PURPOSE ORGANISATION

Wynand van Staden[1], **Martin Olivier**[2]

[12]Information and Computer Security Architectures Research Group
Department of Computer Science
University of Pretoria

[1]+27-11-489-2525, wvs@adam.rau.ac.za

[2]http://www.mo.co.za

ABSTRACT

One of the principles of privacy management is making use of purposes for indicating why data is stored, and also why access to information is requested during the manipulation of the stored data.

Many Privacy Enhancing Technologies (PETs) – such as the Hippocratic Database, Platform for Privacy Preferences (P3P), and Platform for Enterprise Privacy Preferences (EP3P) – make use of purposes as part of their basic operation. Little work has, however, been done on the proper organisation of purposes. Some of the work that has been done suggests that purposes can be placed in a hierarchy. This form of organisation is done to facilitate the logical grouping of purposes. Since purposes are critical to the operation of PETs – to ensure proper privacy management – they must be carefully examined so as to ensure that the way in which they are organised, adds more value to the act of organising and using than that of a logical grouping.

This paper considers the organisation of purposes by arguing that they can be placed in a lattice, and that the use of such a purpose lattice offers improved protection of privacy. It does so by examining the already proposed idea of purpose hierarchies, and extending this idea by placing purposes in a lattice. It considers the representation of purposes in the lattice, the relationship between purposes in the lattice, and also how the relationships presented in a lattice can be used to accommodate a more flexible use of purposes for privacy management.

The paper provides a foundation for future research of purpose organisation and management. Furthermore it introduces the concept of compound purposes, that is, purposes that are defined in terms of other purposes are presented. The use and implication of compound purposes are, however, not yet explored.

1 INTRODUCTION

A fundamental principle of proper privacy management is the use of purposes in two phases. The first phase is the purpose specification phase, during which an entity which stores information indicates the reasons for which the information is stored – their intended purpose with the information. The second phase, is the act of verifying that access to information is restricted based on the intended use as indicated by the entity requesting the information. These two phases are embodied in the OECD's [1] *purpose specification* and *use limitation* principles. In this paper we will use the term *purpose* to refer to the intended use for which an organisation has collected data and the term *reason* to refer to the intended use for which a user requests access to the data. Assume data item x has been collected for purpose P and some user u wants to access the data and supplies reason R. In the classical case access will only be given if u has been authorised to access x using the usual security mechanisms *and* $R = P$. In general, this latter requirement can be relaxed to state that access should be granted if the reason supplied is sufficient. We will denote this by writing $R \geq P$. In the simplest case, a reason is merely a purpose used during data access. It will, however, be argued below that purposes and reasons are not identical.

Many Privacy Enhancing Technologies (PETs) already use the notion of purpose to protect information as part of their basic operation, with the Hippocratic Database [2] as one of the prime examples. In spite of this, purposes have been more or less regarded as a "flag" which can be used to determine authorisation. Other Privacy Enhancing Technologys (PETs) such as Enterprise Privacy Authorisation Language (EPAL) [3] have included the organisation of purposes into hierarchies to allow the grouping of related purposes. The grouping serves to facilitate authorisation decisions. Purposes in EPAL are hierarchical elements [3], meaning that parents in the hierarchy can only be used if that parent's children can be used.

The proposal in this article differs from the EPAL notion in the following ways. Firstly, parent nodes are not merely groupings of children nodes, parent nodes are more general purposes, and children nodes represent more specific purposes. Secondly, a child purpose is said to dominate the parent purpose, thus a child purpose subsumes all it's parent purpose's semantics, thus the parent purpose can be used if any one of the child purposes can be used.

It is well known that many security-related attributes are best arranged in hierarchies or lattices to reduce the number of attributes that have to be explicitly managed [4]. Such organisation simplifies the work of the security officer, because inherent relationships between attributes can be exploited to simplify the task. This is, for example, one of the main reasons for the popularity of Role Based Access Control (RBAC) [5]. The relationships between such attributes can, however, also have unexpected consequences and the semantics of such structuring needs to be properly understood.

