

Towards models for privacy preservation in the face of metadata exploitation

Marine Eviette^{1,2} and Andrew Simpson^{1,3}

¹ Department of Computer Science, University of Oxford
Wolfson Building, Parks Road, Oxford OX1 3QD, UK

² marine.eviette@cybersecurity.ox.ac.uk

³ andrew.simpson@cs.ox.ac.uk

Abstract. It is well understood that today’s society generates enormous volumes of data, with much of that data being used to drive the emerging digital data economy. However, due to dataveillance and related concerns, the magnitude and specificity of this data has given rise to well documented privacy concerns. Amongst these concerns is the unregulated collection of online metadata. Despite being commonly used, metadata is a term that few understand. Yet its richness, prevalence and collection has serious implications for many. In this paper we describe initial work with regards to a means of allowing users to control access to metadata pertaining to them, with a view to addressing these implications. To do so, we leverage the work of the Solid data decentralisation project and build upon the notion of Category-Based Access Control.

1 Introduction

Ubiquitous computing has given rise to vast quantities of data, with some sources suggesting that 2.5 quintillion bytes of data are being created on a daily basis¹. These enormous volumes of data have been used to fuel a growing data economy, with many companies taking advantage of the opportunities that can flow from leveraging fine-grained consumer personalisation. In concert with these opportunities there are challenges — most obviously in the guise of threats to privacy in the context of a market where personal data is traded as a commodity. The acknowledgment of such threats has led to an evolution of privacy guidelines and requirements in the online space, driven by manifestations of data management legislation such as the EU’s General Data Protection Regulation (GDPR)².

Despite increasing concerns over data management with reference to privacy and human rights, the topic of *metadata* has remained relatively unexplored. (In the context of this paper we consider metadata to be any data serving to provide contextual or otherwise additional information about other data.) While the definition of online metadata has remained relatively constant, the information to which it pertains has become far more complex, its volume has become greater,

¹ <https://www.domo.com/learn/data-never-sleeps-5>

² <https://gdpr-info.eu>

and its specificity has, for many, become more worrisome. Online metadata is routinely overlooked since it is often generated without users' explicit consent and awareness due, in part, to the unfavourable online conditions of dark patterns [1,2], privacy-averse designs [3], and general privacy fatigue [4]. As a result, there is often unregulated collection of such metadata, which can have troubling, and potentially disastrous, implications for an individual's privacy.

A single instance of metadata is often trivialised as being harmless, since, in isolation, it may not compromise an individual's privacy. However, it is well understood that, when aggregated with other instances of metadata, sensitive details may be inferred or reidentification may occur. This is akin to the problem of *jigsaw re-identification* [5] in the context of publicly available data, though, in this case, the reidentification may be undertaken by data custodians, data brokers, or third-party organisations. Notably, the issues of concern extend far beyond reidentification due to the magnitude of metadata collection and consequent analysis in the online realm. To this end, we find that the breadth, pervasiveness and implicitness of metadata has led to online browsing habits shaping an extremely detailed trail of online activities via the passive *digital footprints* [6] that are unknowingly left behind. Such digital footprints that are accumulated by means of dataveillance have led to inferences on identity elements that expose user identity while rendering these users powerless in the management of their personal privacy.

At present, there is no single unifying method to identity management as it relates to this problem. As such, we leverage solutions to related problems to give rise to formal models of access control with a view to applying them to metadata and identity management. In doing so, we build upon the work of the Solid data decentralisation project³, which aims to preserve personal privacy on a decentralised Web via Access Control Lists. We hope that, by building upon the Solid project's foundation, we may be able to formalise our specific problem.

It is worth noting that, while the access control 'problem' is well defined and well understood [7], things become more complicated when we start to consider broader privacy concerns: considerations of beneficiary and ownership can cause complex issues [8], not least because it is often the case that, despite (for example) legal obligations, the data-holder will not typically have a vested interest in the protection of privacy.

Deviation from 'traditional' policy requirements has seen the introduction of a variety of novel approaches to access control. For example, in [9] Fernandez *et al.* report upon a framework for secure data collection whereby users are able to specify data management rules through the extension of the *Category-Based Access Control* metamodel [10]. Category-Based Access Control (CBAC) adopts a foundational notion of access control, by considering notions of categories in order to provide flexible access control concepts, which can be specialised for particular needs. In particular, the logical foundations of this approach enable the analysis and verification of policy properties. We argue that utilising such an approach should enable one to reason about the problem at hand. Thus, we

³ <https://solidproject.org>

present a model of CBAC in terms of the Z notation [11, 12], the mathematical language of which is based upon typed set theory and first-order predicate logic. In addition, the schema language of Z allows us to combine aspects of a model in a way that makes reuse and composition of aspects relatively straightforward and also allows us to reason about evolution of states.

