Mask: A System for Privacy-Preserving Policy-Based Access to Published Content *

Mohamed Nabeel, Ning Shang, John Zage, Elisa Bertino
Purdue University, West Lafayette, Indiana, USA
{nabeel, nshang, jzage, bertino}@cs.purdue.edu

ABSTRACT

We propose to demonstrate *Mask*, the first system addressing the seemingly-unsolvable problem of how to selectively share contents among a group of users based on access control policies expressed as conditions against the identity attributes of these users while at the same time assuring the privacy of these identity attributes from the content publisher. *Mask* consists of three entities: a *Content Publisher*, Users referred to as *Subscribers*, and *Identity Providers* that issue certified identity attributes. The content publisher specifies access control policies against identity attributes of subscribers indicating which conditions the identity attributes of a subscriber must verify in order for this subscriber to access a document or a subdocument. The main novelty of Mask is that, even though the publisher is able to match the identity attributes of the subscribers against its own access control policies, the publisher does not learn the values of the identity attributes of the subscribers; the privacy of the authorized subscribers is thus preserved. Based on the specified access control policies, documents are divided into subdocuments and the subdocuments having different access control policies are encrypted with different keys. Subscribers derive the keys corresponding to the subdocuments they are authorized to access. Key distribution in Mask is supported by a novel group key management protocol by which subscribers can reconstruct the decryption keys from the subscription information they receive from the publisher. The publisher however does not learn which decryption keys each subscriber is able to reconstruct. In this demonstration, we show our system using a healthcare scenario.

Categories and Subject Descriptors

I.7.4 [**Document and Text Processing**]: Electronic Publishing; H.2.4 [**Database Management**]: Systems

*A detailed technical treatment of the techniques behind our system is available in [3, 4].

SIGMOD'10, June 6–11, 2010, Indianapolis, Indiana, USA.

General Terms

Design, Management, Security

Keywords

Access Control, Privacy, Identity, Broadcast Systems, Group Key Management

1. INTRODUCTION

The tremendous growth in electronic media has made the publication and sharing of information in either open or closed environments easy and effective. However, selective access to sensitive information should be enforced in order to comply with legal requirements, organizational policies, subscription conditions, and so forth. Access control policies and mechanisms for their automatic enforcement must thus be in place. Modern languages for access control, like XACML, support, among other features, the specification of access control policies in terms of properties of users. Examples of such properties, referred to as *identity attributes*, include age, country of origin and role in an organization. A user satisfies a given access control policy, if his/her identity attributes satisfy all the conditions of the policy. A crucial issue in this context is represented by the fact that often identity attributes record sensitive information about users that need to be protected even from the party publishing the content. Protection is crucial to assure users' privacy not only from outside attackers but also from malicious or careless insiders [2]. We thus need systems able to enforce access control policies based on identity attributes while at the same time assuring the privacy of these attributes. Mask, the system we present in this demo, is the first access control system that addresses such requirements.

Mask consists of the following three entities.

1. *Content Publisher* (Pub) that publishes content on which selective access control is enforced.
2. Users referred to as *Subscribers* (Sub) consume published content based on their credentials.
3. *Identity Providers* (IdP) that issue certified identity attributes.

Mask supports the following two strong privacy and security requirements.

1. Fine-grained selective attribute-based access control for content.

2. The privacy-preserving matching of the users' identity attributes against the access control policies and content consumption. In Mask the content publisher is not able to learn which access control conditions are verified by which users and yet it is certain that users access only the content portions for which they are authorized. Further, the content publisher does not learn which content portions users can access.

In Mask, different content portions are encrypted with different symmetric keys based on the access control policies. Users can derive only the keys associated with the content portions they are authorized to access. In Mask also the key distribution is privacy-preserving in that the content publisher does not learn which users receive which keys. The approach of encrypting different subdocuments with different keys for selective dissemination is not new. Existing approaches however are neither efficient nor privacy preserving. By contrast, Mask supports a novel privacy-preserving key management scheme which is efficient in that join/leave (rekey) operations for some users do not affect other users. Under such a scheme, users derive the keys from some initial secret information and some variable public data. Further, our system eliminates the need for private communication channels between each user and the content publisher for rekey operation.

