as *bridges* in the Tor network [92]. These bridges are Tor relays whose IPs are not public so that they cannot be identified as members of a censorship resistance system. To avoid the censor identifying connections to bridges due to their appearance, these are disguised using so-called pluggable transports [156], which transform the traffic flow following one of the approaches referenced in this section.

Another option to hide the destination is the use of *decoy routing*, also known as *refraction networking* [157, 158]. In decoy routing, clients direct their censored traffic to a benign destination. This traffic includes an undetectable signal that can only be interpreted by a cooperating Internet router. This router deflects the client's traffic to the censored site and returns the responses to the client. Obviously, the cooperating router must be outside of the censor's domain, but, depending on the scheme, it can be on the forward path from the client to the uncensored destination [157, 158], or on the downstream path [159, 160].

5 Privacy Engineering

[161][162]

The growing privacy concerns in society have made the concept of 'privacy by design' very popular among policy makers. This concept advocates for the design and development of systems that integrate privacy values to address users' concerns. However, the literature around this concept rarely addresses the actual processes behind the design, implementation, and integration of privacy protections into products and services.

In this knowledge area, we first gave an overview of the landscape of privacy technologies, and subsequently provided a series of examples in which these technologies are combined to support the use of electronic systems while maintaining core democratic values. In this section, we elaborated on

the design principles behind these, and other, privacy-preserving systems. We briefly discussed how these principles can be used to generally approach the engineering of systems that embed strong privacy protections. We refered the reader to the work by Gürses et al. [161] for a more comprehensive explanation of these principles and their role in the design of privacy-preserving systems.

The two primary goals when designing privacy-preserving systems are to:

- **Minimise trust**: limit the need to rely on other entities to behave as expected with respect to sensitive data. For instance, in mix-based eVoting, trust is not only distributed across the entities managing the mixes, but verifiable shuffles are put in place to limit to a maximum the amount of reliance on the good behaviour of each mix. Similarly, the cryptographic primitives used to implement privacy-preserving electronic petitions do not require trust on the registration authority to protect the identities of the signers.
- **Minimise risk**: limit the likelihood and impact of a privacy breach. For instance, in Tor, compromising one relay does not provide any sensitive information about users' browsing habits. If one compromises the entry node, one cannot learn the destinations of the communications, only the middle nodes of the circuits; and if one compromises the exit node, one cannot learn the origin of the communication.

To minimise both trust and risk, privacy experts typically design systems in accordance with the following strategies:

- *Minimise Collection*: whenever possible, limit the capture and storage of data in the system.
- *Minimise Disclosure*: whenever possible, constrain the flow of information to parties other than the entity to whom the data relates. This refers both to direct flows between senders and receivers, and to indirect flows, e.g., use of control techniques to limit the information available when publishing or querying a dataset.
- *Minimise Replication*: whenever possible, limit the number of entities where data are stored or processed in the clear.
- *Minimise Centralization*: whenever possible, avoid a single point of failure regarding the privacy properties in the system.
- *Minimise Linkability*: whenever possible, limit the capability of the adversary to link data.
- *Minimise Retention*: whenever possible, limit the amount of time information is stored.

Implementing these strategies at first may seem incompatible with maintaining the integrity of the system. For instance, if no information is disclosed or collected, how can one make sure that no entity is abusing the system? If there is no central authority, how can one make sure that authentication and authorisation work as expected? This is where privacy technologies come into play. They enable the design of systems where as little information as possible is revealed to parties other than the ones to which the information relates, and in which there is a minimum need to trust providers or other users in order to preserve the privacy of sensitive information while still pertaining integrity and allowing information exchange.

In order to decide which privacy technology is most adequate to build a system, a first step is to identify the data flows that should be minimised; i.e., those that move data to entities to whom the data do not relate. The second step is to identify the minimal set of data that needs to be transferred to those entities. To identify the minimum required information that needs to be transferred, the designer should attempt to keep as much data as possible out of reach of those entities without harming the functionality of the system. Strategies to minimise unnecessary information flow (based mainly on the technologies introduced throughout Section. 1) are:

- *Keep the data local*: perform any computation on sensitive data on the user side, and only transmit the result of the operation. Additional information, such as zero-knowledge proofs or commitments, may be needed to guarantee the correctness of the operations.
- *Encrypt the data*: encrypt the data locally and send only the encrypted version to other entities. If any operations on the data are needed, see the next point.
- *Use privacy-preserving cryptographic protocols*: process data locally to obtain inputs to a protocol in which, by interacting with the untrusted entities using one of the protocols introduced in the previous sections, the user can obtain or prove information while limiting the information made available to those entities. For instance, using anonymous credentials for authentication without revealing the identity or even the value of an attribute, or using private information retrieval to perform a search on a database without revealing the query to the database holder.
- *Obfuscate the data*: use techniques to control inference to process the data locally and only send the perturbed version to the untrusted entity.
- *Anonymise the data*: process the data locally to remove identifiable information and send it to the untrusted party via an anonymous channel.

