A Purpose-Based Access Control Model

Naikuo Yang Howard Barringer Ning Zhang
School of Computer Science
University of Manchester
Manchester, M13 9PL
{yangn, hbarringer, nzhang}@cs.man.ac.uk

Abstract

Achieving privacy preservation in a data-sharing computing environment is a challenging problem. The requirements for a privacy preserving data access policy should be formally specified in order to be able to establish consistency between the privacy policy and its purported implementation in practice. Previous work has shown that when specifying a privacy policy, the notion of purpose should be used as the basis for access control. A privacy policy should ensure that data can only be used for its intended purpose, and the access purpose should be compliant with the data's intended purpose. This paper presents a mechanism to specify privacy policy using VDM. The entities in the purpose-based access control model are specified, the invariants corresponding to the privacy requirements in privacy policy are specified, and the operations in the model and their proof obligations are defined and investigated.

1 Introduction

With the widespread use of information technology in all walks of life, personal information is being collected, stored and used in various information systems. Achieving privacy preservation in these data-sharing environments has become a major concern. Indeed, privacy is one of the major issues to be handled in many environments, such as the domain of health care [3]and e-Commerce [1, 2].

The Organisation of Economic Co-operation and Development (OECD) guidelines for data protection [16] and Federal Trade Commission (FTC) Fair Information Practice (FIP) principles [8] provide general privacy requirements that organisations should comply with. In parallel to these definitions, some organisations have also published privacy policies, which promise privacy protection practices on data collection, use and disclosure, although these practices may not be implemented. The problem is amplified if personal data is used not only within the organisations that collected the data, but also by other external organisations, such as partner organisations or managing authorities with a legitimate need to access the data. To establish consistency between privacy policy and its purported implementation, privacy protection requirements should be formally specified.

Platform for Privacy Preferences (P3P) [14] is a notable approach for formally specifying privacy policy by service providers. P3P provides a way for a web site to encode its data collection in a machine-readable format known as a P3P policy, which can be compared against a user's privacy preferences. While P3P provides a mechanism for ensuring that users can be informed about privacy policies before they release personal information, it does not provide a mechanism for making sure sites act according to their stated policies. Some other existing approaches [9, 13] for specifying privacy policies are also incomplete, either there is no entity of purpose, or there is no formal description of operations and authorisation mechanisms.

Since privacy policies are concerned with the purposes that a data object is used for rather than the actions that users perform on the data object, traditional access control models cannot easily achieve privacy protection, and the notion of purpose [7] should play a major role in access control model to protect privacy.

This paper focuses on the notion of purpose in a privacy-preserving access control model, and presents a mechanism to describe privacy policy using Vienna Development Method (VDM) [5, 12]. A framework for required elements of a privacy policy is presented, including the invariants corresponding to the privacy requirements and the operations. This ensures that the privacy policy has a clear and unambiguous interpretation.

In detail, the paper is organised as follows. Related work is discussed in Section 2. Section 3 introduces the entities of a purpose-based access control model. Section 4 specifies privacy invariants corresponding to the privacy requirements in a privacy policy. Section 5 specifies the operations in the model and their proof obligations. Finally, Section 6 concludes the work.

0-7695-2876-7/07 $25.00 © 2007 IEEE
DOI 10.1109/IAS.2007.29

IEEE computer society

2 Related Work

Previous work in the area of privacy preservation can be classified into two categories: extending the basic access control model for privacy preservation or specifying privacy requirements.

The Hierarchical Privacy-Sensitive Filtering (HPSF) model [15] was proposed to protect personal privacy of a data owner in relational databases, through defining privacy-sensitive levels for data items. A data owner specifies privacy-sensitive levels (PSL) for his/her data items, and each data user is assigned with a user privacy-sensitive level (UPSL). A user's data access is only allowed if the PSL of the data item is no higher than the UPSL of the user. This approach uses the similar concept as security levels in the multi-level security (MLS) model [4] which mainly aims at preserving the confidentiality of the data. Because privacy-sensitive levels of data usage are different from confidentiality levels, and may vary from individual to individual, it is difficult to define persistent PSLs and UPSLs.

