

Formal Models for Privacy

Alan Abe and Andrew Simpson
Department of Computer Science, University of Oxford
Wolfson Building, Parks Road, Oxford OX1 3QD
United Kingdom

Alan.Abe@cs.ox.ac.uk, Andrew.Simpson@cs.ox.ac.uk

ABSTRACT

The concept of privacy is becoming increasingly important in all of our lives. Unfortunately, however, it is a rather nebulous concept; further, many claim that they consider privacy to be important, yet undertake behaviour that would suggest otherwise — the so-called *privacy paradox*. As technology becomes more pervasive, the need for assurances that individuals' privacy is not compromised by that technology necessarily increases. In this paper, we argue that formal methods have a role to play in helping to provide assurances of privacy in a variety of contexts. As an illustration, we give consideration to a particular scenario: that of data sharing.

Keywords

Formal methods; privacy; Z notation

1. INTRODUCTION

The ubiquity of computing technologies that can gather, store and share personal information has reinvigorated debates surrounding an age-old societal dilemma — balancing the notions of *individual privacy* and the *common good*, which can emerge from the knowledge gained by analysing information based not on the individual, but on the aggregation of information. For example, aggregating information collected in the course of patient care can give rise to study populations of sufficient size and heterogeneity in which well-designed secondary uses have the potential to investigate research questions that could not be pursued via the traditional route of randomised controlled trials. For it is through the understanding gained from such evidence-based knowledge of diseases and interventions that policies and strategies to afford effective protection to the health of communities and improve the quality of human life can emerge. However, there is the potential for harm should the inappropriate disclosure or use of such information compromise an individual's privacy.

Unfortunately, there is a great deal of 'fuzziness' surrounding privacy. First, defining the term is notoriously hard [25].

Second, individuals will often claim that privacy is important to them, yet, on the other hand, will exhibit behaviours that would indicate that the opposite is true: the so-called *privacy paradox* [20]. Third, the disconnect between high-level requirements, rules and guidelines and their low-level implementations can be significant — meaning that it is often difficult for software engineers and database administrators to be confident that their processes and systems behave as intended. Fourth, there are often trade-offs — between privacy and utility, for example — to be made. It is clear, therefore, that the appropriate use of formal methods has much to offer in this context.

In [29], Tshantz and Wing argue that “privacy raises new challenges, and thus new research opportunities, for the formal methods community”. “Privacy-specific needs” — such as statistical / quantitative reasoning and conflicting requirements in the context of trustworthy computing — are identified, with “traditional tools of the trade” — such as formal policy languages, abstraction and refinement, and code-level analysis — suggested as solutions. They further argue: “It is our responsibility as scientists and engineers to understand what can or cannot be done from a technical point of view on privacy Otherwise, society may end up in a situation where privacy regulations put into place are technically infeasible to meet” [29]. We pick up that baton and give consideration to how formal methods can be applied to the modelling and analysis of privacy in the context of data-sharing. (For clarification, by ‘formal methods’ we mean the application of techniques such as Z [14] and B [1], as opposed to the consideration of models for protecting privacy such as *k*-Anonymity [28] and Differential Privacy [10].) There is a long tradition of formal methods being applied to ‘real world’ problems (see, for example, the survey of [30]); this contribution is in that spirit.

To support reasoning about privacy, we consider a broad range of system–environment interactions that can cause a system to transition to an unwanted state. We use Z [14] to describe the model and ProZ [21] to analyse it. Z semantics, which are based on logic and set theory, enable privacy to be modelled from the perspective of “data in a system” [26]. ProZ mechanically validates Z specifications: preservation of an invariant (as related to privacy) and the refinement of one specification (of privacy) by another. Thus, our metric for privacy is a binary result (rather than being probabilistic).

We aim to address the gap between the high-level requirements (derived from, for example, privacy laws and regulations) that are described in natural language and their implementations. We utilise the UK's Data Protection Act

1998 (DPA)¹ and the approaches to protecting data privacy that have been proposed in the literature to inform our interpretation of privacy. It is important to note that we do not define privacy *per se*; rather, we show how a formal approach can be used to characterise it.

2. MOTIVATION

Advances in IT have motivated the promulgation of laws and regulations aimed at strengthening privacy protections for the handling of personal data. Implications for secondary uses (in particular, in the area of medical research) due to the aforementioned DPA in the UK and the Privacy Rule under the Health Insurance Portability Accountability Act of 1996 (HIPAA)² in the USA have been debated (see, for example, [8]). Despite stricter privacy protections, the DPA and HIPAA do allow for circumstances in which personal data may be used or shared without the consent of individuals to include the appropriate *de-identification* of data. The DPA regulates the processing of personal data in their entirety, of which data de-identification is just one aspect. The implication is that the scope of a data-owner’s obligation to privacy protection is not limited to the de-identified data themselves, but also includes their processing.

De-identification is the path envisioned by many to satisfy privacy requirements, and numerous privacy-preserving methods founded on varying conceptualisations of *privacy* and *utility* of de-identified data have been proposed (see, for example, [13]). Empirical validations of these methodologies are typically based on a high-level abstraction of the underlying systems that would perform the processing — often, centrally by a single data controller. Further, considerations of threats have typically been underpinned by the following characterisation: intruders (characterised as a “(hypothetical) user who ‘misuses’ the outputs” to “disclose information about one of the data subjects” [24]) obtain de-identified data, obtain relevant auxiliary information, and then leverage the auxiliary information to compromise the privacy of data subjects. However, processing of data is often distributed across multiple systems and organisations, and, in some cases, across national boundaries. This complexity can introduce uncertainties as to whether a particular privacy-preserving method can be implemented in a manner that maintains confidentiality and privacy. Instead of attacking the shared de-identified data, an intruder may find it ‘easier’ to attack the systems or people that produce them. Thus, we give consideration to how formal models can help in the data release process.