The choice of Z was influenced, in part, by the fact that its notation is relatively straightforward to understand and that its underlying logical structures have much in common with those of relational databases. In addition, it has been used previously in describing models of access control (see, for example, [13]).

Given the foundations provided by the CBAC framework and the Solid decentralisation project, we frame our research problem thus:

How can we leverage a Solid-style approach to data ownership together with formal models of data access to aid in the protection of data subjects' privacy, so as to mitigate inference-driven identity exposure from metadata collection?

2 Background and Motivation

2.1 Metadata and dataveillance

Metadata, in this era of pervasive computing, has become infrastructural [14] in the manner in which it invisibly supports almost all interactions in the online realm, leaving a distinct digital trail for any given user. This digital trail is made up of *digital footprints* [6] from user interaction. Digital footprints can be thought of as identifying features that are used to infer elements or attributes of an identity, or are themselves identity elements or attributes, where identity elements are defined to be single pieces of information that are indicative of an identity [15].

These digital footprints consist of two types: those of a *passive* nature, such as metadata, and those which might be described as *active*, such as a Facebook post or a tweet. Passive digital footprints are comprised primarily of metadata created unintentionally by user activities without the user's explicit consent or knowledge. Collection of this passive data is particularly disconcerting due to the nature of online surveillance, or *dataveillance* [16].

The phenomenon of dataveillance has, in part, motivated the race for storage, analysis and creation of large data volumes whilst simultaneously degrading user privacy and exerting control in intimate settings. Moreover, dataveillance has played a pivotal role in the inference of identity or digital personas, where, as early as the 1990s, there were indications of organisations' ability to create digital personas [17].

Today, dataveillance's role in big data has become drastically more pervasive, and, hence, so too have these inferred digital personas. In fact, in this surveilled online environment, with continuous monitoring in the form of metadata, many truthful identity elements can be inferred without the user needing to explicitly express them online [18], which exposes users' identity information.

With regards to identity management in the offline space, the multiplicity of identity is naturally explored through context, where a single individual’s identity is made up of numerous aspects that they may choose to reveal. This allows individuals to refrain from sharing certain details or identifiers outside of settings that they deem appropriate. Similarly, the contexts in the online space reflects one’s self-presentation whereby individuals are free to choose how they wish to express themselves to others across a multitude of platforms. However, as it pertains to metadata in the online space, this multiplicity struggles to adapt to a persistent, non-forgetting terrain that often lacks context despite the aggressive aggregation and sharing. This is epitomised by revelations that a unique persona can always be inferred for individuals, regardless of the differing personas maintained by these individuals across the online space.

Given that this problem is only set to worsen as technology becomes further intertwined with individuals’ lives with, for example, the increased ubiquity of the IoT and ‘smart’ devices, the generated metadata agglomeration will only further expand. Therefore, regulation and access control mechanisms are needed in this area to divert privacy violations that threaten human rights and to ensure the metadata collected is not easily exploitable.

2.2 Data ownership

The definition of privacy is an all-encompassing, ever-evolving, fluid notion that has enjoyed (and continues to enjoy) many definitions [19]. The varying definitions for privacy can, in part, be seen as a result of discrepancies between individuals’ own conceptions, the varying cultures of belonging, and economic and lawful practices. Despite these variances, privacy is viewed by many as an important factor in human life — to some, a human right — and “[over] 130 countries have constitutional statements regarding the protection of privacy” [20].

As far as we are concerned in this paper, privacy can be viewed as

A limitation on access to self

(This is reminiscent of earlier philosophical definitions, such as those of [21] and [22].)

Defining privacy in this fashion allows us to move towards formulating the problem in terms of *access control*: if we consider self as a collection of data objects, we can begin to see how access control may be applied to this problem. However, while it may be a nice intellectual exercise to reason about access to data objects as a method of protecting individuals’ privacy, there is a clear roadblock in the form and nature of today’s Internet, whereby users typically have no control over their own data and, consequently, no control over managing access to it.

Of course, data ownership has long been a source of debate [23, 24]. Initial hopes for the Web narrated a vision of radical decentralisation and freedom [25], yet its incredible growth has been plagued with centralisation, so that, decades later, we see a significant proportion of traffic on the Web flowing through a comparatively minuscule set of corporations.

The disadvantages of such centralisation follow a similar path to the pitfalls of corrupt central entities, such as corrupt governments and companies, whereby we find the Web today ripe with surveillance, data breaches, privacy loss and manipulation. Indeed, a “few large companies now own important junctures of the Web, and consequently a lot of the data created on [it]” [26]. Thus, while individuals largely remain the source of their data, this does not translate into automatic data ownership. Of course, the complexities of data ownership become even more complex when one considers, for example, transfer of data to third parties, analysis information of data aggregates from various sources, etc.

2.3 An alternative approach

A number of projects (including those described by [27] and [26]) have begun to investigate methods for users to retain (or regain) ownership, as well as consequent control, of their data, through the introduction of decentralised architectures that can be implemented on top of the current Internet.