A task-based access control model was proposed by Fischer-Hübner et al [9]. Data can only be accessed in a controlled manner by executing a task. Privacy invariants, privacy constraints and information flow rules are specified. The task is modeled as a state transition machine, and those invariants, constraints, and information flow rules will be checked during state transitions. The major contribution is two privacy requirements that it illustrated - *necessity of data processing* and *purpose binding* - and demonstrated how a privacy policy may be enforced. However, because access control is based on tasks performed by users, this approach lacks of scalability.

The Privacy-Aware Role-Based Access Control model (PARBAC) [10] was proposed for enforcing privacy policies within an organisation. When making access decision for a user, not only are the roles and permissions of the user considered, but also the business purpose of the operation and the privacy policy that pertains to that user are checked. PARBAC goes beyond traditional access control models in that it not only provides system security from an organisation's perspective, but also protects privacy from a data owner's standpoint. However, this approach is built upon the unconditional trust in organisations that collected data. Besides, although it introduces purposes related to data objects, it doesn't give a systematic entity model.

By extending Flexible Authorisation Framework (FAF) [11] with grantors and obligations, Karjoth et al proposed a model for authorisation management and access control in privacy protecting systems [13]. The Authorisation Specification Language (ASL) framework with the extension of grantors and obligations is used to formalise the privacy policy. The specified privacy policy can serve as the basis of internal access control system. The privacy model entities and the privacy requirements were presented, which provides a complete view of privacy preserving access control system. However, this work doesn't give specification of entities.

A privacy preserving access control approach [6, 7] was proposed based on the notion of purpose. An intended purpose is defined for each data object specifying the intended usage of that data object. The access purpose, on the other hand, is defined for each data access specifying the intentions for which a given data object are accessed. Only when an access purpose is compliant to its intended purpose is the access allowed. This approach provides scalability in a privacy-preserving access control system.

The work presented in this paper is built on this purpose-based approach to privacy preservation. It presents a mechanism for specifying entities and describing a privacy policy in terms of rules in standard VDM, and this enables a comparison of these rules against operations.

3 Model Entities

In a basic access control system, three entities are used: subjects, objects, and operations. For the system to be able to support privacy-preserving data accesses, entities that can be used by data owners to state their privacy requirements and by the system to enforce these requirements should be included. These entities include the privacy-sensitive levels of data objects, the consent of data owner, and the purposes of data usages and data accesses.

This section specifies the entities in our model.

3.1 An Object Data Model

Data objects are typically organised into different object types. An object type corresponds to a set of data objects that satisfy some common properties. For example, data collected from a patient in a medical care context can be classified into types of registration data, treatment history, etc. Classifying data objects into object types allows us to define and administer intended usages and necessary accesses in terms of object types instead of individual data objects, thus making the model more scalable.

Let *Object* denote a set of objects, and let *Type* denote a set of object types. Object type attributes and attribute values can then be defined.

Definition 1 (Object Type Attributes) *denoted as* $TypeAttr$*, are a set of attributes associated with an object type, and these attributes describe the properties for the collection of, and access to, this type of objects.*

Definition 2 (Object Type Attribute Values) *denoted as* $AttrValue$*, are a set of possible values for the object type attributes.*

An object data model is concerned with how the data objects are organised and how they are associated with object type attributes.

Definition 3 (Object Data Model)

$$\begin{aligned} ObjectDataModel :: object &: 2^{Object} \\ type &: 2^{Type} \\ typeAttr &: 2^{TypeAttr} \\ attrValue &: 2^{AttrValue} \\ TypeOf &: Object \rightarrow Type \\ AttrOf &: Type \rightarrow 2^{TypeAttr} \\ ValueOf &: Object \times TypeAttr \rightarrow \\ & \quad AttrValue \end{aligned}$$

inv mk-$ObjectDataModel$(*o, t, ta, av, To, Ao, Vo*) $\triangleq$
(dom $To = o \wedge$ rng $To \subseteq t$) $\wedge$
(dom $Ao = t \wedge$ rng $Ao \subseteq 2^{ta}$) $\wedge$
(dom $Vo = o \times t \wedge$ rng $Vo \subseteq av$)

The invariant for $ObjectDataModel$ states that *TypeOf* is a total function giving the type associated with each object, *AttrOf* is a total function giving the type attributes associated with each type, and *ValueOf* is a total function giving the value of the attributes associated with objects.