3. RELATED WORK

Contributions that have applied formal approaches to privacy include: [27], which is concerned with the modelling and verification of privacy-enhancing protocols; [2], which is concerned with detecting and resolving ambiguities in privacy requirements; [12], which is concerned with the verification of privacy analysis; [16], which presents a formal framework for privacy by design [6]; and [15], which presents a typing system for privacy. Logic-based techniques have been applied to, in particular, the Privacy Rule of HIPAA

¹<http://www.legislation.gov.uk/ukpga/1998/29/data.pdf>

²<http://www.hhs.gov/ocr/privacy/>

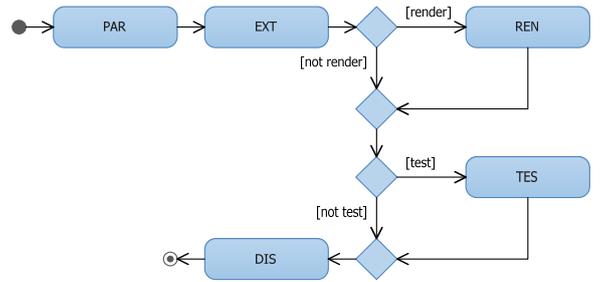


Figure 1: An abstraction of disclosure-processing.

1996 [3, 7, 9]. Elements of a permission (action and resource) for role-based access control (P-RBAC) have been combined with elements of privacy (e.g., purpose, conditions, and obligations) in [18, 19]. Our contribution uses a formal method to characterise and reason about privacy (requirements based on laws and regulations) in the context of its characterisation in terms of, for example, design or code, thereby bridging the gap between abstract notions and concrete representations of privacy.

4. DISCLOSURE-PROCESSING

Our modelling of privacy is motivated by the disclosure of data for secondary uses in which several privacy-preserving methods have been proposed. The processing of personal data begins with their extraction from a source (or sources), and ends with their release in an appropriate form.

We initially abstract disclosure-processing as a unitary system of related processes: we assume processes are instantiated singularly and internally in relation to a data controller. It may be argued that such a unitary system is unrealistic — in that it is unlikely that a relatively simplistic system would be able to effectively or efficiently process large amounts of data, dispersed data, or data involving the collaboration of several data controllers. However, its simplicity allows us to establish an abstraction that we can use to model processing; further, it serves as a starting point for subsequently thinking about more complicated situations.

Our system is comprised of five processes (as per Figure 1):

1. Parameterisation (*PAR*). A data controller determines the parameters that guide each aspect of processing: extraction (*extparam*), rendering (*renparam*), testing (*tesparam*), and dissemination (*disparam*). In certain instances (such as an interactive mode of disclosure), a data controller may permit, on a controlled basis, a data user to determine certain parameters. If auxiliary information is used in the evaluation of data, a data controller creates or obtains the relevant information. The primary inputs are the processing parameters and, where appropriate, auxiliary information that are also its outputs to other processes.
2. Extraction (*EXT*). From a designated source, data that possess certain characteristics or the results of an applied workload are extracted. The primary inputs are the personal data and *extparam* to control the extraction process that may include the location of the data source, the characteristics of the data to be ex-

tracted, the workload to be applied, or the method of extraction. The primary outputs are the personal data that have been extracted (*extdata*) or the results of a workload (extracted information) that has been applied to the personal data.

3. Rendering (*REN*). The extracted data or information may then be transformed via a rendering method into a form determined by a data controller to be appropriate for release. The primary inputs are the *extdata* or extracted information and *renparam* to control the rendering process that may include the method of rendering, the intensity of the method's application, or the data characteristics involved. The primary outputs are the data (*rendata*) or information that have been rendered into a different form.
4. Testing (*TES*). Data or information may be evaluated via the employment of certain tests (pertaining to privacy and/or data utility) as determined by a data controller. Such testing can support a data controller in making a reasoned decision about the release of data for dissemination. The primary inputs are the data (*extdata* or *rendata*) or information (extracted or rendered) to be evaluated, additional data (where appropriate) to support the evaluation (auxiliary information or *extdata*), and *tesparam* to control the testing process that may include the testing methods, the metrics, the data characteristics involved, or the use of other data. The primary outputs are the results of the tests that have been applied to the data or information. Tests of privacy may involve the use of auxiliary information, in particular, to characterise the risk of re-identification based on the likelihood of establishing links between the auxiliary information and the *rendata*; tests of data utility may involve the use of *extdata* in a comparison with the *rendata* in which the same workload is performed on both.
5. Dissemination (denoted *DIS*). Data or information that are considered appropriate for release are then disseminated via a method determined by a data controller. The primary inputs are the data (*extdata* or *rendata*) or information (extracted or rendered) to be disseminated and *disparam* to control the dissemination that may include mode of dissemination or destination for the transmission of the data.

The rendering and testing processes can be repeated until the form in which the data are rendered satisfy criteria for their release. This can be achieved by using different rendering methods, the same method with different parameters, or different evaluation criteria. If it is determined that the rendering of data into an acceptable form is unattainable, processing can be terminated without dissemination.