The Web, as originally formulated, was intended to facilitate easy data sharing between researchers across the Internet in a largely decentralised manner. However, the Web today, as we have argued, tends towards centralisation, with data being controlled and processed by a small number of large companies. To counter the resultant privacy dilemmas, the inventor of the Web, Tim Berners-Lee, and colleagues began work on Solid [28] — a distributed data decentralisation project that seeks to enable information sharing in a privacy-preserving manner.

In the Solid approach, data generated by a user is written to their own individual ‘pods’ (with a pod being ‘personal online data store’). These pods may be hosted wherever the user chooses, and the user can then authorise granular access of the pod to other parties as they please. This means that authenticated applications are allowed to request data, assuming that the user has given the particular application permission.

At this point, it is worth noting that other solutions (such as those described in [29] and [30]) offer approaches to decentralised access control via blockchain technology. However, our motivations — including the consideration of aggregation of metadata and the automatic evolution of access control policies — have led us to conclude that developing a solution based on the Solid approach is appropriate for our needs.

For managing control over data, Solid utilises Web Access Control (WAC) Lists. These Web Access Control Lists have much in common with the Access Control Lists generally used in Discretionary Access Control [31] policies.

Users and groups are attached to URIs, or WebIDs; resources are identified by URLs that may refer to web documents or resources. To handle permissions, these latter resources are accompanied by a set of Authorisation statements that describe:

- which agents have access to the resource, and
- what type, or mode, of access the given agent has.

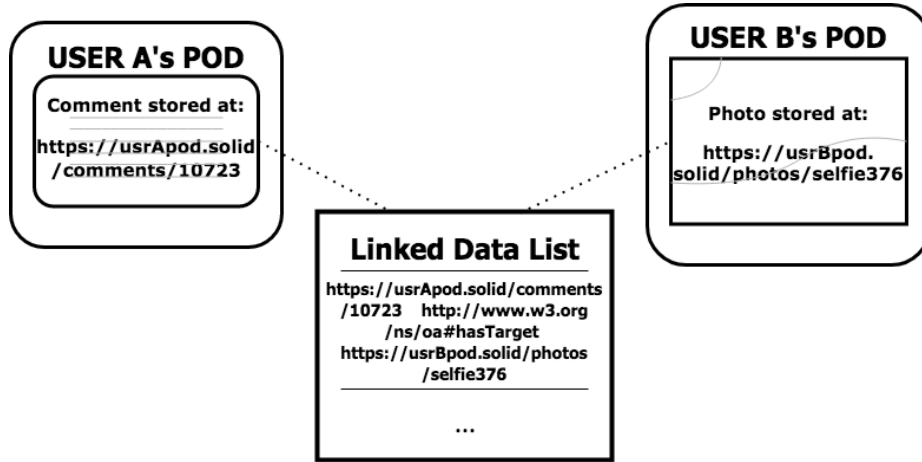


Fig. 1. Links between users' data

These Authorisations are placed into separate WAC documents called Access Control List Resources (ACLs) with the permissions of the ACL resource stored in a *Linked Data* [32] format. Linked Data can be described as typed links that enable explicit connections to be made when necessary. Thus, we may have links between different users' data pods, as depicted in Figure 1.

By adapting the Solid philosophy and principles, there is scope for reasoning about the 'metadata problem' in terms of access control — due to the fact that a fundamental tenet of Solid is that users control access to their own data.

Crucially, though, Solid by itself will be insufficient. With regards to metadata, it is almost inevitable that the pods will not be a sufficient method of protection (as it is usual for metadata to appear 'harmless'). Metadata typically becomes useful as a result of inferences from aggregations. As such, Solid provides the intellectual foundations upon which we build, but we need to go further in terms of reasoning about access. It is the notion of Category-Based Access Control (CBAC) [10] that allows us to do that.

3 CBAC

There is undeniably a trend with regards to the development of novel access control models to handle use cases that are perceived to be new and / or unique. Often these novel approaches have much in common with each other or with previous approaches. An alternative to re-inventing the wheel on a regular basis is to consider a more general, 'primitive' notion upon which new models can be built. The meta-model for Category-Based Access Control (CBAC) [33] is such a notion.

Category-Based Access Control (CBAC) was developed to provide flexible access control concepts that can be specialised for particular needs [33]. With

regards to our problem of interest, CBAC has the potential to provide the foundations for a model that allows one to reason about autonomous changes to permissions.

In CBAC, permissions are assigned to categories of users (as opposed to users); this, in turn, allows permissions to be associated with categories. Categories can be defined on the basis of, for example, roles and resources, as well as attributes and geographical constraints so that permissions can change when a user attribute or geographical location changes, without intervention from an administrator. This would be decidedly beneficial in our case regarding thresholds for data aggregation and inferences, whereupon a permission can be withdrawn.

