Supporting User Privacy Preferences on Information Release in Open Scenarios

C.A. Ardagna,[1] S. De Capitani di Vimercati,[1] S. Foresti,[1] S. Paraboschi,[2] P. Samarati[1]

[1]DTI - Università degli Studi di Milano - 26013 Crema, Italy
firstname.lastname@unimi.it
[2]DIIMM - Università degli Studi di Bergamo - 24044 Dalmine, Italy
parabosc@unibg.it

Abstract

Access control solutions for open systems are typically based on the assumption that a client may adopt approached specifically designed for the server to protect the disclosure of her sensitive information. These solutions however do not consider the specific privacy requirements characterizing the client. In this paper, we put forward the idea of adopting a different model at the client-side, aimed at minimizing the amount of sensitive information released to a server. The model should be based on a formal modeling of the client portfolio and should easily support the definition of privacy preferences and disclosure limitations for empowering the user in the release of her personal information.

1 Introduction

The advancements in the Information Technology allow people to use and access resources and services on the Web anywhere and anytime. Web-based service systems may then receive requests from different, remotely located, parties that may be unknown. This aspect makes the traditional solutions for regulating access to resources and services no more applicable since they are typically based on a preliminary identification and authentication of the requesting client. The server providing a service or a resource has instead the need of identifying and communicating the conditions under which access can be granted, and a client has the need of proving whether she is entitled to get the access. In this context, credentials can be adopted as a convenient way for regulating interactions among different unknown parties in open systems. A server can then give the client a list with all the possible sets of credentials/properties that would enable the service, and the client can choose those credentials/properties that satisfy the server request and at the same time is more convenient with respect to her privacy.

Many existing solutions (e.g., [4, 5, 11, 12, 16]) have proposed novel policy languages for enforcing credential-based access control at the server side. These solutions typically assume that a symmetric approach can be adopted at the client-side, where policies are used to regulate the release of the private information (credentials) of the client. However, since these approaches are specifically target for the server, they do not fully support the client's requirements in term of privacy and data management. An attempt to consider the client's privacy requirements is related to the specification of *secondary use* restrictions, that is, how the personal information can be managed and possibly distributed after the client has released it to an external server. The Platform for Privacy Preferences (P3P) [13] is a well-know XML-based language that addresses the secondary use aspect by allowing clients to assess whether the privacy practices adopted by a server provider comply with her privacy preferences. P3P includes a language and a mechanism for ensuring that clients can be informed about the privacy policies of the server. The corresponding language that would allow clients to specify their preferences as a set of preference-rules is called A P3P Preference Exchange Language (APPEL) [14]. A client however can neither specify the sensitivity of her personal information nor select the "minimal pertinent information" that should be released to the server to gain the access. Although recent proposals (e.g., [8, 10, 15]) have addressed these problems, they result still preliminary since they provide a simplified and partial modeling of the client's personal information and lack of the flexibility to model and manage client's privacy requirements.

In this paper, we address the problem of developing a powerful, flexible, and easy to use solution for specifying privacy preferences to offer clients a more meaningful control over information released to a server. In particular, we aim at developing a fine-grained solution that allows clients to specify their privacy preferences on each single

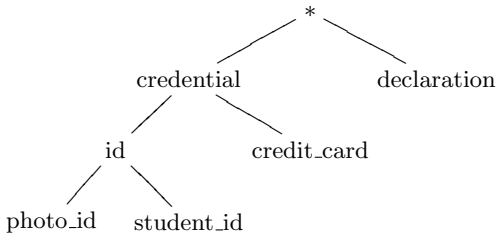

Figure 1: An example of hierarchy of credential types

property and/or credential belonging to them and to take advantage from modern credential technologies (e.g., [1, 9]) to release to the server exactly the information needed to have access and nothing more, in full respect of clients' privacy. This problem is without doubt complex and needs to be considered from different, complementary, and synergistic point of views: usability, social, economic, and technological perspectives. In particular, the development of user-friendly, intuitive user interfaces that enforce the privacy preferences plays an important role for the successful application of the proposed solution. In the following, we focus on the technological aspect and put forward the idea of developing a fine-grained model for the specification of privacy preferences [2, 3].