We will demonstrate Mask by using an electronic health record (EHR) [5, 1] dissemination system in a hospital. Typical hospital stakeholders include employees playing different roles such as receptionist, cashier, doctor, nurse, pharmacist, system administrator and non-employees such as patients. A cashier, for example, need not have access to data in EHRs except the billing information in them, while a doctor or a nurse need not have access to billing information. The typical identity attributes used by the stakeholders in our EHR system, such as role, location and position, can be used as good contextual information to connect with other publicly available information in order to learn sensitive information about individuals, leading to privacy violations. There have been many incidents where hospital employees steal coworkers' identity attributes to impersonate them and carry out highly damaging insider attacks [2]. Our system is a promising step forward to minimize such identity theft incidents.

The rest of the paper is organized as follows. Section 2 presents an overview of Mask and its components. Section 3 describes the demo.

2. TECHNICAL DETAILS

Figure 1 shows a high-level overview of the interactions in an EHR dissemination system adopting Mask. As mentioned in the Introduction, Mask consists of three entities: a *Content Publisher*, *Subscribers*, and *Identity Providers* denoted by Pub, Subs and IdPs, respectively.

The main interactions are:

1. Sub shows proofs of identity attributes to IdP.

2. If IdP is convinced that identity attributes belong to Sub [1], IdP issues a *identity token* ($\mathcal{IT}$) for each such identity attribute. We assume that every IdP issues these tokens in a uniform format $\mathcal{IT} = (\mathsf{nym}, \mathsf{id\text{-}tag}, c, \sigma)$ where nym is a pseudonym for uniquely identifying a Sub in the system, id-tag is the tag of the identity attribute under consideration, c a Pedersen commitment[2] for the identity attribute value, and σ is the IdP's digital signature for nym, id-tag and c.

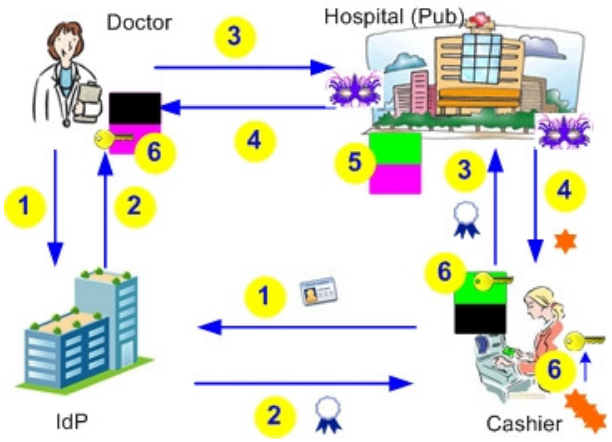

Figure 1: *Mask* system overview

3. Subs register with the Pub in order to access the published content, encoded as documents. During registration, Subs first retrieve the Pub's access control policies, each of which is of the form (*subject*, policy-id), and the *policy configurations*, each of which is a disjunction of some access control policies. The subject of each access control policy, identified by the policy-id, is a conjunction of *identity attribute conditions*. An identity attribute condition has the form $\mathsf{cond} = (\mathsf{id\text{-}tag}\ \ \mathsf{op}\ \ l)$, where op is a comparison operator such as $=, <, >, \leq, \geq, \neq$, and l is a value that can be assumed by the identity attribute referred to by the tag id-tag. A policy is satisfied if and only if all the identity attribute conditions in that policy are satisfied. An example policy with two identity attribute conditions is ("level $\geq$ 58" $\wedge$ "role = nurse"). A policy configuration specifies, for each document, which access control policies are associated with which subdocuments of the document. In order to access a subdocument, Sub should be able to satisfy at least one policy in the policy configuration of the subdocument. After determining which policy configurations are to be satisfied, Sub presents its required identity tokens through OCBE protocols [3]. Note that during this process Pub learns neither the attributes of Sub nor the satisfiability of the corresponding condition. Another important issue is how much privacy is sufficient to prevent inferences based on the conditions to which Subs register. If a Sub registers only for those conditions associated with the subdocuments it wants to access, Pub can infer the values of certain identity

[1]How an identity provider is convinced is outside the scope of this paper. We have developed multi-factor identity verification techniques as part of another project, and we assume that these techniques are used here.

[2]A Pedersen commitment is a type of cryptographic commitment which allows a user to commit to a value while keeping it hidden and preserving the user's ability to reveal the committed value later. See [3, 4] for details.

attributes of the Sub. Therefore, for the maximum privacy, Subs need to register for all conditions having the same id-tag. For example, "role = nurse" and "role = doctor" have the same id-tag role. Thus a Sub will register for both these conditions, even if it cannot satisfy both. We quantify the privacy requirement using the notion of *(m,n)-Privacy* where n is the total number of conditions in the system having the same id-tag and m is the number of conditions that Sub registers. As m approaches n, the level of privacy for Sub increases but also the overhead for Pub increases. For the above example, $(2, 2)$-Privacy for the id-tag "role" provides the maximum privacy for that identity attribute.