The object data model *OM* in our system, of type $ObjectDataModel$, can then be represented as a tuple ⟨*object, type, typeAttr, attrValue, TypeOf, AttrOf, ValueOf*⟩.

3.2 Users and Roles

Users are active entities in a system. The roles reflect the responsibilities of positions or job descriptions in the context of an organisation, e.g. therapist, registration staff, or billing staff in a medical care scenario.

Let *User* denote the set of users. Let *Role* denote the set of roles. *UserRole*: $User \leftrightarrow Role$ is the relation between users and roles.

A user may be assigned with many roles, but the user may not exercise all his roles at the same time. The roles that a user is currently exercising are "active" roles.

Active Roles AR: *User*$\rightarrow 2^{Role}$ is a function that returns the roles for which a user is active.

To specify the set of users to grant access purpose. The concept of *conditional role* was then introduced. It is based on the notion of *Role Attributes* and *System Attributes*.

Definition 4 (Role Attributes) *denoted as RoleAttr, are a set of properties related to the grant of access purpose.*

Every role $r \in Role$ is associated with a set of role attributes, e.g. the specialty of therapists in a medical care scenario.

RoleAttrOf: $Role \rightarrow 2^{RoleAttr}$ is a function that returns the set of role attributes of a role.

Let *RoleAttrValue* denote the set of possible role attribute values.

RoleAttrValueOf: $Role \times RoleAttr \rightarrow RoleAttrValue$ is a function giving the value of role attributes associated with a role.

Definition 5 (System Attributes) *denoted as SysAttr, are properties about the context of access control system.*

Let $SysAttrValue$ denote the set of all possible system attribute values.

SysAttrValueOf: $SysAttr \rightarrow SysAttrValue$ is a function giving the value of the system attributes in a system.

Definition 6 (Conditional Role) *refers to a role with some conditions attached to it.*

$$\begin{aligned} CondRole :: r &: Role \\ cond &: RoleAttrValue \times SysAttrValue \rightarrow B \end{aligned}$$

where B is the boolean set, and *cond*: $RoleAttrValue \times SysAttrValue \rightarrow B$ is a truth-valued function.

$CR : 2^{CondRole}$ is used to hold the set of conditional roles in a system.

Current Conditional Role CCR: *User* $\rightarrow$ *CR* is a function that returns the conditional role the user currently exercises.

Definition 7 (Role Model)

$$\begin{aligned} RoleModel :: role &: 2^{Role} \\ user &: 2^{User} \\ UserRole &: User \leftrightarrow Role \\ AR &: User \rightarrow 2^{Role} \\ roleAttr &: 2^{RoleAttr} \\ roleAttrValue &: 2^{RoleAttrValue} \\ RoleAttrValueOf &: Role \times RoleAttr \rightarrow \\ & \quad RoleAttrValue \\ sysAttr &: 2^{SysAttr} \\ sysAttrValue &: 2^{SysAttrValue} \\ SysAttrValueOf &: sysAttr \rightarrow sysAttrValue \\ CR &: 2^{CondRole} \\ CCR &: user \rightarrow CR \end{aligned}$$

The role model *RM* can be represented as a tuple ⟨*role, user, UserRole, AR, roleAttr, roleAttrValue, RoleAttrValueOf, sysAttr, sysAttrValue, SysAttrValueOf, CR, CCR*⟩.

3.3 Purpose

Data is collected for certain purposes. For example, for medical care, data may be collected for registration or diagnosing. Each data access also serves a certain purpose.

Definition 8 (Purpose) *denoted as Purpose, is the intention of data collection or data access.*

Purposes, depending on their association with objects and subjects, may be called intended purposes or access purposes, respectively.