2 Modeling of the client portfolio

A key requirement for a framework empowering a client with full control over personal information released during the interactions with a server is the definition of an expressive model representing such a personal information. The model should enable the client to easily state her preferences on the disclosure of credentials/properties and should take advantage from emerging credential technologies that support the selective release of properties within credentials, as allowed by novel anonymous credentials (e.g., [1, 9]). In the following, we use the term *portfolio* to refer to the set of certified properties (*credentials*) and uncertified properties (*declarations*) characterizing a client [5].

2.1 Abstractions over credential types

Each credential in the client portfolio is characterized by a unique identifier, a set of certified properties, an issuer, and a type. The type of a credential identifies the properties that the credential certifies. The declarations form a special credential of type "declaration" whose issuer is the client itself. Since specifying privacy preferences on each single property and/or credential may be cumbersome for a client, credential types can be organized in a *hierarchy*, whose leaves represent the types of credentials in the client portfolio, and whose intermediate nodes represent abstractions over them. In this way, a client can specify her privacy preferences in terms of credential abstractions, types, and instances. Privacy preferences specified on abstractions and types are propagated down in the hierarchy following the most specific principle. The root of the hierarchy, denoted $*$, is an abstraction of any type of credential (including declarations) in the client portfolio. Formally, a hierarchy $\mathcal{H}$ of credential types is a pair $(\mathcal{T},\preceq_{isa})$, where $\mathcal{T}$ is the set of all credential types and abstractions over them, and $\preceq_{isa}$ is a partial order relationship over $\mathcal{T}$. Given two types t_i and t_j in $\mathcal{T}$, $t_i \preceq_{isa} t_j$ if t_j is an abstraction of t_i. Figure 1 illustrates a hierarchy of credential types where, for example, *id* is an abstraction of credential types *photo_id* and *student_id* (i.e., *photo_id* $\preceq_{isa}$ *id* and *student_id* $\preceq_{isa}$ *id*).

The hierarchy of credential types represent the common knowledge shared between a client and a server. Since a server has not any knowledge about the client portfolio, any request by the server for a set of (certified) properties may only refer to credential types and abstractions in the hierarchy.

2.2 Atomic vs non-atomic credentials

The need for avoiding the disclosure of unnecessary personal information is an important requirement for a client to preserve her privacy. Unfortunately, the most common kind of credentials used today in distributed systems (e.g., X.509 certificates) can only be released as a whole or not disclosed at all. This implies that the release of such credentials, which we call *atomic*, entails the disclosure of all the properties they certify, thus making impossible the disclosure of the minimal information needed for service access. Our model goes one step forward and takes into consideration modern credential technologies (e.g., U-Prove, and Idemix [6, 7]) supporting the selective release

PROPERTIES

Id	Dep.	Value
SSN		123-45-6789
Name		Bob
Degree		Master
Country		USA
HomeNumber		789-231-044
WorkNumber		789-543-021
MobileNumber		779-523-134
NickName		bob75
VISANum	✓	4353...21
MCNum	✓	5643...18

CREDENTIALS

Id	Atom.	Type	Properties
Passport	✓	*photo_id*	Name,Country
myUnivId		*student_id*	Name,Degree
myVISA	✓	*credit_card*	Name,VISANum
myMC	✓	*credit_card*	Name,MCNum
decl		*declaration*	SSN,Name,Degree,Country,HomeNumber WorkNumber,MobileNumber,NickName,VISANum,MCNum

Figure 2: An example of portfolio

of arbitrary subsets of properties within credentials. We refer to these credentials as *non-atomic*, since each client can individually release properties within credentials, or the existence of the credential itself. The unique restriction imposed by non-atomic credentials is that the release of a property within a credential also entails the release of the existence of the credential certifying it. Clearly, declarations are non-atomic credentials, since they can be individually released by the client when needed.

2.3 Credential-dependent vs credential-independent properties

Properties in the client portfolio can be associated uniquely with the client or can depend on the specific credential certifying them. For instance, name is a property that can be certified by different credentials but its value depends only on the owner of these credentials. A property representing a credit card number can instead be specific of the credential certifying it. In the following, we use the terms *credential-independent* and *credential-dependent* to refer to properties whose values depend only on the credential owner and on the specific credential certifying them, respectively. The importance of distinguish these two classes of properties arise when a client has to define her privacy preferences. As a matter of fact, since credential-independent properties correspond to the same piece of information, a client has to specify her privacy preference for such properties only once.