This paper considers the organisation of purposes in a lattice. The organisation of reasons in a lattice will follow as a logical consequence. The semantics of compound purposes and compound reasons are considered, as well as the relative ordering of such compound purposes and reasons.

Placing purposes in a lattice introduces several properties which can be used to effectively enhance the usefulness of purposes beyond simple "flags". Moreover, the particular representation of purposes as well as their placement within the lattice can be used to effectively exchange

information about purposes between different organisations.

In order to introduce the concept of proper purpose management, the rest of the paper is structured in the following manner: Section 2 provides some background on the general organisation of purposes. Section 3 indicates the representation of purposes. Section 4 indicates how purposes can then be placed in a lattice (using the representation provided in section 3). Section 4.2 introduces the concept of overriding purposes, and the relative suitability of purposes. Section 5 discusses weak and strong purposes, and placing purposes in a lattice. Section 6 introduces the concept of compound purposes, by discussing the notation used to describe a compound purpose. Finally section 7 concludes the paper.

2 BACKGROUND

Many PETs already include purposes as part of their basic operation. This section provides more detail on these PETs, and indicates how they represent purposes.

It has become common practice for enterprises to record information about the people that visit their websites. This information can include the name, age, gender, and other Personal Identifiable Information (PII) about the visitor. The information may be used to conclude a transaction, or simply to be able to provide the visitor with information regarding some product, or service rendered. In many cases the information is required before allowing the visitor to download information.

The information is invariably also used for data mining that allows the enterprise to target market a specific demographic group. The unavoidable negative experiences that can plague the casual, or intentional visitor to a website as the result of leakage of information, has prompted a more responsible management of PII.

The World Wide Web Consortium (W3C) has, as a result of this, developed the P3P [6], which allows an enterprise to indicate their purpose for storing information. The P3P is, however, simply a policy language and not a technology that can be used to enforce the technology [7]. The language does allow the enterprise to indicate for what purpose information is being recorded – this allows the visitor to decline to provide his PII, if he does not agree with the policy presented by the company. P3P provides several standard purposes which can be used, and also allows for the definition of purposes specific to the enterprise. P3P does however not support a complex ordering or organisation of purposes.

Recognising the shortcomings of the P3P, Ashley et al [7], presented the EP3P, which is intended to be a policy specification standard used to construct policy specification languages. Such a policy specification language can be used to create a policy for each PET used by the enterprise.

The first implementation of the standard is called EPAL [3]. EPAL takes the organisation of purposes one step further and places purposes in a hierarchy. Previous technologies – Component Based Architecture for Secure Data Publication (CBASDP) [8] – have also proposed the organisation of purposes into hierarchies. The EPAL uses hierarchies, however, to indicate that a child purpose can be considered a more special, or focused case, than the more general parent purpose.

The Hippocratic Database [2] concept, stores purposes as part of the database schema, which allows the Database Management System (DBMS) to control access to information. Purposes in this case, are simply strings which are stored as part of the schema. When data is manipulated via a query, the user who is requesting the access to the information has to provide a reason for wanting the information. The DBMS is thus capable of controlling access to the information by verifying that the information is requested for the purpose for which it is stored.

Purposes have thus already been employed successfully by PETs. Many of the PETs place purposes in hierarchies, and EPAL, takes the concept one step further. In the following section we consider the representation of purposes.

3 THE REPRESENTATION OF PURPOSES

Using purposes and reasons during the verification phase of access control using purposes inevitably requires that purposes and reasons must be compared. Since simple reasons are identical to purposes, this requires one purpose to be comparable to another purpose. This act of comparing does not assume that all purposes are comparable, but that some partial order exists. The comparability of purposes depends heavily on the way in which purposes are represented. This section briefly considers the representation of purposes.

Representing purposes as strings allows humans to easily read and understand purposes. The negative aspect of using such an approach is that it makes it difficult to reason about purposes computationally. This can be easily solved by associating an *id* with each purpose. The *id* can then be used to identify each purpose, and also to compare purposes with other purposes for equality.

Exchanging purposes between different enterprises is problematic if the enterprises do not agree on a common set of purposes or on a common set of *ids*.