4. For each identity token received from Sub during registration, Pub sends a *conditional subscription secret* (CSS). CSSs are used by Subs to derive the decryption keys for the subdocuments for which they satisfy the policy configuration and are managed by the novel Mask key management scheme, referred to as GKM scheme. Note that Subs can obtain the CSS from the response message if and only if they send a valid identity token.

5. Pub encrypts the subdocuments having the same policy configuration with the same symmetric key and publishes the document together with certain meta information that is used for key derivation in Step (6).

6. Subs who satisfy one or more policy configurations can derive the symmetric decryption keys for the corresponding subdocuments.

We now briefly discuss the three entities of Mask and the operations they perform.

- IdPs issue identity tokens to Subs. For each identity token issuance, an IdP performs two major computations, the calculation of Pedersen commitments and digital signatures.

- Subs engage in three phases: obtaining identity tokens, obtaining CSSs, and viewing subdocuments. For each identity token received from an IdP, Subs verify the Pedersen commitment and the digital signature. Subs obtain a CSS from Pub for each condition they want to satisfy. For privacy reasons, Subs may register for multiple conditions having the same id-tag. To view a subdocument, Subs perform an inexpensive vector inner product and a symmetric key decryption where the key is the result of the inner product.

- Pub performs all the computationally intensive operations in our system. In what follows, we provide a brief technical overview of the operations along with the four main components of Pub: Policy manager, OCBE module, GKM module and Document manager.

  The *policy manager* manages attributes, conditions, access control policies and policy configurations. It is designed as a relational database application where dependencies are enforced as key constraints. Mask provides a Graphical User Interface (GUI) to conveniently manage them.

  The *OCBE module* is responsible for the registration of Subs. Recall that during OCBE protocols, Subs receive CSSs which they use to derive decryption keys. We assume that each Sub has a unique pseudonym nym across all the systems and nym does not change. This module manages a table of CSSs delivered against each nym and each cond. Mask provides an Application Program Interface (API) for Subs to interact with Pub and obtain CSSs in a privacy-preserving manner.

  The *GKM module* implements our GKM scheme. For each subdocument, the GKM module first builds a matrix A of CSSs as described in [3] and then computes a random vector Y from the null space of the matrix A . We call such a vector Y as an *access control vector* (ACV).

  The *document manager* manages the association of access control policies with subdocuments, and the encryption of subdocuments. Once the policies are set, it automatically derives the minimum number of policy configurations required for the document. For each policy configuration, it assigns a unique symmetric key K, and encodes K and Y into a vector X, where Y is the already calculated ACV. Encrypted documents are placed in a public location where any Sub can access.

  An encrypted subdocument has the following XML format:

```
<subdoc-i>
    <payload>encrypted subdocument</payload>
    <timestamp>time</timestamp>
    <encoded-key>ACV</encoded-key>
    <randoms>Random values</randoms>
</subdoc-i>
```

  The <payload> element contains the subdocument encrypted with the symmetric key K. The <timestamp> element records the time of encryption. The <encoded-key> element contains the ACV. The <randoms> element contains the random values used to generate the ACV. Subs use these random values to compute the KEV and derive the key K.

  The encrypted EHR documents are named by the patients' unique pseudonym patient-id. For example, EHR_patient-99.xml is the EHR of the patient with the patient-id patient-99. The subdocuments of a document are encrypted incrementally as and when they get updated. For the first time, Subs get full EHR documents, but for the subsequent requests, based on the timestamps Subs need to get only the updated subdocuments. This can potentially save a lot of bandwidth if the same records are viewed frequently.

3. THE DEMONSTRATION

Mask will be demonstrated by using the EHR dissemination scenario. We have implemented Mask in C/C++ and Java 1.6 on a GNU/Linux kernel version 2.6.28 and MySQL5.0. We have used synthetic data to populate the system and demonstrate its capabilities. During the demo, we will show, but not limited to, the following functions of

Mask:

Obtaining identity tokens: We will show how Subs obtain identity tokens from IdPs.

Managing policies: We will show how to create policies from scratch starting from a repository of identity attribute definitions, and how to update/delete existing policies using our GUIs while maintaining consistency. For example, pop-up add/remove conditions window in Figure 2 shows that the access control policy *acp4* currently has two conditions "role = nur" and "level $\geq$ 59".