5. A SCENARIO

Our fictional scenario is motivated by the UK's implementation of a smart meter programme. In assessing the pertinent issues, Brown [5] argues that: "Because smart meters can collect and share detailed information about energy use and hence household life, their impact on privacy has become a high-profile matter of interest to energy and

privacy regulators, and to privacy campaigners, journalists, and members of the public" [5]. It is an appropriate scenario for illustrating our approach for a variety of reasons. First, certain smart data may be considered to be personal data. Second, there is significant potential for secondary uses of the data. Third, it is an active area of privacy research [23].

We consider the disclosure-processing of customers' personal information and utility consumption data collected via smart meters. To this end, we make the following assumptions. First, smart meters are capable of gathering fine-grained information about a customer's consumption of utilities. Second, smart meters support two-way communications: data gathered by a smart meter can be transmitted to a utility provider and a utility provider can send data (e.g., instructions) to the device. Third, it has been determined that smart meter readings are to be handled as personal data. Finally, a customer's personal information and smart meter readings are held in accordance with legal and regulatory requirements.

Customers of Smart Meter Utilities (SMU) can subscribe to one or more of electricity, gas and water. Customers' consumption data are collected via smart meters that are capable of collecting data about consumption by type of utility, by type of device, and by time. Periodically, SMU sends customers consumption information, charges that are associated with their consumption, and the amount to be deducted from their bank accounts (by contractual agreement). Customers can phone SMU's customer service centre to ask questions about offered services, their accounts, or to report problems. For a reported problem, a technician is dispatched to carry out repairs. Marketing information about products and services is sent to customers. There is keen interest from government, academia, non-profit organisations and utility-related businesses to analyse SMU's utility consumption data.

6. THE UK'S DATA PROTECTION ACT

Privacy regulations differ across the globe, typically influenced by political and cultural factors. They are often imposed in reaction to pressures from the populace and thereby strongly influence the practices of data controllers in the handling of personal data. The 'appropriateness' (or lack thereof), concerning privacy measures undertaken by a data controller will often be litigated.

We consider the UK's Data Protection Act (DPA) as the legal framework to guide our work. Privacy, however, is neither defined nor explicitly characterised as such in the DPA. As exemptions from and contraventions to the Act are specified (primarily) in terms of the *data protection principles* (DPPs)³ and provisions related to the rights of data subjects,⁴ it may be argued that the DPPs and rights of data subjects can inform our interpretation of the DPA (in terms of privacy). The DPPs, as guiding principles for data controllers, prescribes that personal data shall be:⁵

1. processed fairly and lawfully;
2. obtained for specified and lawful purposes and further processed only in a manner compatible with those purposes;

³DPA 1998, Sch 1.

⁴DPA 1998, Pt II.

⁵DPA 1998, Sch 1, Pt I, para 1 to 8.

3. adequate, relevant and not excessive for the purposes for which they are processed;
4. accurate and kept up to date (where necessary);
5. kept no longer than is necessary for the purposes for which they are processed;
6. processed in accordance with the rights of data subjects that have been stipulated in the enactment;
7. protected by the use of appropriate measures (technical and organisational); and
8. restricted to being processed in the European Economic Area unless adequate protections for the rights and freedoms of data subjects can be ensured.

Responsibility resides with a data controller, “who (either alone or jointly or in common with other persons) determines the purposes for which and the manner in which any personal data are, or are to be, processed”.⁶ It is with regards to *purposes* (which we assume are to be specified and lawful) on which compliance with most of the DPPs hinges: ‘fair’ and ‘lawful’ processing, in particular, whether data subjects have been informed or provided consent where appropriate (related to the 1st DPP); obtaining or further processing of data (the 2nd); characteristics of the data (the 3rd); accuracy (the 4th); retention period (the 5th); and violations of the rights of data subjects (the 6th).

Provisions related to the rights of data subjects (in Part II) delineate certain actions of stakeholders and prescribe conditions under which they should or should not occur. Consequently, we can abstract them in terms of the behaviours of (or interactions with) a system and represent prescribed conditions as constraints on those behaviours (or interactions). The aspects of interactions in common are: a triggering event (e.g., a notification in writing or the processing of personal data) initiated by either a data subject or data controller; a transfer of information (that is abstracted as a data object) in which its content and form are prescribed; and prescribed conditions under which a particular action is or is not performed, such as the amount of time (to include a start and duration) a data controller has to respond to a request for information from a data subject.

The de-identification of personal data is not explicitly addressed in the DPA (although a code of practice on anonymisation exists⁷). The crux of the DPA, it may be argued, is the notion of personal data, and, as such, the applicability of the enactment to data depends on whether the data in question are considered to be personal data. It follows that the inflection point at which personal data are no longer subject to the DPA is, then, the point at which the data are processed into a form that is no longer considered to be ‘personal’. We can surmise that data can be considered ‘non-personal’ when data subjects can no longer be identified from the data themselves or in combination with other information likely to be obtained by a data controller.⁸

Our shared interest with data controllers and software engineers is in developing an approach to support reasoning

⁶Data Protection Act 1998, pt I, s 1(1).

⁷<https://ico.org.uk/media/for-organisations/documents/1061/anonymisation-code.pdf>

⁸Data Protection Act 1998, pt I, s 1(1).