By seeking minimisation of trust and using the above techniques, system designers are bound to collect, process and retain fewer data than with other strategies based on compliance with regulation. We recognise that many systems and applications cannot be built without collecting some user-related data. For these cases, designers must take into account the privacy technologies outlined in Section. 2 and Section. 3. These techniques, while requiring trust, help minimise the risk of a breach and, if the breach happens, minimise the impact that the disclosure of data may have for the users.

Privacy evaluation. Once privacy technologies or end-to-end systems have been designed, it is important to conduct a privacy evaluation. This evaluation has the goal of quantifying the level of privacy that the technology, and respectively the system, can provide.

For privacy technologies based on cryptographic primitives, the privacy evaluation is typically covers the cryptographic proofs that ensure that only the intended information is leaked by the operations. On the contrary, for privacy techniques based on obfuscation, it is necessary to carry out an analysis to validate that the combination of techniques provides the desired level of privacy.

A systematic privacy evaluation typically consists of the following steps. First, one needs to model the privacy-preserving mechanism as a probabilistic transformation. This establishes the probability that, given an input, the privacy mechanism returns a given output. Second, one needs to establish the threat model, i.e., what the adversary can see and what is her prior knowledge. Third, assuming that the adversary knows the mechanism, consider how he would annul the effect of the privacy mechanism. This usually entails either doing an analysis of the probability distributions, or using inference techniques such as machine learning to compute what the adversary can learn.

At the end of the process, one usually has a distribution describing the probability that the adversary infers each of the possible inputs. This probability distribution is then used as input to a privacy metric that captures the inference capability of the adversary. We refer the reader to the survey by Wagner and Eckhoff for a comprehensive description of privacy metrics in the literature [162].

CONCLUSIONS

Protecting privacy, as we have described in this knowledge area, is not limited to guaranteeing the confidentiality of information. It also requires means to help users understand the extent to which their information is available online, and mechanisms to enable users to exercise control over this

information. We have described techniques to realise these three privacy conceptions, emphasising the adversarial model in which they operate, as well as providing guidelines to combine these techniques in order to build end-to-end privacy-preserving systems.

Preserving privacy and online rights is not only important for individuals, it is essential to support democratic societies. The deployment of privacy technologies is key to allow users free access to content, and freedom of speech. Of equal importance is to avoid that any entity gaining a disproportionate amount of information about individuals or groups, in order to prevent manipulation and abuse that could damage democratic values.

REFERENCES

[1] U. G. Assembly, "Universal declaration of human rights," December 1948, art. 12, Available at: http://www.unhchr.ch/udhr/lang/eng.pdf.

[2] S. D. Warren and L. D. Brandeis, "The right to privacy," in *Ethical Issues in the Use of Computers*, D. G. Johnson and J. W. Snapper, Eds. Wadsworth Publ. Co., 1985.

[3] A. Rouvroy and Y. Poullet, "The right to informational self-determination and the value of self-development: Reassessing the importance of privacy for democracy," in *Reinventing Data Protection?*, S. Gutwirth, Y. Poullet, P. De Hert, C. de Terwangne, and S. Nouwt, Eds. Springer Netherlands, 2009.

[4] A. F. Westin, *Privacy and Freedom*. Atheneum, 1967.

[5] P. E. Agre and M. Rotenberg, Eds., *Technology and Privacy: The New Landscape*. MIT Press, 1998.

[6] European Commission, "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*, 2016.

[7] G. Danezis and S. Gürses, "A critical review of 10 years of privacy technology," *Surveillance Cultures: A Global Surveillance Society?*, 2010.

[8] S. Gürses and C. Diaz, "Two tales of privacy in online social networks," *IEEE Security and Privacy*, vol. 11, no. 3, pp. 29–37, 2013.

[9] N. Smart, "Cryptography – knowledge area," https://www.cybok.org/media/downloads/Cryptography_KA_-_Issue_1.0_July_2018.pdf, 2018.

[10] G. Acar, C. Eubank, S. Englehardt, M. Juárez, A. Narayanan, and C. Díaz, "The web never forgets: Persistent tracking mechanisms in the wild," in *ACM SIGSAC Conference on Computer and Communications Security*, 2014.