Managing documents: We will show how policy configurations are automatically generated and associated with subdocuments in a document based on the access control policies specified for the document. The document manager GUI allows to edit and view these associations. For example, the highlighted lines in Figure 3 show that the policy configuration *PC4* consists of the policies *acp3* and *acp4*, and it is applicable to <PysicalyExam> and <Plan> subdoucments.

Once the documents are subdocumented, we will also show how to generate ACV's for each subdocument and encrypt with symmetric keys.

Obtaining CSSs: We will show how Subs obtain CSSs using their identity tokens in a privacy preserving manner. Specifically, the steps of the OCBE protocol will be demonstrated.

Accessing documents: We will show how Subs access new as well as updated encrypted documents based on the timestamps. Subs can download updates at the granularity of subdocuments.

Also with the help of GUI and command line tools, we will show the internal working of selected components of Mask.

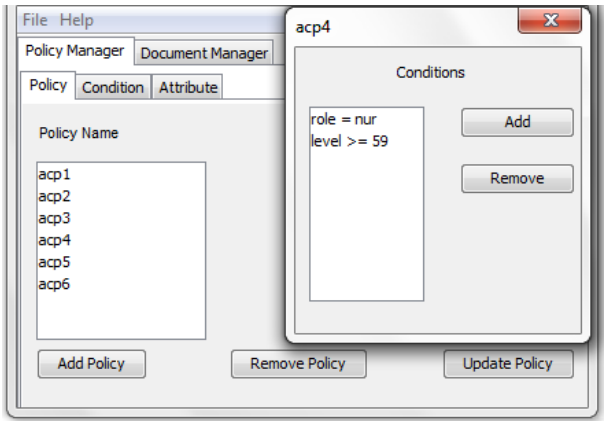

Figure 2: Policy update example in *Mask*

4. CONCLUSION

The problem of disseminating contents to user groups by enforcing identity attribute-based access control policies while at the same time assuring the privacy of the user identity attributes has not been addressed before. We demonstrate Mask, a system implementing an approach to such problem, in the context of a scenario for EHR sharing.

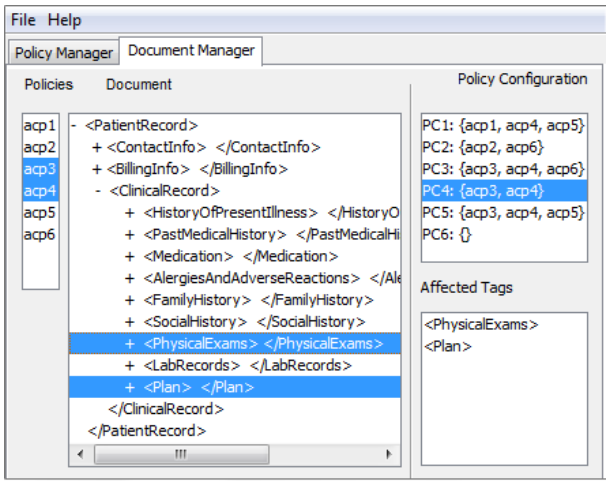

Figure 3: Document policy update example in *Mask*

Acknowledgments

The work reported in this paper has been partially supported by the NSF grant 0712846 "IPS: Security Services for Healthcare Applications," and the MURI award FA9550-08-1-0265 from the Air Force Office of Scientific Research. We would also like to thank the anonymous reviewers for their valuable comments.

5. REFERENCES

[1] M. Eichelberg, T. Aden, J. Riesmeier, A. Dogac, and G. B. Laleci. A survey and analysis of Electronic Healthcare Record standards. *ACM Comput. Surv.*, 37(4):277–315, 2005.

[2] R. Richardson. CSI Computer Crime and Security Survey. Technical report, Computer Security Institute, 2008.

[3] N. Shang, M. Nabeel, F. Paci, and E. Bertino. A privacy-preserving approach to policy-based content dissemination. Technical Report CERIAS TR 2009-27, Purdue University, 2009.

[4] N. Shang, M. Nabeel, F. Paci, and E. Bertino. A privacy-preserving approach to policy-based content dissemination. In *ICDE '10: Proceedings of the 2010 IEEE 26th International Conference on Data Engineering*, 2010.

[5] XML in clinical research and healthcare industries. http://xml.coverpages.org/healthcare.html.