Associating an *id* with each purpose makes it computationally more efficient to use purposes. The problems just identified can therefore be solved in one of three ways. Firstly, a translation scheme can be presented to allow the translating of one set of purposes into another equivalent set.

The second method requires that the generation of *ids* are done in a similar way across different enterprises. One such way is requiring that purpose *ids* are generated from the string representation of purposes. The string representation can be taken from verbs in a standard dictionary. There are however, other aspects to consider for this, and further discussion of this topic is deferred for another paper.

The third and final method, requires that a standard set of purposes are created and that enterprises make use of the standard set of purposes, as well as the standard set of *ids*. This requirement is no different from standard medical practice in which there is a code for each specific diagnosis made.

Throughout the course of this paper, it is therefore assumed that a finite set of purposes is used, and that each one of these purposes are labelled uniquely. Thus, for example, a purpose such as “send email to client” may be labelled as P_1, while another purpose such as “send billing

information" may be labelled as P_2. When a subject thus requests access to information, he can simply supply the reason he wants access as, for instance, P_2.

4 PLACING PURPOSES IN A LATTICE

It has already been indicated, in this paper, and in work done elsewhere such as EPAL, that the placement of purposes in a hierarchy provides a more effective use of purposes. Placing purposes in a lattice allows for much more power in determining authorisation. This power is afforded by several aspects which will be covered in more detail below.

4.1 The relationship between purposes

Suppose that an enterprise stores the email addresses of clients. These addresses can be used to send out general news about the enterprise, send billing information, and catalogues about products on offer. A client can thus provide his email address and the DBMS will only provide his email address to a subject querying the database who supplies an appropriate reason for wanting the data.

It is intuitively clear that "sending email" and "send billing information" are in this case related, since the act of sending billing information requires an email address, and the act of sending an email also requires an email address. In fact, "sending billing information" is in this case a much clearer indication of purpose than that of "sending email". Not surprisingly, this form of relationships between purposes produces a hierarchy. Consider figure 1. Sending email

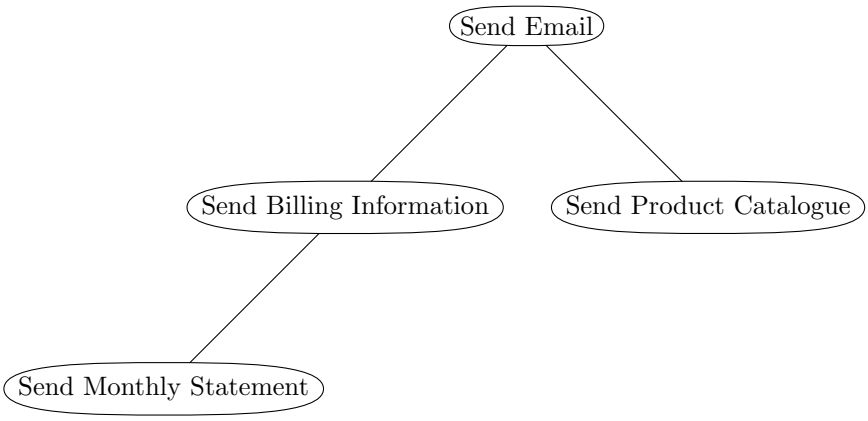

Figure 1: A simple purpose hierarchy

is a parent node in the hierarchy to the sending of billing information purpose and the sending of catalogues. However, the sending of catalogues and the sending of billing information are seemingly different purposes altogether (one may be considered a marketing action, whilst the other is part of the completion of a transaction). That is to say, no relationship exists between them, other than sharing the same parent node.

Of more importance however is the fact that certain nodes in the hierarchy are *weaker* than other nodes. This concept is discussed in more detail in the following section.

4.2 The relative suitability of purposes

It is clear that the ordering of purposes in a hierarchy, implies that some relationship exists between the purposes. If the ordering of roles in a role graph [9] is used as a model, it is possible to state that purposes located near the root of the hierarchy, are more general purposes – in fact the reverse of the role graph model. Purposes located near the leaves are then the most specific purposes for storing data.