The CBAC meta-model works on interactions between the following sets [10]:

- A countable set C of categories, where c_0, c_1 , etc. denote arbitrary category identifiers.
- A countable set P of principals, where p_0, p_1 , etc. denote principals.
- A countable set A of named atomic actions, where a_0, a_1 , etc. denote arbitrary action identifiers.
- A countable set R of resource identifiers, where r_0, r_1 , etc. denote arbitrary resources.
- A countable set S of situational identifiers, where s_0, s_1 , etc. denote possible situations that may occur in the system.
- A countable set E of event identifiers, where e_0, e_1 , etc. denote possible events that may happen in the system.

With respect to *permissions* and *authorisations*:

- A permission is an ordered pair $(a, r) \in A \times R$, consisting of an action, $a \in A$, and a resource, $r \in R$.
- An authorisation is a triple, $(p, a, r) \in P \times A \times R$, which associates a permission with a principal, $p \in P$.

Continuing, the meta-model of [10] details the following relations:

- Principal–category assignment: $PCA \subseteq P \times C$, where $(p, c) \in PCA$ if, and only if, the principal $p \in P$ is assigned to the category $c \in C$.
- Permission–category assignment: $ARCA \subseteq A \times R \times C$, where $(a, r, c) \in ARCA$ if, and only if, action $a \in A$ on resource $r \in R$ can be performed by principals associated with the category $c \in C$.
- Authorisations: $PAR \subseteq P \times A \times R$, where $(p, a, r) \in PAR$ if, and only if, the principal $p \in P$ can perform the action $a \in A$ on the resource $r \in R$.

In the following, we shall use ‘syntactic sugar’ of the form $pca(p, c)$, $arca(a, r, c)$ and $par(p, a, r)$ to capture these concepts.

Subsequently, Barker [10] determines that the set of $par(p, a, r)$ facts that hold with respect to the specification of a particular access control policy may be expressed in first-order terms thus:

$$\begin{aligned} \forall p : P; a : A; r : R \bullet \exists c : C \bullet \\ par(p, a, r) \Leftrightarrow pca(p, c) \wedge arca(a, r, c) \end{aligned}$$

A further relationship, ρ , captures the notion of the existence of a relationship, such as inclusion, holding between categories:

$$\begin{aligned} \forall p : P; a : A; r : R \bullet \exists c, c' : C \bullet \\ par(p, a, r) \Leftrightarrow pca(p, c) \wedge \rho(c, c') \wedge arca(a, r, c') \end{aligned}$$

Although the model does not consider aspects such as sessions, delegations, denial of permissions or conflict resolution strategies, it is noted that, from this initial basis, these notions can naturally be accommodated [10]. This has been illustrated in practice by the aforementioned contribution of Fernandez *et al.* [9].

4 A Formal Model

As a first step, we have developed a formal model of the Category-Based Access Control meta-model, \mathcal{M} [10], in terms of the Z schema language [11, 12]. The intention is that we should be in a position to leverage and build upon this model as we move forward. As well as providing the necessary formal foundations, Z schemas provide a degree of flexibility — allowing us to add or remove optional elements (such as situational and event identifiers) in a relatively straightforward fashion. In this section we present key aspects of the formal model.

4.1 Types and relations

Our interpretation of the meta-model \mathcal{M} captures the key components of the original presentation as faithfully as possible. For example, the original characterisation gives rise to six *basic types*: C , the set of categories; P , the set of principals/users; A , the set of actions; R , the set of resources; S , the situational identifier set; and E , the event identifier set. (These last two are optional for any particular instantiation.)

Thus, we have the following declaration:

$$[C, P, A, R, S, E]$$

As already discussed, we may consider permissions and authorisations thus:

- A permission is an ordered pair (a, r) consisting of an action $a \in A$ and a resource $r \in R$.
- An authorisation is a triple (p, a, r) that associates the constituent parts of a permission with a principal $p \in P$.

We define the sets $Perm$ (for permissions) and $Auth$ (for authorisations) thus:

$$\begin{aligned} Perm &== A \times R \\ Auth &== P \times Perm \end{aligned}$$

In the previous section, we discussed three relations: principal–category assignment, permission–category assignment, and authorisations. We capture these in our Z model thus.

$$\begin{aligned} PCA &== P \leftrightarrow C \\ ARCA &== C \leftrightarrow Perm \\ PAR &== P \leftrightarrow Perm \end{aligned}$$

Here, $P \leftrightarrow C$ captures the set of all possible relations between the set P and the set C (i.e. the set of all sets of pairs appearing in the Cartesian product $P \times C$) — with PCA being an abbreviation for this collection. The sets of relations, $ARCA$ and PAR , are defined similarly: the former captures relations of type $C \leftrightarrow Perm$; the latter captures relations of type $P \leftrightarrow Perm$.

