Activity-based Access Control Model to Hospital Information

Le Xuan Hung[1], Sungyoung Lee[1], Young-Koo Lee[1], and Heejo Lee[2]
[1]*Dept. of Computer Engineering, Kyung Hee University, Korea,*
[2]*Dept. of Computer Science and Engineering, Korea University, Korea*
{lxhung, sylee}@oslab.khu.ac.kr, yklee@khu.ac.kr, heejo@korea.ac.kr

Abstract

Hospital work is characterized by the need to manage multiple activities simultaneously, constant local mobility, frequently interruptions, and intense collaboration and communication. Hospital employees must handle a large amount of data that is often tied to specific work activities. This calls for a proper access control model. In this paper, we propose a novel approach, Activity-based access Control Model (ACM). Unlike conventional approaches which exploit user identity/role information, ACM leverages user's activities to determine the access permissions for that user. In ACM, a user is assigned to perform a number of actions if s/he poses a set of satisfactory attributes. Access permissions to hospital information are granted according to user's actions. By doing this, ACM contributes a number of advantages over conventional models: (1) facilitates user's work; (2) reduces complexity and cost of access management. Though the design of ACM first aims to support clinical works in hospitals, it can be applied in other activity-centered environments.

1. Introduction

Hospital work is characterized by the need to manage multiple activities simultaneously, constant local mobility, frequently interruptions, and intense collaboration and communication [1][2]. These conditions impose important demand on users that need to frequently switch between tasks, contributing to a decrease in efficiency and becoming a source of errors and mishaps. Users must constantly log in and out of devices, starting and stopping sets of applications, looking for what types of information needed for their care and requesting to access that information repeatedly. The design of a proper access control model to increase efficiency of user's work and decrease complexity and cost of policy management is requisite.

Traditional access control models exploit user identity/role information to determine the set of access permissions [6]-[11][15]-[17]. The policy specification of these models tightly couple identity/role of users with their permissions. This coupling requires security administrators to envisage all types of missions that a user could carry out in the organization so that they can grant proper access privileges to that user. It practically increases complexity and cost to manage access policy. Some works solely adopt context to limit the applicability of the available permissions [13][14]. Other works use context as a foundation to authorize access [12]. However, their concept of context is so general, for example location context, time context, system context, etc. It does not specify any mission of a user. As consequent, these approaches fail to work in activity-centered environments like hospitals.

In this paper, we propose a new approach, *Activity-based access Control Model* (ACM), to solve the problem. ACM is a part of our current TBSI (*Trust-based Security Infrastructure for ubiquitous computing*) project [3] incorporating with CAMUS (*Context-Aware Middleware for Ubiquitous Computing Systems*) [4][5]. ACM is our on-going research work to provide an efficient access control mechanism for pervasive healthcare project at *u-Life Care Research Center* (u-LCRC), Korea. In ACM model, permissions are assigned to user's activities. This means that each user's activity achieves a number of access permissions so that the user can carry out his/her job. Each user will be granted to perform certain activities if s/he posses a set of attributes satisfied policies. By doing this, ACM contributes a number of advantages over existing approaches: facilitates user's work; reduces complexity and cost of policy management.

The rest of the paper is organized as follows. Section 2 presents ACM model. Section 3 describes policy specification language based on XML. ACM framework is shown in Section 4. Section 5 briefly mentions related works. Finally, section 6 concludes the paper and outlines our future work.