[11] N. Borisov, I. Goldberg, and E. A. Brewer, "Off-the-record communication, or, why not to use PGP," in *WPES*. ACM, 2004.

[12] J. Camenisch and A. Lysyanskaya, "An efficient system for non-transferable anonymous credentials with optional anonymity revocation," in *Advances in Cryptology - EUROCRYPT*, 2001.

[13] D. Chaum, "Blind signatures for untraceable payments," in *CRYPTO*, 1982.

[14] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Found. Trends Theor. Comput. Sci.*, 2014.

[15] M. Marlinspike, "Advanced cryptographic ratcheting (blog)," https://whispersystems.org/blog/advanced-ratcheting/, 2013.

[16] A. Narayanan and V. Shmatikov, "De-anonymizing social networks," in *IEEE Symposium on Security and Privacy (S&P 2009)*, 2009.

[17] F. Shirazi, M. Simeonovski, M. R. Asghar, M. Backes, and C. Díaz, "A survey on routing in anonymous communication protocols," *ACM Comput. Surv.*, 2018.

[18] D. Adrian, K. Bhargavan, Z. Durumeric, P. Gaudry, M. Green, J. A. Halderman, N. Heninger, D. Springall, E. Thomé, L. Valenta, B. VanderSloot, E. Wustrow, S. Zanella-Béguelin, and P. Zimmermann, "Imperfect forward secrecy: how diffie-hellman fails in practice," *Commun.*

ACM, 2019.
[19] "The Transport Layer Security (TLS) Protocol Version 1.3," RFC 8446, 2018.
[20] M. W. Lucas, *PGP & GPG: Email for the Practical Paranoid*, 1st ed. No Starch Press, 2006.
[21] C. Alexander and I. Goldberg, "Improved user authentication in off-the-record messaging," in *WPES*, 2007.
[22] N. Unger and I. Goldberg, "Improved strongly deniable authenticated key exchanges for secure messaging," *PoPETs*, 2018.
[23] O. W. Systems, "Signal protocol library for java/android," https://github.com/signalapp/libsignal-protocol-java.
[24] K. Cohn-Gordon, C. J. F. Cremers, B. Dowling, L. Garratt, and D. Stebila, "A formal security analysis of the signal messaging protocol," in *IEEE European Symposium on Security and Privacy, EuroS&P*, 2017.
[25] E. Hesamifard, H. Takabi, M. Ghasemi, and R. N. Wright, "Privacy-preserving machine learning as a service," *PoPETs*, 2018.
[26] D. X. Song, D. A. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *IEEE Symposium on Security and Privacy*, 2000.
[27] N. Borisov, G. Danezis, and I. Goldberg, "DP5: A private presence service," *PoPETs*, 2015.
[28] P. Mittal, F. G. Olumofin, C. Troncoso, N. Borisov, and I. Goldberg, "Pir-tor: Scalable anonymous communication using private information retrieval," in *20th USENIX Security Symposium*, 2011.
[29] R. R. Toledo, G. Danezis, and I. Goldberg, "Lower-cost $\in$-private information retrieval," *PoPETs*, 2016.
[30] W. Aiello, Y. Ishai, and O. Reingold, "Priced oblivious transfer: How to sell digital goods," in *Advances in Cryptology - EUROCRYPT 2001*, 2001.
[31] J. Camenisch, M. Dubovitskaya, and G. Neven, "Unlinkable priced oblivious transfer with rechargeable wallets," in *Financial Cryptography and Data Security*, 2010.
[32] P. Mishra, R. Poddar, J. Chen, A. Chiesa, and R. A. Popa, "Oblix: An efficient oblivious search index," in *2018 IEEE Symposium on Security and Privacy, SP 2018*, 2018.
[33] S. Sasy, S. Gorbunov, and C. W. Fletcher, "Zerotrace : Oblivious memory primitives from intel SGX," in *25th Annual Network and Distributed System Security Symposium, NDSS*, 2018.
[34] R. A. Popa, C. M. S. Redfield, N. Zeldovich, and H. Balakrishnan, "Cryptdb: protecting confidentiality with encrypted query processing," in *SOSP*, 2011.
[35] F. Kerschbaum, "Frequency-hiding order-preserving encryption," in *ACM Conference on Computer and Communications Security*, 2015.
[36] K. Lewi and D. J. Wu, "Order-revealing encryption: New constructions, applications, and lower bounds," in *ACM Conference on Computer and Communications Security*, 2016.
[37] V. Bindschaedler, P. Grubbs, D. Cash, T. Ristenpart, and V. Shmatikov, "The tao of inference in privacy-protected databases," *PVLDB*, 2018.
[38] P. Grubbs, M. Lacharité, B. Minaud, and K. G. Paterson, "Pump up the volume: Practical database reconstruction from volume leakage on range queries," in *ACM Conference on Computer and Communications Security*, 2018.
[39] P. Grubbs, T. Ristenpart, and V. Shmatikov, "Why your encrypted database is not secure," in *HotOS*, 2017.
[40] M. Naveed, S. Kamara, and C. V. Wright, "Inference attacks on property-preserving encrypted databases," in *ACM Conference on Computer and Communications Security*, 2015.
[41] H. Corrigan-Gibbs and D. Boneh, "Prio: Private, robust, and scalable computation of aggregate statistics," in *NSDI*, 2017.
[42] L. Melis, G. Danezis, and E. D. Cristofaro, "Efficient private statistics with succinct sketches," in *NDSS*, 2016.
[43] D. W. Archer, D. Bogdanov, Y. Lindell, L. Kamm, K. Nielsen, J. I. Pagter, N. P. Smart, and R. N. Wright, "From keys to databases - real-world applications of secure multi-party computation," *Comput. J.*, 2018.