Definition 9 (Intended Purpose) *is the specified usages for which the data objects are collected.*

Intended purpose specifies the property of data objects.

IP: $object(OM) \cup type(OM) \rightarrow 2^{Purpose}$ is a function that returns intended purposes of a data object or type.

Definition 10 (Access Purpose) *is intentions for accessing data objects.*

Access purpose specifies the property of data accesses.

Authorised Access Purpose AAP: $CR(RM) \rightarrow 2^{Purpose}$ is a function that returns authorised access purposes.

Definition 11 (Purpose Model)

$$\begin{aligned} PurposeModel :: purpose &: 2^{Purpose} \\ IP &: object(OM) \cup type(OM) \rightarrow 2^{Purpose} \\ AAP &: CR(RM) \rightarrow 2^{purpose} \end{aligned}$$

The purpose model *PM* can be represented as a tuple ⟨*purpose, IP, AAP*⟩.

3.4 Requests, Transactions, and Accesses

This section specifies the entities for accessing data objects, namely, requests, transactions and accesses.

Definition 12 (Request)

$$\begin{aligned} Request :: obj &: object(OM) \\ ap &: purpose(PM) \end{aligned}$$

The request is denoted as a 2-tuple $\langle obj, ap \rangle$. We use $Req : 2^{Request}$ to denote the set of requests in a system.

Current Request CReq: $CR(RM) \rightarrow Req$ is a function that returns the request currently presented.

Definition 13 (Transactions) *denoted as Transaction, are the executions or procedures to perform a request.*

To ensure an object is accessed in a controlled manner, only specified transactions may be allowed. For example, the diagnosing request consists of three transactions: reading treatment history, analysing medical test results, and appending new diagnosis to the treatment history.

Current Transaction CT: $CR(RM) \rightarrow Transaction$ is a function that returns the transaction currently performed.

Authorised Transactions AT: $Req \rightarrow 2^{Transaction}$ is a function returns the authorised transactions for a request.

Model entities related to object accesses are access modes, necessary access, and current access.

Definition 14 (Access Modes) *are the modes of accesses performed on data objects.*

Let *Mode* denote the set of access modes. $Mode = \{create, read, write, append, delete\}$

Definition 15 (Necessary Accesses) *are the accesses that are needed to achieve an access purpose.*

$$\begin{aligned} NecAcc :: ap &: Purpose \\ tp &: type(OM) \\ trans &: Transaction \\ x &: Mode \end{aligned}$$

NA: 2^{NecAcc} denotes the set of necessary accesses.

Definition 16 (Current Accesses) *are accesses that a conditional role is performing.*

$$\begin{aligned} CurAcc :: cr &: CR(RM) \\ obj &: object(OM) \\ x &: Mode \end{aligned}$$

CA: 2^{CurAcc} denotes the set of current accesses.

Definition 17 (Access Model)

$$\begin{aligned} AccessModel :: Req &: 2^{Request} \\ CReq &: CR(RM) \rightarrow Req \\ Trans &: 2^{Transation} \\ CT &: CR(RM) \rightarrow Trans \\ AT &: Req \rightarrow 2^{Trans} \\ Mode & \\ NA &: 2^{NecAcc} \\ CA &: 2^{CurAcc} \end{aligned}$$

The access model *AM* can be represented as a tuple ⟨*Req, CReq, Trans, CT, AT, Mode, NA, CA*⟩.

Having defined the entities in our purpose-based access model, the system state can be defined.

3.5 The State of A System

A system state consists of the state variables corresponding to the components defined in previous sections: OM, RM, PM, AM.

state PPS of
$OM : ObjectDataModel$
$RM : RoleModel$
$PM : PurposeModel$
$AM : AccessModel$
inv ...
init ...
end

The initialisation condition on the state is defined as:
init $\sigma \triangleq \{\sigma.object(OM) = \{\} \wedge \sigma.type(OM) = \{\} \wedge$
$\sigma.purpose(PM) = \{\} \wedge \sigma.AAP(PM) = \{\mapsto\} \wedge$
$\sigma.Req(AM) = \{\} \wedge \sigma.Trans(AM) = \{\} \wedge$
$\sigma.CA(AM) = \{\} \wedge \sigma.NA(AM) = \{\}\}$

Having defined the entities in a purpose-based access control model and system state, we are now in the position to specify privacy requirements.