2. Activity-based Access Control Model

13th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications(RTCSA 2007)
0-7695-2975-5/07 $25.00 © 2007 IEEE

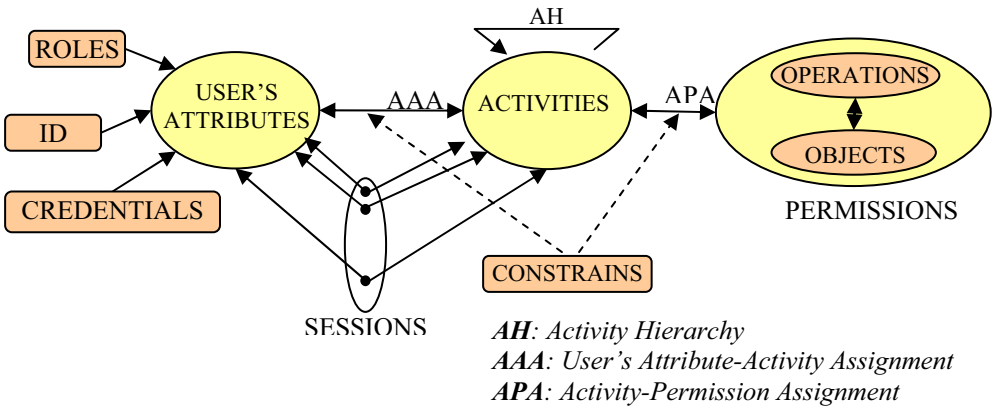

Figure 1. Activity-based Access Control Model

Core ACM recognizes five administrative elements: (1) *users (i.e. user's attributes)*, (2) *activities* and (3) *permissions*, where *permissions* are composed of (4) *objects* and (5) *operations*. Figure 1 shows the relationships between these elements. *User* is a human-being interacting with a computing system. *Activity* is a human activity. It differs from the term '*task'* in a workflow system in the sense that is does not model nor control real-word human activities. An activity can be created and modified according to the desire of the user. *Object* is a medical data resource as well as system resource. It can be thought of as a container that contains information. An operation is an executable image of a program, for example 'read', 'write', 'execute'. *Permission* is an authorization to perform certain operations within the system. *Constraint*, similar to the concept of constraint from RBAC model, is defined as a predicate which applied to a relation between two ACM elements returns a value of '*acceptable'* or '*not-acceptable*'; 'separation of duty' (*SoD*) is an example of constraints.

Now, we define above basic sets as follows:

- $\boldsymbol{C}$: set of credentials containing user's attributes (user's role, user's identification are considered user's attributes). $2^{\boldsymbol{C}}$ is used to present the set of $\boldsymbol{C}$. We define $\boldsymbol{C_{sub}}$ as a sub-set of $\boldsymbol{C}$.
- $\boldsymbol{A}$: set of all activities. $2^{\boldsymbol{A}}$ is used to present the set of $\boldsymbol{A}$.
- $\boldsymbol{OBJ}$: set of all objects
- $\boldsymbol{OPR}$: set of all operations (i.e. access modes) to objects
- $\boldsymbol{P}$: set of all permissions. $2^{\boldsymbol{OPR\times OBJ}}$ is used to present the set of $\boldsymbol{P}$.

Originating from the basic sets, we derive other sets which associate these sets as follows:

- $\boldsymbol{AAA} \subseteq \boldsymbol{C}\times\boldsymbol{A}$, a many-to-many mapping between a set of user's attributes and activities (attributes-activity assignment).
- $\boldsymbol{APA} \subseteq \boldsymbol{A}\times\boldsymbol{P}$, a many-to-many mapping between activities and permissions (activity-permission assignment).
- The function *assigned_activities* is defined as $(\boldsymbol{C_{sub}} \subseteq \boldsymbol{C}) \rightarrow 2^{\boldsymbol{M}}$, the mapping of sub-set of $\boldsymbol{C}$ onto a set of activities. Formally:

$assigned\_activity(\boldsymbol{C_{sub}})=\{a\in\boldsymbol{A}|(\boldsymbol{C_{sub}},a)\in\boldsymbol{AAA}\}$.

- The function *assigned_permissions* is defined as $(a{:}\boldsymbol{A}) \rightarrow 2^{\boldsymbol{P}}$ (i.e. $2^{\boldsymbol{OPR\times OBJ}}$), the mapping of activity a onto a set of permissions. Formally:

$assigned\_permission(a)=\{p\in\boldsymbol{P}|(a,p)\in\boldsymbol{APA}\}$.