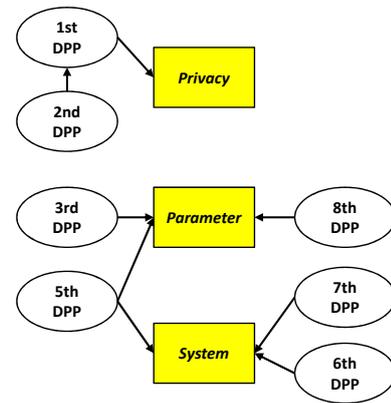


Figure 2: Mapping relevant DPPs.

about the processing of personal data by a system in a required manner (such as in accordance with the DPA). We next consider the application of Z [14] to model a system associated with the release of data in accordance with its encompassing regulatory environment.

7. A FORMAL MODEL

7.1 A Notion of Privacy

We consider *informational privacy* — whereby individuals control information about themselves and are able to determine how it is communicated. We interpret (on the basis of context) the determination by a data controller of ‘purposes’ and ‘manner’ as specifications. We consider *purposes* to be the aspect of privacy that its (privacy) specification hinges on; the *manner* is combined with purposes to delineate processing with sufficient granularity. In Figure 2, the DPPs (except for the 4th, which we consider as an assumption) are mapped to their representation in our model. Implementations of *Privacy* and *Parameter* are represented as schemas. The *System* is associated with either functional or structural aspects of our model, in particular: access control (related to the 7th DPP); retention of the data that are produced (related to the 5th DPP); and instantiations of *Privacy* and *Parameter*.

In relation to a purpose, the 1st DPP (and related aspects of the DPA) provide us with two other types upon which to construct a notion of *Privacy*: a data subject being informed and the provision of consent by, or on behalf of, a data subject. We start by introducing these types.

[*Purpose, Inform, Consent*]

We consider purpose to mean “the reason for which something is done”⁹ and we interpret ‘something’ to mean the disclosure-processing of personal data. We assume that, for individuals to have consented, they would have had to have been adequately informed (about the processing) prior to its provision. We do not distinguish between a consent that has been explicitly or implicitly given, but, rather, that a data controller has determined a data subject has consented to certain processing. The granularity of the characterisation of

⁹www.oxforddictionaries.com/definition/english/purpose

each variable should be sufficiently descriptive as to provide, in particular, data subjects with an appropriate level of understanding of: the reasons that underpin the processing of their data; what a data subject has been informed about in relation to processing; and what they have consented to be done with their data. The higher the granularity with which *Purpose*, *Inform* and *Consent* are described by a data controller, the more likely it is that processing will be consistent with legal and regulatory requirements for privacy.

We consider a parameter to represent “a numerical or other measurable factor forming one of a set that defines a system or sets the conditions of its operation”¹⁰. In particular, we consider a parameter to be the condition under which processing is carried out: a parameter is determined in relation to a particular process. This corresponds to the primary outputs of the *PAR* process of Section 4. We abstract a parameter as a tuple formed by elements: the type of information (*Attribute*) and its value (*Data*).

[*Attribute*, *Data*]

The types of information that may be described as parameters are as follows.

- *Data recipients* are persons or entities to whom the data are to be shared. We do not consider a person who is able to access the data during processing as a recipient, but, instead, as a user in the context of an access control mechanism.
- *Data characteristics* are considered as inputs to, and outputs of, a particular process. From an input, a process under certain conditions then produces an output. The data characteristics of the required output are described implicitly as a set of conditions on the process. As sensitive personal data pertain to certain types of information (e.g., racial or ethnic origin), we assume that the data have been recorded in a manner that facilitates their identification by a process.
- *Processing methods* are mechanisms by which the data are processed. Where there are choices, a data controller determines the mechanism or multiple mechanisms that are applied during processing.
- *Data privacy and utility* are abstracted on the basis of the processing conditions that have produced the data or the results of tests applied to the data. The requirements for privacy and utility of de-identified data are described as parameters such that: processing produces data in which certain attributes are absent (e.g., the Safe Harbor method¹¹); an appropriate (as determined by a data controller) privacy (and/or utility) parameter (e.g., $k = 10$) related to a rendering algorithm (e.g., k -Anonymity [28]) is satisfied during processing; or results of data testing (privacy and/or utility) satisfy criteria (as determined by a data controller).

7.2 A Construct of Privacy

Disclosure-processing begins with the extraction of personal data from a source and ends with or without their disclosure. This sequence of events (or processes), from a

¹⁰ www.oxforddictionaries.com/definition/english/parameter

¹¹ Privacy Rule of HIPAA 1996 §164.514(b)(2).

beginning to an end, constitutes an instance of processing. Then, privacy (from beginning to end) ought to be defined in relation to each instance of processing. Our notion of privacy is based on three factors related to: the privacy rights that are afforded to data subjects by laws and regulations; the manner in which personal data are processed; and the data that are produced as the result of processing.