[44] Á. Kiss, J. Liu, T. Schneider, N. Asokan, and B. Pinkas, "Private set intersection for unequal set sizes with mobile applications," *PoPETs*, 2017.
[45] M. Nagy, E. D. Cristofaro, A. Dmitrienko, N. Asokan, and A. Sadeghi, "Do I know you?: efficient and privacy-preserving common friend-finder protocols and applications," in *Annual Computer Security Applications Conference, ACSAC*, 2013.
[46] S. Nagaraja, P. Mittal, C. Hong, M. Caesar, and N. Borisov, "Botgrep: Finding P2P bots with structured graph analysis," in *19th USENIX Security Symposium*, 2010.
[47] P. Baldi, R. Baronio, E. D. Cristofaro, P. Gasti, and G. Tsudik, "Countering GATTACA: efficient and secure testing of fully-sequenced human genomes," in *ACM Conference on Computer and Communications Security, CCS*, 2011.
[48] E. D. Cristofaro and G. Tsudik, "Practical private set intersection protocols with linear complexity," in *Financial Cryptography*, 2010.
[49] M. J. Freedman, K. Nissim, and B. Pinkas, "Efficient private matching and set intersection," in *EUROCRYPT*, 2004.
[50] B. Pinkas, T. Schneider, C. Weinert, and U. Wieder, "Efficient circuit-based PSI via cuckoo hashing," in *EUROCRYPT (3)*, 2018.
[51] J. Balasch, A. Rial, C. Troncoso, B. Preneel, I. Verbauwhede, and C. Geuens, "Pretp: Privacy-preserving electronic toll pricing," in *USENIX Security Symposium*, 2010.
[52] A. Rial and G. Danezis, "Privacy-preserving smart metering," in *WPES*, 2011.
[53] H. Corrigan-Gibbs, D. Boneh, and D. Mazières, "Riposte: An anonymous messaging system handling millions of users," in *2015 IEEE Symposium on Security and Privacy, SP*, 2015.
[54] C. Hazay and K. Nissim, "Efficient set operations in the presence of malicious adversaries," in *Public Key Cryptography*, 2010.
[55] J. Camenisch, "Concepts around privacy-preserving attribute-based credentials - making authentication with anonymous credentials practical," in *Privacy and Identity Management*, ser. IFIP Advances in Information and Communication Technology, 2013.
[56] M. Koning, P. Korenhof, G. Alpár, and J.-H. Hoepman, "The ABC of ABC: An analysis of attribute-based credentials in the light of data protection, privacy and identity," *HotPETS*, 2014.
[57] J. Camenisch, S. Hohenberger, M. Kohlweiss, A. Lysyanskaya, and M. Meyerovich, "How to win the clonewars: efficient periodic n-times anonymous authentication," in *13th ACM Conference on Computer and Communications Security, CCS*, 2006.
[58] R. Henry and I. Goldberg, "Thinking inside the BLAC box: smarter protocols for faster anonymous blacklisting," in *12th annual ACM Workshop on Privacy in the Electronic Society, WPES*, 2013.
[59] P. P. Tsang, M. H. Au, A. Kapadia, and S. W. Smith, "Blacklistable anonymous credentials: blocking misbehaving users without ttps," in *ACM Conference on Computer and Communications Security, CCS*, 2007.
[60] J. Camenisch, M. Drijvers, and J. Hajny, "Scalable revocation scheme for anonymous credentials based on n-times unlinkable proofs," in *ACM on Workshop on Privacy in the Electronic Society, WPES*, 2016.
[61] IBM, "Identity mixer," https://www.zurich.ibm.com/identity_mixer/.
[62] Privacy by Design Foundation, "IRMA: I Reveal My Attributes," https://privacybydesign.foundation/irma-explanation/.
[63] E. Ben-Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza, "Zerocash: Decentralized anonymous payments from bitcoin," in *IEEE Symposium on Security and Privacy, SP*, 2014.
[64] I. Miers, C. Garman, M. Green, and A. D. Rubin, "Zerocoin: Anonymous distributed e-cash from bitcoin," in *IEEE Symposium on Security and Privacy, SP*, 2013, pp. 397–411.
[65] E. Ben-Sasson, A. Chiesa, E. Tromer, and M. Virza, "Succinct non-interactive zero knowledge for a von neumann architecture," in *23rd USENIX Security Symposium*, 2014.
[66] Y.-A. de Montjoye, C. A. Hidalgo, M. Verleysen, and V. D. Blondel, "Unique in the crowd: The