4 Privacy Invariants in A Purpose-Based Access Control Model

A privacy policy has been given in [9]:

A subject may only have access to personal data if this access is necessary to perform its current task, and only if the subject is authorised to perform this task. The subject may only access data in a controlled manner by performing a transformation procedure, for which the subject's current task is authorised. In addition, the purpose of its current task must correspond to the purposes for which the data was obtained or consent must be given by the data subjects.

Using the entities specified in Section 3, and according to the process of data access, privacy requirements can be expressed in following invariants (we place "′" behind a state variable to refer to the variable in the new system state):

(a) Data Collection Invariants

(a1) A data object can be created if and only if it is necessary for the conditional role to fulfill its current request.

Given two successive system states v, v',
$v = (OM, RM, PM, AM)$,
$v' = (OM', RM', PM', AM')$,
(v, v') satisfies privacy invariant-(a1), iff
$\forall\, cr \in CR(RM),\ type_j \in type(OM), ap \in purpose(PM)$:
$obj \notin object(OM) \wedge \langle obj, ap\rangle = CReq(AM)(cr)$
$\wedge \langle ap, type_j, CT(AM)(cr), create\rangle \notin NA(AM)$
$\Longrightarrow obj \notin object(OM') \vee TypeOf(OM')(obj) \neq type_j$

(a2) A data object may be created if and only if the purpose of a conditional role's current request match the purpose of the object's type.

(v, v') satisfies privacy invariant-(a2), iff
$obj \notin object(OM) \wedge \langle obj, ap\rangle = CReq(AM)(cr) \wedge$
$ap \notin IP(PM)(type_j) \Longrightarrow obj \notin object(OM') \vee$
$TypeOf(OM')(obj) \neq type_j$

(b) Role Authorisation Invariants

These invariants specify the authorisation of conditional role, access purpose and transaction.

(b1) A user's current conditional role has to be authorised.

For a system state $v = (OM, RM, PM, AM)$,
v satisfies privacy invariant-(b1), iff
$\forall\, u \in user(RM),\ \langle r, cond\rangle \in CR(RM)$:
$\langle r, cond\rangle = CCR(RM)(u) \Longrightarrow r \in AR(RM)(u) \wedge$
$cond \Leftrightarrow \mathsf{true}$

(b2) A conditional role's access purpose in its current request has to be authorised for the conditional role.

v satisfies privacy invariant-(b2), iff
$\langle obj, ap\rangle = CReq(AM)(cr) \Longrightarrow ap \in AAP(PM)(cr)$

(b3) A conditional role's current transaction has to be authorised for the conditional role's current request.

v satisfies privacy invariant-(b3), iff
$trans = CT(AM)(cr) \Longrightarrow$
$trans \in AT(AM)(CReq(AM)(cr))$

(c) Data Access Constraints

(c1) A conditional role may only have current access to a data object if the access of executing the transaction on the object type is the necessary access for the access purpose.

v satisfies privacy invariant-(c1), iff
$\langle obj, ap\rangle = CReq(AM)(cr) \wedge \langle cr, obj, x\rangle \in CA(AM) \Rightarrow$
$\langle ap, Typeof(OM)(obj), CT(AM)(cr), x\rangle \in NA(AM)$

This invariant specifies the necessity of data access.

(c2) A conditional role may only have current access to a data object, if the purpose of its current request is compliant to the intended purposes of the type of the object.

v satisfies privacy invariant-(c2), iff
$\langle obj, ap\rangle = CReq(AM)(cr) \wedge \langle cr, obj, x\rangle \in CA(AM) \Rightarrow$
$ap \in IP(PM)(Typeof(OM)(obj))$

This specifies purpose compliance of data access.

(c3) A conditional role may delete a data object, if and only if it is necessary for its current request.