We abstract privacy as a set of constraints on a system to keep it from transitioning to an unwanted state (that violates privacy). To formally represent these constraints, we consider: *Purpose*, *Inform* and *Consent* to abstract the rights of data subjects; the notion of *Parameter* to abstract the manner in which data are processed; and, from our abstraction of disclosure-processing, the data that are produced as *extdata* and *rendata*. To this end, we introduce the schema *Privacy*:

$$Privacy \hat{=} [\textit{purpose} : \textit{Purpose}; \\ \textit{inform} : \textit{Inform}; \textit{consent} : \textit{Consent}]$$

In Section 7.1, we abstracted parameters as sets of tuples formed by two basic types: *Attribute* and *Data*. Consequently, parameters are captured by the schema *Parameter*:

$$Parameter \hat{=} [\textit{extparam}, \textit{renparam}, \textit{tesparam}, \\ \textit{disparam} : \mathbb{P}(\textit{Attribute} \times \textit{Data})]$$

Our construct (with respect to a disclosure-processing instance) is formed by linking *Privacy* and *Parameter* instances, as well as the data (*extdata* and *rendata*) via a process identifier (denoted *PID*). The same *PID* links the privacy construct to an access control mechanism. Thus, the privacy construct and its implementation have to be considered in context.

7.3 A System

Our model is in three parts: an access control mechanism to support an implementation of privacy (*ACCORE*); the source of personal data that are to be processed for release (*DataStore*); and the relation of a process identifier to an instantiation of the privacy construct and the data produced during processing.

As privacy can be characterised as a constraint on processing and security is a constraint on the access to a process, an access control mechanism can provide a platform for the implementation of privacy. We extend the formal model of role-based access control [11] of [22] by incorporating the notion of privacy as an element of a permission.

We consider three basic types: *Action*, *Resource*, and a set of unique identifiers, *PID*:

$$[\textit{Action}, \textit{Resource}, \textit{PID}]$$

Permissions combine elements of these types:

$$Perm == Action \times Resource \times PID$$

We introduce two further types: *User* and *Role*.

$$[\textit{User}, \textit{Role}]$$

The schema *ACCORE* captures an RBAC policy:

$$ACCORE \hat{=} [\textit{user} : \mathbb{P} \textit{User}; \textit{role} : \mathbb{P} \textit{Role}; \\ \textit{perm} : \mathbb{P} \textit{Perm}; \textit{ur} : \textit{User} \leftrightarrow \textit{Role}; \\ \textit{rp} : \textit{Role} \leftrightarrow \textit{Perm}; \textit{up} : \textit{User} \leftrightarrow \textit{Perm} \mid \\ \textit{ur} \in \textit{user} \leftrightarrow \textit{role} \wedge \textit{rp} \in \textit{role} \leftrightarrow \textit{perm} \wedge \\ \textit{up} = \textit{ur} \circ \textit{rp}]$$

Here: *user*, *role* and *perm* represents the current set of users, the current set of roles and the current set of permissions, respectively; *ur* captures associations of users with assigned roles; *rp* captures associations of roles with assigned permissions; and *up* captures associations of users with their permissions. For a user to have access (via a permission) to an aspect of the system, the user has to have been assigned a role that is associated with that permission.

Alice, in her role as a data controller, has been assigned the permission (*add, par, nullpid*). She can access the *PAR* process and add instantiations of *Privacy* to the system. John, in his role as a Customer Service Representative, has been assigned the permission (*run, ext, nullpid*). To perform his duties, he is able to access most of a customer's personal data. Let us assume that Alice has defined an instantiation of the schema *Privacy* for the disclosure-processing of billing information that she has associated with a *PID* of 1. Although John is able to access most of a customer's personal data, he is not able to run the *EXT* process based on parameters where *PID* = 1.

Personal data (in the form of microdata) that have been obtained and recorded are often held in a relational database. A particular characteristic of an individual is abstracted as an *Attribute-Data* pair. Our abstraction of a source of personal data for disclosure-processing is twofold: a relational database as a table of data (denoted *table*); and the processed data (denoted *shared*) as sets of *Attribute-Data* pairs. This is collected together in the schema *DataStore*:

$$\begin{aligned} \text{DataStore} \hat{=} [& \text{table} : \mathbb{P}(\text{Attribute} \rightarrow \text{Data}); \\ & \text{shared} : \mathbb{P}(\text{Attribute} \times \text{Data}) \mid \\ & \forall t_1, t_2 : \text{table} \bullet \text{dom } t_1 = \text{dom } t_2 \wedge \\ & \forall s : \text{shared} \bullet \exists f : \text{table} \bullet s \in f] \end{aligned}$$

Here, the information pertaining to data subjects are abstracted as rows of data (in which *table* represents the totality of data being held by a data-owner) and the data that are to be processed (*shared*) have been drawn from *table*.

A *PID* value is mapped to an instantiation of *Privacy*. The same *PID* value is used to define a permission for access control that is associated with an instance of processing. This same *PID* value is mapped to an instantiation of *Parameter* that has been defined by a data controller. Then, processes produce data (*extdata* and/or *rendata*) based on *Privacy* and *Parameter* that are related to the same *PID* value. The data are added to the system as a binary relation using the same *PID* value.

Our system comprises the schema *ACCORE*, the schema *DataStore*, and functions mapping elements of *PID* to elements of *Privacy*, *Parameter*, *extdata* and *rendata*, respectively. (For the sake of brevity, we omit constraints.)

$$\begin{aligned} \text{System} \hat{=} [& \text{ACCORE}; \text{DataStore}; \\ & \text{privacy} : \text{PID} \rightarrow \text{Privacy}; \\ & \text{parameter} : \text{PID} \rightarrow \text{Parameter}; \\ & \text{extdata} : \text{PID} \rightarrow \mathbb{P}(\text{Attribute} \times \text{Data}); \\ & \text{rendata} : \text{PID} \rightarrow \mathbb{P}(\text{Attribute} \times \text{Data})] \end{aligned}$$

It is assumed that *extdata* and *rendata* have been produced from *shared*, and a particular instantiation of *Parameter* and data that are produced during processing (*extdata* and *rendata*) are tied to a particular instantiation of *Privacy* (via a *PID*).