privacy bounds of human mobility," *Scientific Reports*, 2013.
[67] N. Homer, S. Szelinger, M. Redman, D. Duggan, W. Tembe, J. Muehling, J. V. Pearson, D. A. Stephan, S. F. Nelson, and D. W. Craig, "Resolving individuals contributing trace amounts of dna to highly complex mixtures using high-density snp genotyping microarrays," *PLOS Genetics*, 2008.
[68] P. Golle and K. Partridge, "On the anonymity of home/work location pairs," in *Pervasive Computing*, 2009.
[69] L. Sweeney, "Achieving k-anonymity privacy protection using generalization and suppression," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 2002.
[70] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam, "l-diversity: Privacy beyond k-anonymity," in *International Conference on Data Engineering, ICDE*, 2006.
[71] N. Li, T. Li, and S. Venkatasubramanian, "t-closeness: Privacy beyond k-anonymity and l-diversity," in *ICDE*. IEEE Computer Society, 2007.
[72] K. El Emam and F. K. Dankar, "Protecting Privacy Using k-Anonymity," *Journal of the American Medical Informatics Association*, 2008.
[73] B. Gedik and L. Liu, "Protecting location privacy with personalized k-anonymity: Architecture and algorithms," *IEEE Transactions on Mobile Computing*, 2008.
[74] M. Stegelmann and D. Kesdogan, "Gridpriv: A smart metering architecture offering k-anonymity," in *Trust, Security and Privacy in Computing and Communications (TrustCom)*, 2012.
[75] E. Balsa, C. Troncoso, and C. Díaz, "OB-PWS: obfuscation-based private web search," in *IEEE Symposium on Security and Privacy*, 2012.
[76] A. Gadotti, F. Houssiau, L. Rocher, and Y. de Montjoye, "When the signal is in the noise: The limits of diffix's sticky noise," *CoRR*, vol. abs/1804.06752, 2018.
[77] B. Hoh, M. Gruteser, H. Xiong, and A. Alrabady, "Preserving privacy in GPS traces via uncertainty-aware path cloaking," in *ACM Conference on Computer and Communications Security, CCS*, 2007.
[78] K. Chatzikokolakis, C. Palamidessi, and M. Stronati, "A predictive differentially-private mechanism for mobility traces," in *Privacy Enhancing Technologies - 14th International Symposium, PETS*, 2014.
[79] R. Chow and P. Golle, "Faking contextual data for fun, profit, and privacy," in *ACM Workshop on Privacy in the Electronic Society, WPES*, 2009.
[80] S. T. Peddinti and N. Saxena, "On the privacy of web search based on query obfuscation: A case study of trackmenot," in *10th International Symposium Privacy Enhancing Technologies PETS*, 2010.
[81] J. L. Raisaro, J. Troncoso-Pastoriza, M. Misbach, J. S. Sousa, S. Pradervand, E. Missiaglia, O. Michielin, B. Ford, and J.-P. Hubaux, "Medco: Enabling secure and privacy-preserving exploration of distributed clinical and genomic data," *IEEE/ACM transactions on computational biology and bioinformatics*, 2018.
[82] J. J. Kim, "A method for limiting disclosure in microdata based on random noise and transformation," in *Proceedings of the section on survey research methods*. American Statistical Association, 1986.
[83] W. E. Yancey, W. E. Winkler, and R. H. Creecy, "Disclosure risk assessment in perturbative microdata protection," in *Inference Control in Statistical Databases, From Theory to Practice*, 2002.
[84] C. Dwork, "Differential privacy," in *ICALP (2)*, ser. Lecture Notes in Computer Science, 2006.
[85] S. Oya, C. Troncoso, and F. Pérez-González, "Is geo-indistinguishability what you are looking for?" in *WPES*, 2017.
[86] S. Meiser and E. Mohammadi, "Tight on budget?: Tight bounds for r-fold approximate differential privacy," in *ACM SIGSAC Conference on Computer and Communications Security*, 2018.
[87] K. Chatzikokolakis, M. E. Andrés, N. E. Bordenabe, and C. Palamidessi, "Broadening the scope