4.2 The $MModel$ schema

The schema *Instance* captures the ‘current’ categories, principles and permissions of the system under consideration and is defined thus.

$\begin{aligned} &Instance \\ &permset : \mathbb{F} Perm \\ &principalset : \mathbb{F} P \\ &catset : \mathbb{F} C \end{aligned}$

Here, *permset* is a finite set of elements of type *Perm*; *principalset* is a finite set of elements of type *P*; and *catset* is a finite set of elements of type *C*.

The $MModel$ schema is then defined as follows.

$\begin{aligned} &MModel \\ &Instance \\ &pca : PCA \\ &arca : ARCA \\ &par : PAR \end{aligned}$
$\begin{aligned} &\text{dom } pca \subseteq principalset \\ &\text{ran } pca \subseteq catset \\ &\text{dom } arca \subseteq catset \\ &\text{ran } arca \subseteq permset \\ &\text{dom } par \subseteq principalset \\ &\text{ran } par \subseteq permset \\ &par = pca \circ arca \end{aligned}$

The inclusion of *permset*, *principalset* and *catset* via the *Instance* schema allows different instances of the metamodel to operate on different combinations of permissions, principals and categories. The sets *pca*, *arca* and *par* capture principal–category assignment, permission–category assignment and authorisations respectively.

The constraints are direct derivatives of the rules placed upon the sets as defined in the original model. The first six constraints simply restrict the domains and ranges of the *par*, *arca* and *pca* relations (via constraints that leverage the *dom* and *ran* operations). The final constraint captures the fact that the *par* relation can be viewed as the relational composition of *pca* and *arca*: a pair (p_1, p_2) , for some $p_1 \in P$ and some $p_2 \in Perm$, appears in *par* if, and only if, there is some $c \in C$, such that $(p_1, c) \in pca$ and $(c, p_2) \in arca$.

4.3 Support for modularity

To support modularity, we utilise two schemas. The first, which is concerned with the attributes of the *Instance* schema, is defined thus:

<i>SetupRetained</i>	_____
$\Delta MModel$	
$catset' = catset$	
$principalset' = principalset$	
$permset' = permset$	

This schema is used in operations that are concerned only with the relations *arca*, *par* and *pca*.

The second such schema, *ModelRetained*, is used in operations that are concerned only with the sets of categories, principles and permissions.

<i>ModelRetained</i>	_____
$\Delta MModel$	
$par' = par$	
$arca' = arca$	
$pca' = pca$	

4.4 Example operations: permissions

In category-based access control (CBAC), permissions are not assigned to individuals users, but, instead, to categories of users. These categories can then be changed as necessary.

As a means of illustrating operations on our model, we present operations that are specifically concerned with explicit changes to the set of permissions (as opposed to operations concerned with changes made by category or principal assignments).

We start by defining a schema, *UpdatePerms*:

<i>UpdatePerms</i>	_____
<i>ModelRetained</i>	
$catset' = catset$	
$principalset' = principalset$	

To any given instance of the model we want to be able to define a starting set of permissions on the set of actions and resources. As such, we utilise a schema, *AllocatePerms* to allocate a chosen set of permissions to the particular instance:

<i>AllocatePerms</i>	_____
<i>UpdatePerms</i>	
<i>allocation?</i> : \mathbb{F}_1 <i>Perm</i>	
<i>permset</i> = \emptyset	
<i>permset'</i> = <i>allocation?</i>	

The operation *AddPerms* allows the set of permissions to be augmented:

<i>AddPerms</i>	_____
<i>UpdatePerms</i>	
<i>allocation?</i> : \mathbb{F}_1 <i>Perm</i>	
<i>permset</i> $\neq \emptyset$	
<i>permset</i> \cap <i>allocation?</i> = \emptyset	
<i>permset'</i> = <i>permset</i> \cup <i>allocation?</i>	

The operation *RemovePerms* captures the notion of permissions being removed via this route:

<i>RemovePerms</i>	_____
Δ <i>MModel</i>	
<i>allocation?</i> : \mathbb{F}_1 <i>Perm</i>	
<i>permset</i> $\neq \emptyset$	
<i>allocation?</i> \subseteq <i>permset</i>	
<i>permset'</i> = <i>permset</i> \setminus <i>allocation?</i>	
<i>par'</i> = <i>par</i> \triangleright <i>allocation?</i>	
<i>pca'</i> = <i>pca</i>	
<i>arca'</i> = <i>arca</i> \triangleright <i>allocation?</i>	
<i>principalset'</i> = <i>principalset</i>	
<i>catset'</i> = <i>catset</i>	

Here, \triangleright is the range co-restriction operator: in this case, all elements of the set *allocation?* are removed from the ranges of both *par* and *arca*. These updates are necessary as the removal of permissions from *permset* can impact upon both *par* and *arca*.