Assume that Alice has added an instantiation of *Privacy*

(where *PID* = 1) to the system for billing. The *EXT* and *DIS* processes are involved in disclosure-processing. Further assume that Alice has used the same process identifier to add to the system, as an instantiation of *Parameter* (where *PID* = 1), the parameters to constrain disclosure-processing. John has been assigned the (*run, ext, nullpid*) permission, which allows him access to more personal data about a customer than would be required to process data for billing. Based on this, Alice has decided that John can assume the added responsibility for billing. Although John would gain access to an additional process (*DIS*), Alice believes that the permission would be sufficiently restrictive. A new role is created for billing and is assigned the permissions (*add, ext, 1*) and (*run, dis, 1*). This new role is then assigned to John. Now let us assume that users are granted access to processes (or resources) only if permissions that have been assigned to them (via roles) match criteria that have been defined for those processes. For billing, John would be allowed to: access the *EXT* process; from a source, extract the relevant personal data based on constraints prescribed by *Privacy* (where *PID* = 1) and *Parameter* (where *PID* = 1); add the output of extraction to the system in which *PID* = 1 is mapped to *extdata*; then, access the *DIS* process; and disseminate the *extdata*, based on constraints that are prescribed by *Parameter* (where *PID* = 1). Here, John is able to run the *DIS* process only if the appropriate extracted data (*extdata*) have been added to the system (where *PID* = 1). The system will not allow John to alter instantiations of *Privacy* and *Parameter* that are associated with a process nor the data that are produced.

8. ANALYSIS

A system interacts with its environment, which may involve users of, and threats to, it. Interactions between a system and these potential actors (directly or indirectly via external processes) can cause a system to transition from one state to another. As a starting point for modelling systems of increasing complexity, we consider a data-sharing situation in which the state-space is tractable.

We have given consideration to our smart meter scenario (in the context of data-sharing), the DPA (for privacy requirements), and our Z-based model of a system — both its (privacy) specification (denoted *System_S*) and its implementation (denoted *System_I*), with *System_I* being a more concrete representation of a system congruent with the specification. In addition, we adapt the *Basic Security Theorem* of [4] (while being aware of its limitations, as discussed in, for example, [17]) so that: if a system satisfies our privacy requirements in its initial state, and there is a guarantee that every transition also ensures that our privacy requirements are met, then we may conclude that all states will satisfy our privacy requirements.

We abstracted the processing of personal data for disclosure in terms of the schema *System*, before and after an execution of a process (represented as an operation). We represented 26 such operations in terms of Z. Operations were of two types: those that affected the variables in the schema *System* and those that did not. In relation to requirements, predicates were defined for operations such that privacy was assured. By varying the values of inputs, the same operation was used to model multiple data-sharing situations.

As an example, consider the operation *TestRenderData*.

$$\begin{aligned}
& \text{TestRenderData} \hat{=} \\
& [\exists \text{System}; \\
& \quad i? : \text{PID}; u? : \text{User}; b! : \text{Boolean}; \\
& \quad t? : \mathbb{P}(\text{Attribute} \times \text{Data}) \mid \\
& \quad i? \neq \text{nullpid} \wedge \\
& \quad i? \in \bigcap \{ \text{dom } \text{privacy}, \\
& \quad \quad \text{dom } \text{parameter}, \text{dom } \text{extdata} \} \wedge \\
& \quad (\text{parameter}(i?).\text{tesparam} \neq \emptyset \wedge \\
& \quad (\text{run}, \text{tes}, i?) \in \text{up}(\downarrow \{u?\}) \wedge \\
& \quad t? \neq \emptyset \wedge \\
& \quad t? \subseteq (\text{parameter}(i?).\text{tesparam} \wedge \\
& \quad \text{rendata}(i?) \subseteq t? \Rightarrow b! = T \wedge \\
& \quad \neg(\text{rendata}(i?) \subseteq t? \Rightarrow b! = F \wedge \\
& \quad \text{privacy}' = \text{privacy} \wedge \\
& \quad \text{parameter}' = \text{parameter} \wedge \\
& \quad \text{extdata}' = \text{extdata} \wedge \\
& \quad \text{rendata}' = \text{rendata}]
\end{aligned}$$

This tests the rendered data, leaving the state of the system unchanged. We assume that testing is applied to data to be disseminated. Preconditions to test *rendata* are: the *PID* value is not null; the *PID* value appears in the domains of *privacy*, *parameter* and *rendata*; in relation to the instantiation of *Parameter*, parameters for testing (*tesparam*) have been defined; and, in relation to the *PID* value, a user is permitted to perform the action (to run the test) and access the resource (*TES* process). In relation to the *PID* value associated with the operation, the *rendata* are tested based on parameters constrained by *tesparam*. We abstract testing as a comparison between the data being tested and parameters. Based on the result of testing the data, a user may decide whether or not to proceed with dissemination. (We assume a type *Boolean*, with two elements: *T* and *F*.)