(v, v') satisfies privacy invariant-(c3), iff
$obj \in object(OM) \wedge \langle obj, ap\rangle = CReq(AM)(cr) \wedge$
$\langle ap, Typeof(OM)(obj), CT(AM)(cr), delete\rangle \notin$
$NA(AM) \Longrightarrow obj \in Object(OM')$

This specifies the necessity of data object deletion.

(c4) A conditional role may delete a data object, if and only if the purpose of its current request is compliant to the intended purpose of the type of the object.

(v, v') satisfies privacy invariant-(c4), iff
$obj \in object(OM) \wedge \langle obj, ap\rangle = CReq(AM)(cr) \wedge ap \notin$
$IP(PM)(Typeof(OM)(obj)) \Longrightarrow obj \in object(OM')$

This specifies purpose compliance of object deletion.

The invariant of the system state PPS is the conjunction of these expressions, denoted as $inv\text{-}PPS$.

The invariants have been specified in this section. Next we specify the model rules, and give the proof obligations.

5 Model Rules

In this section, an example of formal specifications of model rules are given. The precondition and the postcondition are used to specify the rules. Proof obligations [5] of operations show that the operations are satisfiable.

Rule: create-object

Conditional role *cr* requests to create an object *obj* with the type *tp*. This is specified as following:

$create\text{-}object(cr : CR(RM), obj, tp : type(OM))$
ext rd $RM : RoleModel$ rd $PM : PurposeModel$
rd $AM : AccessModel$ wr $OM : ObjectDataModel$
pre $obj \notin object(OM) \wedge \langle obj, ap\rangle = CReq(AM)(cr) \wedge$
$ap \in IP(PM)(tp) \wedge$
$\langle ap, tp, CT(AM)(cr), create\rangle \in NA(AM)$
post $OM' = \langle object(OM) \cup \{obj\}, type(OM),$
$typeAttribute(OM), attributeValue(OM),$
$TypeOf(OM) \cup \{obj \mapsto tp\}, AttributeOf(OM),$
$ValueOf(OM)\rangle$

The precondition says that obj is not already in the set of objects, and to create object obj in current request is necessary access. The postcondition says that obj in included in the new object data model.

Next, defined symbols representing the operation's precondition and postcondition are introduced.

$pre\text{-}create\text{-}object(cr, obj, tp, OM, RM, PM, AM) \stackrel{\text{def}}{=}$
$obj \notin object(OM) \wedge \langle obj, ap\rangle = CReq(AM)(cr) \wedge ap \in IP(PM)(tp) \wedge \langle ap, tp, CT(AM)(cr), create\rangle \in NA(AM)$
$post\text{-}create\text{-}object(obj, tp, OM, OM') \stackrel{\text{def}}{=}$
$OM' = \langle object(OM) \cup \{obj\}, type(OM),$
$typeAttribute(OM), attributeValue(OM),$
$TypeOf(OM) \cup \{obj \mapsto tp\}, AttributeOf(OM),$
$ValueOf(OM)\rangle$

The following is satisfiability obligation associated with this operation.

Proof Obligation: Operation $create\text{-}object$ is satisfiable.

$create\text{-}object\text{-}sat$
$OM : ObjectDataModel;\ RM : RoleModel;$
$PM : PurposeModel;\ AM : AccessModel;\ inv\text{-}PPS;$
$pre\text{-}create\text{-}object(cr, obj, tp, OM, RM, PM, AM)$

$\exists\, obj, type : type(OM), OM : ObjectDataModel,$
$OM' : ObjectDataType \cdot$
$post\text{-}create\text{-}object(obj, tp, OM, OM') \wedge inv\text{-}PPS'$

Next we give proof for this satisfiability obligation.