4.5 Example operations: principals

In CBAC, for users to be granted permissions, they must first be assigned to categories. This results in operations on principals typically involving the set *pca*

and, at times, *par*. As such, the schema *UpdatePrincCat* ensures the set *arca* remains unchanged.

<i>UpdatePrincCat</i>	_____
<i>SetupRetained</i>	
<i>arca'</i> = <i>arca</i>	

In order to assign a principal to a particular category, we consider two cases. The first such case is in which a principal is being assigned to category that does not have any assigned permissions (i.e. for a category *c*, it is the case that $c \notin \text{dom } arca$). The second such case is in which a principal is being assigned to a category that already has assigned permissions (i.e. for a category *c*, it is the case that $c \in \text{dom } arca$). As such, the schema *AllocatePrincCat* involves an **if** clause with respect to the set of authorisations added to *par*.

<i>AllocatePrincCat</i>	_____
<i>UpdatePrincCat</i>	
$p? : P; c? : C$	
$v : \mathbb{F} \text{Auth}$	
$p? \mapsto c? \notin pca$	
$v = \text{if } c? \in \text{dom } arca$	
then $\{m : arca \mid \text{first } m = c? \bullet (p?, \text{second } m)\}$	
else \emptyset	
$pca' = pca \cup \{p? \mapsto c?\}$	
$par' = par \cup v$	

To handle the complementary case — removal of a principal from a given category — we define *RemovePrincCat*. In order to remove a principal from a category, we must consider whether the category has assigned permissions: if the category has no assigned permissions, the only change made is to the set *pca*; otherwise, the set *par* may also change (as reflected by the *IF* clause).

<i>RemovePrincCat</i>	_____
<i>UpdatePrincCat</i>	
$p? : P; c? : C$	
$d : \mathbb{F} C; v : \mathbb{P} \text{Auth}; pe : \text{Perm}$	
$p? \mapsto c? \in pca$	
$c? \mapsto pe \in arca \vee c? \notin \text{dom}(arca)$	
$d = \{n : pca \mid \text{first } n = p? \wedge \text{second } n \neq c? \bullet \text{second } n\}$	
$v = \{m : arca \mid$	
$\text{first } m = c? \wedge \text{second } m \notin \text{ran}(d \triangleleft arca) \bullet (p?, \text{second } m)\}$	
$par' = \text{if } (pe \in \text{ran}(d \triangleleft arca) \wedge v = \emptyset)$	
then <i>par</i>	
else $par \setminus v$	
$pca' = pca \setminus \{p? \mapsto c?\}$	

4.6 Example operations: categories

Categories are, by definition, at the heart of CBAC. Categories can be characterised as a class of entities that share some property.

In our model, the set of categories, *catset* captures the categories that exist in the current system. The schema *AddCat* allows us to add categories:

<i>AddCat</i>
<i>ModelRetained</i>
$c? : \mathbb{F}_1 C$
$c? \cap \text{catset} = \emptyset$
$\text{permset}' = \text{permset}$
$\text{principalset}' = \text{principalset}$
$\text{catset}' = \text{catset} \cup c?$

The schema *RemCat* is concerned with removing categories. Category removal needs to handle the cases where: the category has principals previously assigned; the category has no assigned principals; and the category has permissions assigned to it and, as such, can be found in *arca*.

<i>RemCat</i>
$\Delta MModel$
$c? : C$
$pe : \mathbb{F} Perm$
$c? \in \text{catset}$
$pe = \text{if } c? \in \text{dom } arca$
then $\{m : \text{permset} \mid c? \mapsto m \in arca\}$
else \emptyset
$pca' = pca \triangleright \{c?\}$
$arca' = \{c?\} \triangleleft arca$
$par' = par \triangleright pe?$
$\text{permset}' = \text{permset}$
$\text{principalset}' = \text{principalset}$
$\text{catset}' = \text{catset} \setminus \{c?\}$

Within a given category, members are associated with permissions that have been assigned to the category. As such, operations on categories often involve the sets *arca* and *par*. Thus, we define a schema, *UpdateCatPerms*, which ensures that *pca* remains unchanged:

<i>UpdateCatPerms</i>
<i>SetupRetained</i>
$pca' = pca$

We may consider the assignment of permissions to categories (as found in the set $arca$), where we consider how assignment of permissions applies to a category that does not have any assigned principals, i.e. for a category c , it is the case that $c \notin \text{ran } pca$. Here, we reason that the only set to be updated is $arca$ and so $v = \emptyset$. We may also consider the handling of category–permission assignment when the category already has assigned principals, i.e. for a category c , it is the case that $c \in \text{ran } pca$. Here, $arca$ is updated, as is the set of authorisations: par must change to reflect an assignment of permissions to principals found in the given category.