of differential privacy using metrics," in *Privacy Enhancing Technologies*, 2013.
[88] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *ACM Conference on Computer and Communications Security*, 2017.
[89] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Geo-indistinguishability: differential privacy for location-based systems," in *ACM Conference on Computer and Communications Security*, 2013.
[90] J. M. Abowd, "The U.S. census bureau adopts differential privacy," in *KDD*, 2018.
[91] G. Danezis and C. Diaz, "A survey of anonymous communication channels," Microsoft Research, Tech. Rep. MSR-TR-2008-35, 2008.
[92] R. Dingledine, N. Mathewson, and P. F. Syverson, "Tor: The second-generation onion router," in *13th USENIX Security Symposium*, 2004.
[93] A. Johnson, C. Wacek, R. Jansen, M. Sherr, and P. F. Syverson, "Users get routed: traffic correlation on tor by realistic adversaries," in *ACM SIGSAC Conference on Computer and Communications Security*, 2013.
[94] D. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Commun. ACM*, 1981.
[95] A. M. Piotrowska, J. Hayes, T. Elahi, S. Meiser, and G. Danezis, "The loopix anonymity system," in *USENIX Security Symposium, USENIX Security*, 2017.
[96] G. Danezis and I. Goldberg, "Sphinx: A compact and provably secure mix format," in *IEEE Symposium on Security and Privacy (S&P)*, 2009.
[97] G. Acar, M. Juárez, N. Nikiforakis, C. Diaz, S. Gürses, F. Piessens, and B. Preneel, "Fpdetective: dusting the web for fingerprinters," in *2013 ACM SIGSAC Conference on Computer and Communications Security*, 2013.
[98] A. Vastel, P. Laperdrix, W. Rudametkin, and R. Rouvoy, "FP-STALKER: tracking browser fingerprint evolutions," in *IEEE Symposium on Security and Privacy, SP*, 2018.
[99] N. Nikiforakis, A. Kapravelos, W. Joosen, C. Kruegel, F. Piessens, and G. Vigna, "Cookieless monster: Exploring the ecosystem of web-based device fingerprinting," in *IEEE Symposium on Security and Privacy*, 2013.
[100] P. Laperdrix, B. Baudry, and V. Mishra, "Fprandom: Randomizing core browser objects to break advanced device fingerprinting techniques," in *ESSoS*, 2017.
[101] F. Roesner, T. Kohno, and D. Wetherall, "Detecting and defending against third-party tracking on the web," in *USENIX Symposium on Networked Systems Design and Implementation, NSDI*, 2012.
[102] L. Olejnik, M. Tran, and C. Castelluccia, "Selling off user privacy at auction," in *Network and Distributed System Security Symposium, NDSS*, 2014.
[103] H. Zang and J. Bolot, "Anonymization of location data does not work: a large-scale measurement study," in *Conference on Mobile Computing and Networking, MOBICOM*, 2011, pp. 145–156.
[104] J. Pang and Y. Zhang, "Deepcity: A feature learning framework for mining location check-ins," in *Conference on Web and Social Media, ICWSM*, 2017.
[105] S. Gambs, M. Killijian, and M. N. del Prado Cortez, "Show me how you move and I will tell you who you are," *Trans. Data Privacy*, 2011.
[106] G. Zhong, I. Goldberg, and U. Hengartner, "Louis, lester and pierre: Three protocols for location privacy," in *Privacy Enhancing Technologies*, 2007.
[107] A. Narayanan, N. Thiagarajan, M. Lakhani, M. Hamburg, and D. Boneh, "Location privacy via private proximity testing," in *NDSS*, 2011.
[108] Z. Lin, D. F. Kune, and N. Hopper, "Efficient private proximity testing with GSM location sketches," in *Financial Cryptography*, 2012.
[109] M. Li, K. Sampigethaya, L. Huang, and R. Poovendran, "Swing & swap: user-centric approaches towards maximizing location privacy," in *WPES*, 2006.