Example 2.1 *Figure 2 illustrates an example of a portfolio, including properties* SSN, Name, Degree, Country, HomeNumber, WorkNumber, MobileNumber, *and* NickName, *which are credential-independent, and* VISANum *and* MCNum, *which are credential-dependent. The portfolio contains credentials* Passport (*of type* photo_id), myUnivId (*of type* student_id), myVISA *and* myMC (*both of type* credit_card), *and a declaration* decl *including all the properties in the portfolio. The non-atomic credentials are* myUnivId *and* decl.

3 Privacy preference requirements and specifications

When accessing a service/resource, a client may need to release personal information according to a server request, stating the information about which conditions need to be satisfied for the access to be granted. A client has then to determine which properties/credentials disclose to gain access while also limiting the release of sensitive information that is not strictly necessary. For instance, suppose that for accessing a specific service, a server requires the address or the email of the requesting client. In this case, the client may prefer to release her address instead of her email since she consider the email more sensitive than the address. Clearly, if there are several alternative options from which the client can choose, the task of determining which properties/credentials release may become complex. A simple, while effective and flexible, solution to express privacy preferences on the release of properties/credentials is then needed to: *i)* automatically regulate the disclosure of sensitive information upon server requests; and *ii)* preserve the privacy of the client by providing a support for determining the minimal information that has to be released to acquire a service.

An essential step for defining a solution for the specification of privacy preferences consists in identifying the main requirements that it should satisfy. We now briefly discuss such requirements in the following.

- *Fine-grained specification.* The framework for privacy preference specifications should allow clients to define their privacy preferences on each single property and credential in the portfolio, thus providing a fine-grained control on their personal information. A privacy preference should reflect how much a client values the disclosure of a given personal information. For instance, a client should have the possibility to specify that her email is more valuable than her address, meaning that if a server requests the email of the address for granting the access to a service, the address is selected and released.

- *Support for sensitive associations.* There are cases where two or more elements in the portfolio should be considered more (or less) sensitive when their values are associated than when either appears separately. For instance, the association between user `Name` and her `Degree` can be considered more sensitive than the properties singularly taken or the association between `HomeNumber` and `Country` carries less information than the information carried by these two properties since from the area code reported in `HomeNumber` it is possible to infer the country. A client may then need to specify how she values the association among different pieces of information.

- *Support for constraints on disclosure.* There are also cases where the associations of the portfolio elements should be forbidden since a client never wants to release some information in association or where some limitations should be taken into consideration when disclosing personal information. For instance, a client may want to protect the association between her `Name` and her `NickName`, meaning that the association between these two properties should never be released, or may want to release only one among her telephone numbers.

A possible solution for expressing the requirements above consists in using *sensitivity labels* that reflect how much a client values the release of the credentials/properties in the portfolio. Clearly, our solution to be effective needs to be also supported by user-friendly interfaces that allow a client to easily express her preferences.

3.1 Fine-grained specifications

Each credential and property in the portfolio is associated with a sensitivity label λ that belongs to an arbitrary set Λ characterized by a partial order relationship $\succeq$ and a composition operator $\oplus$. The partial order relationship allows a client to declare whether a property/credential should be considered more or less sensitive than another property/credential. Intuitively, if an element is considered more sensitive than another element, the sensitivity label of the first will dominate the sensitivity label of the second. The sensitivity label associated with a property then reflects how much the user considers the property sensitive. For instance, if property `SSN` is considered more sensitive than property `Name`, $\lambda(\texttt{SSN}) \succ \lambda(\texttt{Name})$. The sensitivity label associated with a credential reflects the sensitivity of the *existence* of the credential, independently from the properties it certifies. For instance, the release of a dialysis certificate including properties `Name` and `Country`, not only discloses demographical information, but also the fact that the client has a specific health problem. The sensitivity associated with the disclosure of an atomic credential corresponds therefore to the composition of the sensitivity labels of all properties certified by the credential and the sensitivity label of the existence of the credential itself. For non-atomic credentials, their disclosure may be evaluated by considering only the existence if this is the unique information released about them.