$CategoryPerms$ $UpdateCatPerms$ $c? : C; pe? : Perm$ $v : \mathbb{P} Auth$
$c? \mapsto pe? \notin arca$ $v = \{m : pca \mid second\ m = c? \bullet (first\ m, pe?)\}$ $arca' = arca \cup \{c? \mapsto pe?\}$ $par' = par \cup v$

The next schema provides a method to remove permissions from a category and update the authorisation set, par . In the first case, we deal with the removal of permissions from a category that does not have any principals assigned to it, and as a result, the only set to change is $arca$. This is in contrast to the case where the category has principals assigned to it. This difference is reflected in the need to update the authorisation set, par , if these permissions are the sole result of membership of the category in question, i.e. for a category c_1 , it is the case that $\forall pe : arca(c_1) \bullet pe \notin \text{ran}(\{c_1\} \triangleleft arca)$:

$RemoveCatPerms$ $UpdateCatPerms$ $c? : C; pe? : Perm; v? : \mathbb{P} Auth$
$(v? = \emptyset \wedge c? \notin \text{ran}(pca))$ $pe? \in permset$ $c? \mapsto pe? \in arca$ $arca' = arca \setminus \{c? \mapsto pe?\}$ $par' = par \setminus v?$

The following schema encapsulates an update or ‘swap’ on the permissions found within a category. Here, for a category, c_1 , with $c_1 \mapsto perm_1 \in arca$, if we wish to ‘swap’ the permission $perm_1$ with $perm_2$ so that we have $c_1 \mapsto perm_2 \in arca$, we can define this for a category with or without principals. In the case of a category without principals, we see changes to $arca$ only; for the other case, we see changes to both $arca$ and par .

<i>SwapCatPerms</i>
<i>UpdateCatPerms</i>
$c? : C$
$pe?, qe? : Perm$
$v, d, w : \mathbb{P} Auth$
$c? \mapsto qe? \in arca$
$c? \mapsto pe? \notin arca$
$v = \{m : pca \mid second\ m = c? \bullet (first\ m, qe?)\}$
$d = \{n : v; s : pca; t : arca \mid$
$first\ n = first\ s \wedge second\ s = first\ t \wedge second\ s \neq c? \bullet$
$(first\ n, second\ t)\}$
$w = \{m : pca \mid second\ m = c? \bullet (first\ m, pe?)\}$
$arca' = (arca \setminus \{c? \mapsto qe?\}) \cup \{c? \mapsto pe?\}$
$par' = ((par \setminus v) \cup d) \cup w$

5 Conclusion

We have presented a preliminary model of the Category-Based Access Control (CBAC) meta-model in terms of the schema language of Z. The model enables one to reason about potential extensions, to account for, for example, the dynamic, contextual nature of privacy as it relates to metadata.

Our problem of interest relates specifically to the issue of metadata collection as it applies to personal data, consent and dataveillance. As we have discussed, metadata collection informs the identity profiles companies create for their users [18].

This threat to users' privacy is entwined with the issue of data ownership on the Internet, where we see that users rarely have explicit ownership of their own data. To address this, we have identified a possible stepping stone in the form of the Solid proposal. The Solid proposal, as has been outlined, focuses on access control for users' data pods, whereby individuals may authorise granular access of their data as they please.

Our next task is to build upon our initial model to capture the Solid proposal's Web Access Control Lists. Subsequently, we will be in a position to build upon it, focusing on the specific problems faced due to collation of metadata. By reasoning about user data pods, we will be able to implement our own privacy notions that handle the complexities of inferences from metadata — allowing us to make further progress with respect to our main research question.

Acknowledgments

The authors would like to thank the anonymous reviewers for their helpful and constructive comments. Marine Eviette's research is funded by the EPSRC via the Centre for Doctoral Training in Cyber Security at the University of Oxford.