[110] J. Krumm, “Inference attacks on location tracks,” in *Pervasive*, 2007.
[111] M. Gruteser and D. Grunwald, “Anonymous usage of location-based services through spatial and temporal cloaking,” in *MobiSys*, 2003.
[112] J. Krumm, “Realistic driving trips for location privacy,” in *Pervasive*, 2009.
[113] A. Acquisti, I. Adjerid, and L. Brandimarte, “Gone in 15 seconds: The limits of privacy transparency and control,” *IEEE Security Privacy*, 2013.
[114] M. Madejski, M. Johnson, and S. M Bellovin, “The failure of online social network privacy settings,” Tech Report CUCS-010-11, Columbia University, 2011.
[115] H. Harkous, K. Fawaz, R. Lebret, F. Schaub, K. G. Shin, and K. Aberer, “Polisis: Automated analysis and presentation of privacy policies using deep learning,” in *USENIX Security Symposium*, 2018.
[116] J. Bonneau, J. Anderson, and L. Church, “Privacy suites: shared privacy for social networks,” in *Symposium on Usable Privacy and Security, SOUPS*, 2009.
[117] J. Lin, B. Liu, N. M. Sadeh, and J. I. Hong, “Modeling users’ mobile app privacy preferences: Restoring usability in a sea of permission settings,” in *Symposium on Usable Privacy and Security, SOUPS*, 2014.
[118] Q. Ismail, T. Ahmed, K. Caine, A. Kapadia, and M. K. Reiter, “To permit or not to permit, that is the usability question: Crowdsourcing mobile apps’ privacy permission settings,” *PoPETs*, 2017.
[119] World Wide Web Consortium (W3C), “Platform for Privacy Preferences (P3P),” https://www.w3.org/P3P, 2007.
[120] L. Cranor, M. Langheinrich, and M. Marchiori, “A p3p preference exchange language 1.0 (appel 1.0),” World Wide Web Consortium, Working Draft WD-P3P-preferences-20020415, 2002.
[121] J. Byun and N. Li, “Purpose based access control for privacy protection in relational database systems,” *VLDB J.*, 2008.
[122] G. Karjoth, M. Schunter, and M. Waidner, “Platform for enterprise privacy practices: Privacy-enabled management of customer data,” in *Privacy Enhancing Technologies, PET*, 2002.
[123] S. Wilson, F. Schaub, R. Ramanath, N. M. Sadeh, F. Liu, N. A. Smith, and F. Liu, “Crowdsourcing annotations for websites’ privacy policies: Can it really work?” in *WWW*, 2016.
[124] H. Liu, P. Maes, and G. Davenport, “Unraveling the taste fabric of social networks,” *Int. J. Semantic Web Inf. Syst.*, 2006.
[125] Y. Wang, P. G. Leon, K. Scott, X. Chen, A. Acquisti, and L. F. Cranor, “Privacy nudges for social media: an exploratory facebook study,” in *International World Wide Web Conference, WWW*, 2013, pp. 763–770.
[126] A. Acquisti, I. Adjerid, R. Balebako, L. Brandimarte, L. F. Cranor, S. Komanduri, P. G. Leon, N. Sadeh, F. Schaub, M. Sleeper, Y. Wang, and S. Wilson, “Nudges for privacy and security: Understanding and assisting users’ choices online,” *ACM Comput. Surv.*, 2017.
[127] T. Paul, D. Puscher, and T. Strufe, “Improving the usability of privacy settings in facebook,” *CoRR*, vol. abs/1109.6046, 2011.
[128] D. Biswas and V. Niemi, “Transforming privacy policies to auditing specifications,” in *HASE*, 2011.
[129] D. Froelicher, P. Egger, J. S. Sousa, J. L. Raisaro, Z. Huang, C. Mouchet, B. Ford, and J. Hubaux, “UnLynx: A decentralized system for privacy-conscious data sharing,” *PoPETs*, 2017.
[130] C. A. Neff, “A verifiable secret shuffle and its application to e-voting,” in *ACM Conference on Computer and Communications Security*. ACM, 2001.
[131] S. Khattak, T. Elahi, L. Simon, C. M. Swanson, S. J. Murdoch, and I. Goldberg, “Sok: Making sense of censorship resistance systems,” *PoPETs*, 2016.
[132] D. J. Solove, ““I’ve Got Nothing to Hide" and other misunderstandings of privacy,” *San Diego Law Review*, 2007.
[133] D. Boneh and P. Golle, “Almost entirely correct mixing with applications to voting,” in *ACM*