Purposes that are not very specific, are used to accomplish more general tasks, and are therefore termed *weak*. On the other hand, purposes that are very specific, can only be used for specific tasks and are therefore termed *strong*.

Purposes that are stronger than other purposes can also be considered to subsume all those purposes that they are stronger than, in the same fashion in which a particular role can subsume other roles in the role graph. A stronger purpose thus also implies the weaker one.

Definition 1 *When a purpose P_1 is considered to be weaker than another purpose P_2, it is indicated by writing $P_1 \leq_p P_2$. This indicates that if reason $R = P_2$ is given, it is a good enough reason – suitable – to get access to information stored for purpose P_1.*

5 THE PURPOSE LATTICE

Although adequate, the hierarchy of purposes presented here is not sufficient. Suppose for instance that a law enforcement agency requests access to the email addresses of all the clients stored in the database. Attempting to specify each *strongest* purpose for each branch of the hierarchy can be quite cumbersome for a hierarchy sporting hundreds of entries.

It is therefore necessary to provide a *strongest* purpose, a possible ultimate purpose for storing information – thereby also requiring possible ultimate reason for wanting to access information. Revisiting figure 1 and adding a strongest purpose to it produces figure 2. The graph presented here contains a partially ordered list with a lower and upper bound, or a lattice. Organisation of the purposes into a lattice can in this case provide a closed structure, like that of a role graph.

A Purpose Lattice (PL) is defined to be a directed acyclic graph containing all the purposes the enterprise uses for storing and accessing data. Nodes in the PL represent purposes, and a directed edge from node A to node B in the PL means that B is a stronger purpose than purpose A (see definition 1).

Any purpose for which a directed edge exists from another purpose is said to be stronger than that purpose or that it dominates the other purpose, and by implication, if that particular purpose is supplied as a reason for wanting to access data, it is at least as good to gain access to data as the purpose for which the data was stored.

The weakest purpose in the PL is of course a reason which is dominated by all the purposes in the PL.

Definition 2 *A weakest purpose is any purpose such that if x_i is the weakest purpose, then $\forall x_i, x_j \in PL, x_i <_p x_j, i \neq j$.*

The strongest purpose conversely is that purpose that dominates all other purposes.

Definition 3 *A strongest purpose is any purpose such that if x_i is the strongest purpose, then $\forall x_i, x_j \in PL, x_j <_p x_i, i \neq j$.*

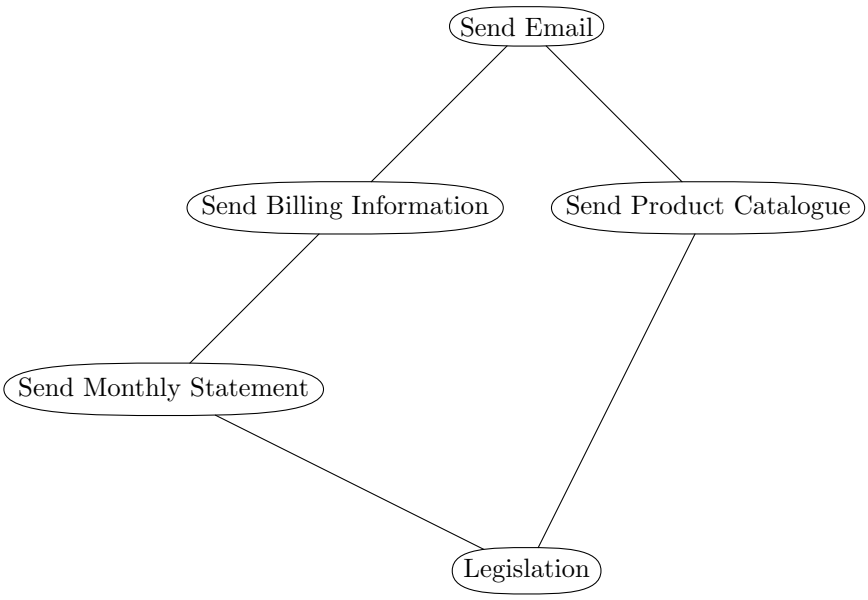

Figure 2: Purposes Lattice

Placing purposes in a lattice like this, allows the enterprise to specify all the purposes for which data is stored in a particular database.