from $OM : ObjectDataModel; RM : RoleModel;$
$PM : PurposeModel; AM : AccessModel; inv\text{-}PPS;$
$pre\text{-}create\text{-}object(cr, obj, tp, OM, RM, PM, AM)$
1 $\{obj\} : 2^{object(OM')}$
2 $object(OM) \cup \{obj\} : 2^{object(OM')}$
3 $\{obj \mapsto tp\}$: $object(OM') \rightarrow type(OM')$
4 $TypeOf(OM) \cup \{obj \mapsto tp\}$: $object(OM') \rightarrow type(OM')$
5 from $inv\text{-}PPS$
5.1 $ap \in IP(PM)(tp)$
5.2 $\langle ap, tp, CT(AM)(cr), create\rangle \in NA(AM)$
infer $object(OM') = object(OM) \cup \{obj\} \wedge$
$TypeOf(OM') = TypeOf(OM) \cup \{obj \mapsto tp\}$
6 $\exists\, obj, tp : type(OM), OM : ObjectDataModel,$
$OM' : ObjectDataModel \cdot inv\text{-}PPS'$
infer $\exists obj, type : type(OM),$
$OM : ObjectDataModel, OM' : ObjectDataType \cdot$
$post\text{-}create\text{-}object(obj, tp, OM, OM') \wedge inv\text{-}PPS'$

6 Conclusions

Privacy requirements should be formally specified and enforced in order to establish consistency between privacy preservation promises and practices. In this paper, based on purpose-based access control approach, entities of the model were formally specified. With this it is possible to present invariants corresponding to privacy requirements. We included purpose into our model, which makes it clear to specify privacy requirements. In order to complete the access control model, specifications of operations and their proof obligations have also been investigated. We have shown how this can be achieved using VDM.

Acknowledgment. This research is supported by China Scholarship Council and Department for Education and Skills under Scholarship for Excellence. The authors would like to thank the reviewers for their insightful comments.

References

[1] M. Ackerman, L. F. Cranor, and J. Reagle. Privacy in E-Commerce: Examining user scenarios and privacy preferences. In *ACM Conference on Electronic Commerce*, 1999.

[2] A. Acquisti. Privacy in electronic commerce and the economics of immediate gratification. In *CECOMM*, 2004.

[3] R. J. Anderson. Privacy technology lessons from healthcare. In *IEEE Symposium on Security and Privacy*, 2000.

[4] D. E. Bell and L. LaPadula. Secure computer systems: Unified exposition and multics interpretation. *MITRE technical report, MITRE Corporation*, 2997:ref A023 588, 1976.

[5] I. C. Bicarregui, I. S. Fitzgerald, P. A. Lindsay, R. Moore, and B. Ritchie. *Proof in VDM: A Practitioner's Guide*. Springer, London, 1994.

[6] J. Byun, E. Bertino, and N. Li. Purpose based access control for privacy protection in relational database systems. Technical Report 2004-52, Purdue University, 2004.

[7] J. Byun, E. Bertino, and N. Li. Purpose based access control of complex data for privacy protection. In *Proceedings of the tenth ACM symposium on Access control models and technologies*, pages 102–110, 2005.

[8] Federal Trade Commission (FTC). Privacy online: A report to congress, June 1998.

[9] S. Fischer-Hübner and A. Ott. From a formal privacy model to its implementation. In *Proceedings of 21st NIST-NCSC National Information Systems Security Conference*, pages 512–524, 1998.

[10] Q. He. Privacy enforcement with an extended role-based access control model. Technical Report TR-2003-09, North Carolina State University, Feb. 28 2003.

[11] S. Jajodia, P. Samarati, M. L. Sapino, and V. S. Subrahmanian. Flexible support for multiple access control policies. *ACM Transaction on Database System*, 26(2):214–260, 2001.

[12] C. B. Jones. *Systematic Software Development using VDM (second edition)*. Prentice Hall, 1990.

[13] G. Karjoth and M. Schunter. A privacy policy model for enterprises. In *CSFW*, pages 271–281, 2002.

[14] M. Marchiori. The platform for privacy preferences 1.0 (P3P1.0) specification. Technical report, W3C, Apr. 2002.

[15] H. Oberholzer. A privacy protection model to support personal privacy in relational databases. Technical report, Rand afrikanns university, May 2001.

[16] Organisation of Economic Co-operation and Development (OECD). OECD guidelines on the protection of privacy and transborder flows of personal data, Oct. 26 2004.