Note that our general definition of sensitivity label permits to use different approaches to define such labels. For instance, sensitivity labels could be classical multilevel security classifications (e.g., Top Secret, Secret, Confidential, Unclassified, possibly with associated categories) with the $\oplus$ operator corresponding to the *least upper bound*. Also, they could be positive integer values, where the $\oplus$ operator can be either the *sum* (i.e., $\lambda_i \oplus \lambda_j = \lambda_i + \lambda_j$) and therefore reflecting an *additive property*, the *maximum* (i.e., $\lambda_i \oplus \lambda_j = max(\lambda_i, \lambda_j)$), or the *minimum* (i.e., $\lambda_i \oplus \lambda_j = min(\lambda_i, \lambda_j)$).

3.2 Associations and constraints

The sensitivity label for the combined release of a set of properties and/or credentials is naturally obtained by composing (operator $\oplus$) their sensitivity labels. The cases where the release of some elements of the portfolio do not simply correspond to the composition of their sensitivity labels (it could be either higher or lower) can be easily modeled allowing a client to express how much she values the association. The sensitivity label of the association must then be considered when composing the sensitivity labels of the involved elements. We also aim at easily supporting constraints on the disclosure of properties and credentials. In particular, the following types of associations and constraints may need to be considered.

PROPERTIES

λ(SSN) : 20	λ(WorkNumber) : 8
λ(Name) : 1	λ(MobileNumber) : 12
λ(Degree) : 5	λ(NickName) : 1
λ(Country) : 2	λ(VISANum) : 10
λ(HomeNumber): 4	λ(MCNum) : 15

CREDENTIALS

$\lambda(Passport)$: 4
$\lambda(myUnivId)$: 6
$\lambda(myVISA)$: 3
$\lambda(myMC)$: 8
$\lambda(decl)$: 0

SENSITIVE VIEWS

λ({Name, Degree}) : 5

DEPENDENCIES

λ({HomeNumber, Country}) : -2

FORBIDDEN VIEWS

{Name,NickName}

DISCLOSURE LIMITATIONS

{HomeNumber,WorkNumber,MobileNumber}$_1$

Figure 3: Portfolio sensitivity specifications

- *Sensitive views* ($\lambda > 0$) model situations where the combined release of a set of properties and/or credentials carries *more* information than the composition of the sensitivity labels of its elements. For instance, sensitivity label $\lambda(\{\texttt{Name}, \texttt{Degree}\})$ represents the *additional* sensitivity due to the association between properties Name and Degree.
- *Dependencies* ($\lambda < 0$) model situations where the combined release of a set of properties and/or credentials carries *less* information than the composition of the sensitivity labels of its elements. For instance, sensitivity label $\lambda(\{\texttt{HomeNumber}, \texttt{Country}\})$ represents the sensitivity label to be *removed* due to the fact that the area code in the phone number carries information about the country, meaning that the combined release of Country together with HomeNumber does not release additional information with respect to the release of HomeNumber only. Note that, in general, the sensitivity of an association can take any value, provided that disclosing several elements in association overall has a sensitivity which is at least the one of the most sensitive element.
- *Forbidden views* model situations where some portfolio elements cannot be jointly released; any proper subset of a forbidden view can instead be disclosed. For instance, properties Name and NickName have each a sensitivity label that has to be considered when each property is released, but their association should never be disclosed.
- *Disclosure limitations* model situations where a client may need to enforce a limitation of the kind *at most n of these elements* can be jointly released. For instance, a user may specify that at most one among HomeNumber, WorkNumber, or MobileNumber, can be released.[1]

Example 3.1 *Figure 3 illustrates an example of sensitivity labels for properties and credentials in the portfolio illustrated in Figure 2. For simplicity, in the example, we use integer numeric values as sensitivity labels, composed through the sum operator. The figure also reports one sensitive view* {Name, Degree}*; one dependency* {HomeNumber, Country}*; one forbidden view* {Name,NickName}*; and one disclosure limitation* {HomeNumber,WorkNumber,MobileNumber}$_1$*, where the subscript* 1 *states that at most* 1 *among the properties composing the constraint can be released.*

Note that a client may also need to specify different preferences for different servers. For instance, a client may be willing to release her credit card information to her bank to perform a payment, but she may not want to release her dialysis certificate. By contrast, she may be interested to disclose to a hospital her dialysis certificate to reserve an exam, but she may not want to release her credit card information. If the client needs to express different privacy requirements for different servers, she may assign multiple labels to each component in her portfolio, one for each server.