It also allows the specification of more complex purposes for storing data, or the specification of *compound purposes*

6 COMPOUND PURPOSES

A *compound purpose*, as the name suggests, is a purpose that is compounded from other purposes, that is, it is a purpose that is defined in terms of the conjunction or disjunction of other purposes; a compound reason is a reason that is defined in terms of the conjunction or disjunction of other reasons.

This allows the combination of purposes that seem logically disconnected, but may be required as part of a particular business rule. For example, a particular purpose for storing an address may be for sending a parcel, or for sending a product catalogue. Consider figure 3, which illustrates a sample purpose lattice. The reason for making use of purposes in the first place is to ensure that data is only released when the reason for using it is sufficient.

By making use of compound purposes it is possible to indicate that either one reason is enough to get access to data stored for one of many purposes, or that several reasons must be supplied in order to get access to the data.

The reasons provided by the user when wanting data can be encoded as part of their access token, or it can be supplied by the user as part of the query. In either case it becomes possible to associate a particular set of reasons (by using compounds) with a particular user. It also becomes possible to define reasons such as “because Alice said I could” for use when requesting access to information.

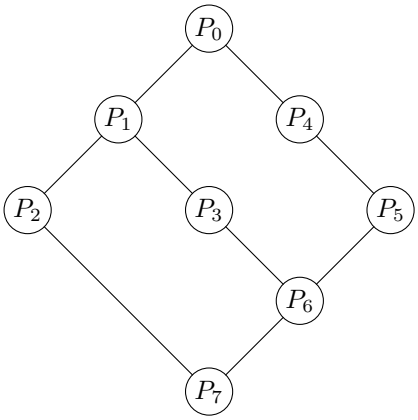

Figure 3: Example purpose lattice

6.1 Or compounds

An or-compound is indicated by using the symbol $+$ as a binary operator between two or more operands. In particular a distinction is made between the semantics of the $+$ operator for purposes, and the $+$ operator for reasons. The former is indicated as $+_p$, while the latter is indicated as $+_r$. The distinction will be clarified later in this section.

Consider the purposes P_1 and P_4, with reasons $R_1 = P_1$ and $R_2 = P_4$. Now suppose that a particular datum is stored for purpose $P_1 +_p P_4$. Thus, any reason provided by the user that is either stronger than P_1 or P_4 will be sufficient to gain access to the datum. Thus,

$$\begin{aligned} R_1 &\geq P_1 +_p P_4 \\ R_2 &\geq P_1 +_p P_4 \end{aligned}$$

Obviously any reason that is stronger than either R_1 or R_2 will also be sufficient.

It is now possible to provide a compound reason for wanting access to information. For example, a user might indicate that they want client information because they want to send parcels to some, and product catalogues to others.

$$R_1 +_r R_2 \geq P_1 +_p P_2$$

The important distinction between $+_p$ and $+_r$ becomes clear at this point. If data is stored for $P_1 +_p P_2$, then any reason stronger than either P_1 or P_2 can be given in order to gain access to the stored data.

If the reason provided when requesting access to data is $R_1 +_r R_2$, then the DBMS must not provide all the data which can be used for either R_1 or R_2, as this potentially divulges information that was stored for a purpose for which R_2 was sufficient but not R_1, or vice versa. Thus the DBMS must only release data which can be used for both R_1 and R_2.

$$\begin{aligned} R_1 +_r R_2 &\not\geq P_1 \\ R_1 +_r R_2 &\not\geq P_2 \end{aligned}$$

A second binary operator, is the and-compound operator, discussed in the following section.

6.2 And compounds

An and-compound (indicated with the symbol $\cdot$) indicates that data is stored for two or more purposes simultaneously, and will consequently be used for reasons that satisfy all the purposes simultaneously. For instance, if a particular datum is stored for purpose $P_1 \cdot P_2$, then only reasons that are simultaneously stronger than $P_1 \cdot P_2$ can be given in order to be granted access to data. For example, an email address may be stored for purchase notification and billing information. Thus, only when a user wants to send a purchase notification and billing information will the email address be released.