System_S was the basis from which we derived an implementation (or less abstract representation) of privacy. As an example, consider the operation *AddParameter*. The specification is the addition of schema *Parameter* to the system. To limit the number of distinct combinations of parameters in the implementation, we constrained the operation further. (Again, for the sake of brevity, we provide the constraints in *System_I*, but not in *System_S*):

$$\begin{aligned}
& \text{AddParameter} \hat{=} \\
& [((p?.\text{extparam} \neq \emptyset \wedge p?.\text{disparam} \neq \emptyset) \wedge \\
& \quad (p?.\text{renparam} = \emptyset \wedge p?.\text{tesparam} = \emptyset \wedge \\
& \quad p?.\text{disparam} \subseteq p?.\text{extparam}) \vee \\
& \quad (p?.\text{renparam} = \emptyset \wedge p?.\text{tesparam} \neq \emptyset \wedge \\
& \quad p?.\text{extparam} \subseteq p?.\text{tesparam} \wedge \\
& \quad p?.\text{disparam} \subseteq p?.\text{extparam}) \vee \\
& \quad (p?.\text{renparam} \neq \emptyset \wedge p?.\text{tesparam} = \emptyset \wedge \\
& \quad p?.\text{renparam} \subseteq p?.\text{extparam} \wedge \\
& \quad p?.\text{disparam} \subseteq p?.\text{renparam}) \vee \\
& \quad (p?.\text{renparam} \neq \emptyset \wedge p?.\text{tesparam} \neq \emptyset \wedge \\
& \quad p?.\text{tesparam} \subseteq p?.\text{extparam} \wedge \\
& \quad p?.\text{renparam} \subseteq p?.\text{tesparam} \wedge \\
& \quad p?.\text{disparam} \subseteq p?.\text{renparam}))]
\end{aligned}$$

ProZ was applied to our model to analyse privacy (in terms of satisfying requirements). Specifically, the ProZ animator was used to verify (on a step-by-step basis) that each state satisfied its privacy requirements. To analyse all reachable states, ProZ's model and refinement checkers were used. However, a lower cardinality for variable values (than that used with the animator) and a higher level of abstraction for access control had to be applied to support a tractable

Model	States	Transitions
<i>System_S</i>	614,385	2,382,109
<i>System_I</i>	29,225	114,867

Table 1: Total states and transitions covered by the ProZ model checker.

state space. The number of states and transitions covered by ProZ without finding deadlocks, invariant violations and errors are presented in Table 1. Furthermore, *System_I* was considered to be a trace refinement of *System_S* by ProZ. The analysis found that (irrespective of data-sharing situation), in the execution of operations, the system transitioned to states that satisfied requirements for privacy. However, this assurance of privacy is limited to the context (in terms of the state of a system): an individual's privacy could still be violated even though a system, during the processing of their data, never transitioned to an unwanted state (e.g., an insider using their system authorisations for nefarious purposes or an attacker assuming the identity of an authorised system user).

The assurance of privacy is highly dependent on the values that are determined and set by a data controller for the attributes associated with *Privacy* and *Parameter*. In assigning a permission (that consists of a *PID* value) to a role, there has to be regard for the instantiations of *Privacy* and *Parameter* that are associated with the *PID* value to satisfy privacy requirements. To minimise the need for manual intervention, information related to privacy (e.g., a data subject being informed, sensitive personal data, or time) has to be expressed or annotated in such a manner that facilitates processing. Interactions (in particular, those that are recurrent) with data subjects in which a data controller incurs a temporal obligation (e.g., data controller notifications, data subject requests or data retention) has to be handled primarily by the system. In our model, we assume that a data controller's obligation is satisfied instantaneously following a triggering system event.

9. CONCLUSIONS

We have shown, in the spirit of Tschantz and Wing's contribution [29], how the judicious use of a formal model can help in thinking about privacy. In doing so, we have paid attention to a particular problem: that of data-sharing.

There are three areas of further research that we intend to pursue in the near future. First, we will use Z schema calculus to model privacy in the context of systems of increasing complexity. Second, we intend constructing formal models of threats. Finally, we intend modelling the aforementioned Privacy Rule of HIPAA, the privacy requirements of which have been used by a number of authors to validate proposed approaches in the context of data-sharing.

10. REFERENCES

- [1] J.-R. Abrial. *The B-Book: Assigning Meanings to Programs*. Cambridge University Press, 1996.
- [2] I. Agrafiotis, S. Creese, M. Goldsmith, and N. Papanikolaou. Applying formal methods to detect and resolve ambiguities in privacy requirements. In S. Fischer-Hübner, P. Duquenoy, M. Hansen, R. Leenes, and G. Zhang, editors, *Privacy and*