The privacy preferences expressed through sensitivity labels associated with properties, credentials, and associations along with the disclosure constraints support the automatic selection of the information to be released to a server. Intuitively, given a server request expressed in terms of properties and/or credentials that a requesting client

[1] Note that forbidden view constraints can be seen as a specific kind of disclosure limitation constraints, specifying that at most $n-1$ among n portfolio elements can be released.

can release, the client determines set of properties/credentials whose release satisfies the service request and, at the same time, the corresponding sensitivity is minimized. The sensitivity of a release is obtained by composing all the sensitivity labels of the disclosed elements, of the exposed sensitive views, and of the released dependencies. In addition, the release to be considered acceptable from the client must not violate any disclosure constraint.

4 Conclusions

We illustrated the main privacy requirements of a client, interacting with a server, to acquire a service online. The definition of sensitivity labels for the properties, credentials, and associations, together with the specification of disclosure constraints, permit a client to express her privacy preferences in an effective and flexible way. A client can then identify the subset of her portfolio that better satisfies the server request, minimizing the amount of disclosed sensitive information.

References

[1] A. Anderson and H. Lockhart. *SAML 2.0 profile of XACML*. OASIS, September 2004.

[2] C.A. Ardagna, S. De Capitani di Vimercati, S. Foresti, S. Paraboschi, and P. Samarati. Minimizing disclosure of private information in credential-based interactions: A graph-based approach. In *Proc. of PASSAT 2010*, Minneapolis, Minnesota, USA, August 2010.

[3] C.A. Ardagna, S. De Capitani di Vimercati, S. Foresti, S. Paraboschi, and P. Samarati. Supporting privacy preferences in credential-based interactions. In *Proc. of WPES 2010*, Chicago, Illinois, USA, October 2010.

[4] C.A. Ardagna, S. De Capitani di Vimercati, S. Paraboschi, E. Pedrini, P. Samarati, and M. Verdicchio. Expressive and deployable access control in open Web service applications. *IEEE TSC*, 2010. (to appear).

[5] P. Bonatti and P. Samarati. A uniform framework for regulating service access and information release on the Web. *Journal of Computer Security (JCS)*, 10(3):241–272, 2002.

[6] S. Brands. Rethinking public key infrastructure and digital certificates – building in privacy. *MIT Press*, 2000.

[7] J. Camenisch and A. Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *Proc. of EUROCRYPT 2001*, Innsbruck, Austria, May 2001.

[8] W. Chen, L. Clarke, J. Kurose, and D. Towsley. Optimizing cost-sensitive trust-negotiation protocols. In *Proc. of INFOCOM 2005*, Miami, FL, USA, March 2005.

[9] D. Hardt, J. Bufu, and J. Hoyt. OpenID attribute exchange 1.0, December 2007. `http://openid.net/developers/specs/`.

[10] P. Kärger, D. Olmedilla, and W.-T. Balke. Exploiting preferences for minimal credential disclosure in policy-driven trust negotiations. In *Proc. of SDM 2008*, Auckland, New Zealand, August 2008.

[11] A. J. Lee, M. Winslett, J. Basney, and V. Welch. The Traust authorization service. *ACM Transactions on Information and System Security (TISSEC)*, 11(1):1–33, February 2008.

[12] T. Ryutov, L. Zhou, C. Neuman, T. Leithead, and K. E. Seamons. Adaptive trust negotiation and access control. In *Proc. of SACMAT 2005*, Stockholm, Sweden, June 2005.

[13] W3C. *Platform for privacy preferences (P3P) project*, April 2002.

[14] World Wide Web Consortium. *A P3P Preference Exchange Language 1.0 (APPEL1.0)*, April 2002.

[15] D. Yao, K.B. Frikken, M.J. Atallah, and R. Tamassia. Private information: To reveal or not to reveal. *ACM Transactions on Information and System Security (TISSEC)*, 12(1):1–27, October 2008.

[16] T. Yu, M. Winslett, and K.E. Seamons. Supporting structured credentials and sensitive policies through interoperable strategies for automated trust. *ACM Transactions on Information and System Security (TISSEC)*, 6(1):1–42, February 2003.