If $R_1 = P_1$ and $R_2 = P_2$ then,

$$\begin{aligned} R_1 &\not\geq P_1 \cdot_p P_2 \\ R_2 &\not\geq P_1 \cdot_p P_2 \\ R_1 +_r R_2 &\not\geq P_1 \cdot_p P_2 \\ R_1 \cdot_r R_2 &\geq P_1 \cdot_p P_2 \end{aligned}$$

The reason $R_1 +_r R_2 \not\geq P_1 \cdot_p P_2$, stems from the fact that the compound reason indicates that the data will be used for either one of the reasons provided, whereas the compound purpose specifies that it must be used for both.

7 SUMMARY AND FUTURE WORK

The use of purposes in a privacy context can greatly enhance the capabilities of the implementation of the access control policy. The Hippocratic DBMS takes a step in this direction by requiring that the purpose for which data is stored be made part of a databases schema. Subjects then request access to information by providing their purpose for the data.

Purposes, then, form a critical part of the Hippocratic DBMS. It is for this reason that these purposes must be organised properly, in order to make proper use of them. The current model for the Hippocratic database, unfortunately, does not provide a proper model for the management of purposes, despite the fact that the organisation of purposes into a hierarchy has been suggested.

This paper proposed that purposes can be organised by using a lattice. Purposes placed in the lattice are considered to be a subset of a finite set of purposes, and each purpose can therefore be labelled uniquely. Making use of the hierarchy formed by lattice, more general purposes are placed near the upper bound of the lattice. Children purposes converge on the lower bound and are considered to be more specific, or better purposes for wanting to access information. Thus users can provide better purposes in order to access information stored for more general purposes.

Making use of the proposed lattice and the relative suitability of purposes will allow the Hippocratic Database to make effective use of purposes.

A notation for combining purposes to form compound purposes was also provided.

Future work for purpose management includes the investigation of using techniques provided by Nyanchama and Osborn [10, 11] to integrate different purpose lattices.

The possibility of indicating negative purposes and reasons, for example “email addresses are not stored for sending junk mail”, are currently also being investigated.

References

[1] OECD guidelines on the protection of privacy and transborder flows of personal data. Technical report, Organisation for Economic Co-operation and Development, 1980.

[2] Rakesh Agrawal, Jerry Kiernan, Ramakrishan Srikant, and Yirong Xu. Hippocratic databases. In *Proceedings of the 28th VLDB Conference*, Hong Kong, China, 2002.

[3] Paul Ashley, Satoshi Hada, Günter Karjoth, Calvin Powers, and Matthias Schunter. Enterprise privacy authorisation language (EPAL 1.1). Technical report, International Business Machines Corporation, 2003.

[4] Fausto Rabatti, Elisa Bertino, Won Kim, and Darrel Woelk. A model of authorisation for next-generation database systems. In *ACM Transactions on Database Systems*, volume 16, pages 88–131. ACM, March 1991.

[5] Ravi S Sandhu. Role-based access control. *Advances in Computers*, 46, 1998.

[6] The platform for privacy preferences (P3P1.1) specification. Technical report, W3C, Available at http://www.w3.org/TR/2004/WD-P3P1, 2004.

[7] Paul Ashley, Satoshi Hada, and Günter Karjoth. E-p3p privacy policies and privacy authorisation. In *WPES'02*, Washington, November 2002.

[8] P. Bonatti, E. Damiani, S. de Capitani, and P. Samarati. A component-based architecture for secure data publication. In *Proceedings of the 17th Annual Computer Security Applications Conference*, page 309. IEEE Computer Society, 2001.

[9] Matunda Nyanchama and Sylvia Osborn. Access rights administration in role-based security systems. *Database Security VIII: Status and Prospects*, August 1994.

[10] Matunda Nyanchama and Sylvia Osborn. The role graph model and conflict of interest. *ACM Trans. Inf. Syst. Secur.*, 2(1):3–33, 1999.

[11] Sylvia Osborn. Integrating role graphs: a tool for security integration. *Data and Knowledge Engineering*, pages 317–333, 2002.