Conference on Computer and Communications Security, CCS, 2002.
[134] M. Jakobsson, A. Juels, and R. L. Rivest, "Making mix nets robust for electronic voting by randomized partial checking," in *USENIX Security Symposium*, 2002.
[135] A. Fujioka, T. Okamoto, and K. Ohta, "A practical secret voting scheme for large scale elections," in *AUSCRYPT*, 1992.
[136] O. Baudron, P. Fouque, D. Pointcheval, J. Stern, and G. Poupard, "Practical multi-candidate election system," in *PODC*, 2001.
[137] A. Juels, D. Catalano, and M. Jakobsson, "Coercion-resistant electronic elections," in *Towards Trustworthy Elections*, 2010.
[138] M. R. Clarkson, S. Chong, and A. C. Myers, "Civitas: Toward a secure voting system," in *2008 IEEE Symposium on Security and Privacy (S&P)*, 2008.
[139] J. Berg, "The dark side of e-petitions? exploring anonymous signatures," 2017.
[140] C. Diaz, E. Kosta, H. Dekeyser, M. Kohlweiss, and G. Nigusse, "Privacy preserving electronic petitions," *Identity in the Information Society*, 2008.
[141] A. Sonnino, M. Al-Bassam, S. Bano, and G. Danezis, "Coconut: Threshold issuance selective disclosure credentials with applications to distributed ledgers," *arXiv preprint arXiv:1802.07344*, 2018.
[142] M. C. Tschantz, S. Afroz, anonymous, and V. Paxson, "Sok: Towards grounding censorship circumvention in empiricism," in *IEEE Symposium on Security and Privacy, SP*, 2016.
[143] R. Newman, "The Church of Scientology vs. anon.penet.fi," http://www.spaink.net/cos/rnewman/anon/penet.html, 1997.
[144] R. Anderson, "The eternity service," in *Proceedings of Pragocrypt '96*, 1996.
[145] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong, "Freenet: A distributed anonymous information storage and retrieval system," in *Workshop on Design Issues in Anonymity and Unobservability*, 2000.
[146] G. Tian, Z. Duan, T. Baumeister, and Y. Dong, "A traceback attack on freenet," in *INFOCOM*, 2013.
[147] S. Roos, F. Platzer, J. Heller, and T. Strufe, "Inferring obfuscated values in freenet," in *NetSys*, 2015.
[148] B. Levine, M. Liberatore, B. Lynn, and M. Wright, "Statistical detection of downloaders in freenet," in *IWPE@SP*, 2017.
[149] M. Waldman and D. Mazières, "Tangler: a censorship-resistant publishing system based on document entanglements," in *ACM Conference on Computer and Communications Security*, 2001.
[150] H. M. Moghaddam, B. Li, M. Derakhshani, and I. Goldberg, "Skypemorph: protocol obfuscation for tor bridges," in *ACM Conference on Computer and Communications Security*, 2012.
[151] 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 *ACM Conference on Computer and Communications Security*, 2012.
[152] A. Houmansadr, C. Brubaker, and V. Shmatikov, "The parrot is dead: Observing unobservable network communications," in *IEEE Symposium on Security and Privacy*, 2013.
[153] C. Brubaker, A. Houmansadr, and V. Shmatikov, "Cloudtransport: Using cloud storage for censorship-resistant networking," in *Privacy Enhancing Technologies*, 2014.
[154] D. Fifield, C. Lan, R. Hynes, P. Wegmann, and V. Paxson, "Blocking-resistant communication through domain fronting," *PoPETs*, 2015.
[155] R. McPherson, A. Houmansadr, and V. Shmatikov, "Covertcast: Using live streaming to evade internet censorship," *PoPETs*, 2016.
[156] T. T. Project.
[157] J. Karlin, D. Ellard, A. W. Jackson, C. E. Jones, G. Lauer, D. Mankins, and W. T. Strayer, "Decoy routing: Toward unblockable internet communication," in *FOCI*, 2011.
[158] E. Wustrow, S. Wolchok, I. Goldberg, and J. A. Halderman, "Telex: Anticensorship in the net-

work infrastructure," in *USENIX Security Symposium*, 2011.

[159] C. Bocovich and I. Goldberg, "Secure asymmetry and deployability for decoy routing systems," *PoPETs*, 2018.

[160] M. Nasr, H. Zolfaghari, and A. Houmansadr, "The waterfall of liberty: Decoy routing circumvention that resists routing attacks," in *ACM Conference on Computer and Communications Security*, 2017.

[161] S. Gürses, C. Troncoso, and C. Diaz, "Engineering privacy by design reloaded," in *Amsterdam Privacy Conference*, 2015.

[162] I. Wagner and D. Eckhoff, "Technical privacy metrics: A systematic survey," *ACM Comput. Surv.*, 2018.