References

1. Lothar Fritsch. Privacy dark patterns in identity management. In *Open Identity Summit (OID)*, 5-6 october 2017, Karlstad, Sweden., pages 93–104. Gesellschaft für Informatik, 2017.
2. Arunesh Mathur, Gunes Acar, Michael J Friedman, Elena Lucherini, Jonathan Mayer, Marshini Chetty, and Arvind Narayanan. Dark patterns at scale: Findings from a crawl of 11k shopping websites. *Proceedings of the ACM on Human-Computer Interaction*, 3(CSCW):1–32, 2019.
3. Christoph Bösch, Benjamin Erb, Frank Kargl, Henning Kopp, and Stefan Pfattheicher. Tales from the dark side: Privacy dark strategies and privacy dark patterns. *Proceedings on Privacy Enhancing Technologies*, 2016(4):237–254, 2016.
4. Hanbyul Choi, Jonghwa Park, and Yoonhyuk Jung. The role of privacy fatigue in online privacy behavior. *Computers in Human Behavior*, 81:42–51, 2018.
5. Dalal Al-Azizy, David Millard, Nigel Shadbolt, and Kieron O’Hara. Deanonymisation in linked data: A research roadmap. In *World Congress on Internet Security (WorldCIS-2014)*, pages 48–52. IEEE, 2014.
6. Kevin Lewis. Three fallacies of digital footprints. *Big Data & Society*, 2(2):2053951715602496, 2015.
7. Messaoud Benantar. *Access control systems: security, identity management and trust models*. Springer Science & Business Media, 2005.
8. Michael Carl Tschantz and Jeannette M Wing. Formal methods for privacy. In *International Symposium on Formal Methods*, pages 1–15. Springer, 2009.
9. Maribel Fernández, Murat Kantarcioglu, and Bhavani Thuraisingham. A framework for secure data collection and management for Internet of Things. In *Proceedings of the 2nd Annual Industrial Control System Security Workshop (ICSS 2016)*, pages 30–37. ACM, 2016.
10. Steve Barker. The next 700 access control models or a unifying meta-model? In *Proceedings of the 14th ACM Symposium on Access Control Models And Technologies (SACMAT 2009)*, pages 187–196. ACM, 2009.
11. J. M. Spivey. *The Z Notation: A Reference Manual*. Prentice Hall, 1992.
12. Jim Woodcock and Jim Davies. *Using Z: specification, refinement, and proof*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996.
13. D. J. Power, M. A. Slaymaker, and A. C. Simpson. On formalising and normalising role-based access control systems. *The Computer Journal*, 52(3):303–325, 2009.
14. Jeffrey Pomerantz. *Metadata*. MIT Press, 2015.
15. Duncan Hodges, Sadie Creese, and Michael Goldsmith. A model for identity in the cyber and natural universes. In *2012 European Intelligence and Security Informatics Conference*, pages 115–122. IEEE, 2012.
16. Roger Clarke. Information technology and dataveillance. *Communications of the ACM*, 31(5):498–512, 1988.
17. Roger Clarke. The digital persona and its application to data surveillance. *The information society*, 10(2):77–92, 1994.
18. José Van Dijck. Datafication, dataism and dataveillance: Big data between scientific paradigm and ideology. *Surveillance & society*, 12(2):197–208, 2014.
19. William A Parent. Privacy, morality, and the law. *Philosophy & Public Affairs*, pages 269–288, 1983.
20. Privacy International. What is privacy. <https://www.privacyinternational.org/explainer/56/what-privacy>, October 2017.
21. Roland Garrett. The nature of privacy. *Philosophy Today*, 18(4):263–284, 1974.

22. Ruth Gavison. Privacy and the limits of law. *The Yale Law Journal*, 89(3):421–471, 1980.
23. Marshall Van Alstyne, Erik Brynjolfsson, and Stuart Madnick. Why not one big database? principles for data ownership. *Decision Support Systems*, 15(4):267–284, 1995.
24. Neil Foshay, Avinandan Mukherjee, and Andrew Taylor. Does data warehouse end-user metadata add value? *Communications of the ACM*, 50(11):70–77, 2007.
25. John Perry Barlow. A declaration of independence for cyberspace. <https://www.eff.org/cyberspace-independence>, 1996.
26. Guy Zyskind, Oz Nathan, and Alex Pentland. Enigma: Decentralized computation platform with guaranteed privacy. *arXiv preprint arXiv:1506.03471*, 2015.
27. Max Van Kleek, Daniel Alexander Smith, Nigel Shadbolt, et al. A decentralized architecture for consolidating personal information ecosystems: The webbox. *Proc Workshop on Personal Information Management (PIM)*, pages 177–189, 2012.
28. Andrei Vlad Sambra, Essam Mansour, Sandro Hawke, Maged Zereba, Nicola Greco, Abdurrahman Ghanem, Dmitri Zagidulin, Ashraf Aboulnaga, and Tim Berners-Lee. Solid: a platform for decentralized social applications based on linked data. Technical report, Technical Report, MIT CSAIL & Qatar Computing Research Institute, 2016.
29. Aafaf Ouaddah, Anas Abou Elkalam, and Abdellah Ait Ouahman. Fairaccess: a new blockchain-based access control framework for the internet of things. *Security and Communication Networks*, 9(18):5943–5964, 2016.
30. Michael Nofer, Peter Gomber, Oliver Hinz, and Dirk Schiereck. Blockchain. *Business & Information Systems Engineering*, 59(3):183–187, 2017.
31. Ravi S Sandhu and Pierangela Samarati. Access control: principle and practice. *IEEE communications magazine*, 32(9):40–48, 1994.
32. Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data: The story so far. In *Semantic services, interoperability and web applications: emerging concepts*, pages 205–227. IGI Global, 2011.
33. Steve Barker. Personalizing access control by generalizing access control. In *Proceedings of the 15th ACM symposium on Access control models and technologies*, pages 149–158, 2010.