- Identity Management for Life*, volume 352 of *IFIP Advances in Information and Communication Technology*, pages 271–282, 2011.
- [3] A. Barth, A. Datta, J. Mitchell, and H. Nissenbaum. Privacy and contextual integrity: Framework and applications. In *Proceedings of the 2006 IEEE Symposium on Security and Privacy (SP 2006)*, pages 184–198. IEEE, 2006.
 - [4] D. E. Bell and L. J. La Padula. Secure computer systems: Mathematical foundations, MTR-2547, Vol. I. Technical Report ESD-TR-73-278-I, The MITRE Corporation, Bedford, March 1973.
 - [5] I. Brown. Britain’s smart meter programme: A case study in privacy by design. *International Review of Law, Computers & Technology*, 28(2):172–184, 2014.
 - [6] A. Cavoukian. Privacy by design: The 7 foundational principles. Information and Privacy Commissioner of Ontario, Canada, 2009.
 - [7] O. Chowdhury, A. Gampe, J. Niu, J. von Ronne, J. Bennett, A. Datta, L. Jia, and W. H. Winsborough. Privacy promises that can be kept: A policy analysis method with application to the HIPAA Privacy Rule. In *Proceedings of the 18th ACM Symposium on Access Control Models and Technologies (SACMAT 2013)*, pages 3–14. ACM, 2013.
 - [8] C. Davies and R. Collins. Balancing potential risks and benefits of using confidential data. *The British Medical Journal*, 333(7563):349–351, 2006.
 - [9] H. DeYoung, D. Garg, L. Jia, D. Kaynar, and A. Datta. Experiences in the logical specification of the HIPAA and GLBA privacy laws. In *Proceedings of the 9th Annual ACM Workshop on Privacy in the Electronic Society (WPES 2010)*, pages 73–82. ACM, 2010.
 - [10] C. Dwork. Differential Privacy. In M. Bugliesi, B. Preneel, V. Sassone, and I. Wegener, editors, *Proceedings of the 33rd International Colloquium on Automata, Languages and Programming (ICALP 2006), part II*, volume 4052 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2006.
 - [11] D. Ferraiolo, D. R. Kuhn, and R. Chandramouli. *Role-Based Access Control*. Artech House, 2003.
 - [12] X. Fu. Conformance verification of privacy policies. In M. Bravetti and T. Bultan, editors, *Proceedings of the 7th International Workshop on Web Services and Formal Methods (WS-FM 2010)*, volume 6551 of *Lecture Notes in Computer Science*, pages 86–100. Springer, 2011.
 - [13] B. C. M. Fung, K. Wang, R. Chen, and P. S. Yu. Privacy-preserving data publishing: A survey of recent developments. *ACM Computing Surveys*, 42(4):14:1–14:53, 2010.
 - [14] ISO/IEC. ISO/IEC 13658: Information Technology — Z Formal Specification Notation — Syntax, Type System and Semantics. ISO/IEC, 2002.
 - [15] D. Kouzapas and A. Philippou. A typing system for privacy. In S. Counsell and M. Núñez, editors, *Proceedings of Software Engineering and Formal Methods (SEFM) 2013 Collocated Workshops*, Lecture Notes in Computer Science, pages 56–68. Springer, 2014.
 - [16] D. Le Métayer. Privacy by design: A formal framework for the analysis of architectural choices. In *Proceedings of the 3rd ACM conference on Data and Application Security and Privacy (CODASPY 2013)*, pages 95–104. ACM, 2013.
 - [17] J. McLean. A comment on the “basic security theorem” of Bell and LaPadula. *Information Processing Letters*, 20(2):67–70, 1985.
 - [18] Q. Ni, E. Bertino, and J. Lobo. An obligation model bridging access control policies and privacy policies. In *Proceedings of the 13th ACM Symposium on Access Control Models and Technologies (SACMAT 2008)*, pages 133–142. ACM, 2008.
 - [19] Q. Ni, D. Lin, E. Bertino, and J. Lobo. Conditional privacy-aware role based access control. In J. Biskup and J. López, editors, *Proceedings of the 12th European Symposium On Research In Computer Security (ESORICS 2007)*, volume 4734 of *Lecture Notes in Computer Science*, pages 72–89. Springer, 2007.
 - [20] P. A. Norberg, D. R. Horne, and D. A. Horne. The privacy paradox: Personal information disclosure intentions versus behaviors. *Journal of Consumer Affairs*, 41(1):100–126, 2007.
 - [21] D. Plagge and M. Leuschel. Validating Z specifications using the ProB animator and model checker. In J. W. M. Davies and J. Gibbons, editors, *Integrated Formal Methods*, volume 4591 of *Lecture Notes in Computer Science*, pages 480–500. Springer, 2007.
 - [22] D. J. Power, M. A. Slaymaker, and A. C. Simpson. On formalizing and normalizing role-based access control systems. *The Computer Journal*, 52(3):305–325, 2009.
 - [23] L. Sankar, S. Rajagopalan, S. Mohajer, and H. Poor. Smart meter privacy: A theoretical framework. *IEEE Transactions on Smart Grid*, 4(2):837–846, 2013.
 - [24] C. Skinner. Statistical disclosure risk: Separating potential and harm. *International Statistical Review*, 80(3):349–368, 2012.
 - [25] D. J. Solove. Conceptualizing privacy. *California Law Review*, 90(4):1087–1155, 2002.
 - [26] J. M. Spivey. *The Z Notation: a Reference Manual*. Prentice Hall, 2nd edition, 1992.
 - [27] S. Suriadi, C. Ouyang, J. Smith, and E. Foo. Modeling and verification of privacy enhancing protocols. In *Proceedings of the 11th International Conference on Formal Engineering Methods (ICFEM 2009)*, volume 5885 of *Lecture Notes in Computer Science*, pages 127–146. Springer, 2009.
 - [28] L. Sweeney. k-anonymity: A model for protecting privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5):557–570, 2002.
 - [29] M. C. Tschantz and J. M. Wing. Formal methods for privacy. In A. Cavalcanti and D. Dams, editors, *Proceedings of the 2nd World Congress on Formal Methods (FM 2009)*, volume 5850 of *Lecture Notes in Computer Science*, pages 1–15. Springer, 2009.
 - [30] J. C. P. Woodcock, P. G. Larsen, J. Bicarregui, and J. Fitzgerald. Formal methods: Practice and experience. *ACM Computing Surveys*, pages 19:1–19:36, 2009.