Now we define *Object Access Authorization* (OAA) as a user with set of credentials $\boldsymbol{C_{sub}}$ can perform an operation *opr* on object *obj* only if there exits an activity $a \in \boldsymbol{A}$ such that the permission authorizes the performance of *opr* on *obj*. Formally:

$$\boldsymbol{OAA}: \boldsymbol{C}\times\boldsymbol{OPR}\times\boldsymbol{OBJ}\rightarrow\{true, false\}$$

$$\mathbf{OAA}(\boldsymbol{C_{sub}}, opr, obj) = \begin{cases} true \text{ ;if user with } \boldsymbol{C_{sub}} \text{ can perform} \\ \qquad \text{operation } opr \text{ on object } obj \\ false \text{ ;otherwise} \end{cases}$$

$$\boldsymbol{OAA}(\boldsymbol{C_{sub}}, opr, obj) \Rightarrow$$

$$\exists\,(C_{sub}\subseteq\boldsymbol{C}, a\in\boldsymbol{A}, p\in\boldsymbol{P}),$$
$$a\in assigned\_activities(\boldsymbol{C_{sub}}) \wedge$$
$$p\in assigned\_permission(a) \wedge (opr, obj)\in\boldsymbol{P}$$

In our proposed model, the constraints are similar to constraint in RBAC model. The constraints are diverse and administrated according to access control policies of each

system. We therefore leave the constraint investigation for design of specific applications.

3. ACM Policy Specification Language

We are developing the policy specification for ACM based on XML (XACM). XACM contains: (1) *user's attributes* element, (2) *activity* element, (3) *permission* elements, (4) *AAA* elements, and (5) *APA* elements. Figure 2 shows the simplified XML schema that defines XACM policy.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified">
<xs:element name="XACMPolicy" type="XACM_policy_type"/>
< xs:element name="attr" type="attr_type"/>
<xs:element name="activity" type="activity_type"/>
<xs:element name="permission" type="permission_type"/>
<xs:element name="AAA" type="AAA_type"/>
<xs:element name="APA" type="APA_type"/>
<xs:complexType name="attr_type">
      <xs:attribute name="attr_id" type="xs:string" use="required"/>
      <xs:attribute name="description" type="xs:string" use="optional"/>
      ...
</xs:complexType>
<xs:complexType name="activity_type">
      <xs:attribute name="activity_id" type="xs:string" use="required"/>
      <xs:attribute name="description" type="xs:string" use="optional"/>
      ...
</xs:complexType>
<xs:complexType name="permission_type">
      <xs:attribute name="permission_id" type="xs:string" use="required"/>
      <xs:attribute name="object" type="xs:string" use="required"/>
      <xs:attribute name="operation" type="xs:string" use="required"/>
      ...
</xs:complexType>
<xs:complexType name="AAA_type">
      <xs:sequence>
      <xs:element name="attr_id" type="xs:string" maxOccurs="unbounded"/>
      <xs:element name="activity_id" type="xs:string"
maxOccurs="unbounded"/>
      <xs:sequence>
      ...
</xs:complexType>
<xs:complexType name="APA_type">
      <xs:sequence>
      <xs:element name="activity_id" maxOccurs="unbounded"/>
      <xs:element name="permission_id" maxOccurs="unbounded"/>
      </xs:sequence>
      ...
</xs:complexType>
<xs:complexType name="XACM_policy_type">
      <xs:sequence>
      <xs:element ref="attr" maxOccurs="unbounded"/>
      <xs:element ref="activity" maxOccurs="unbounded"/>
      <xs:element ref="permission" maxOccurs="unbounded"/>
      <xs:element ref="AAA" maxOccurs="unbounded"/>
      <xs:element ref="APA" maxOccurs="unbounded"/>
      </xs:sequence>
      </xs:complexType>
</xs:schema>
```

Figure 2. Simplified XML-based Policy Specification for ACM Model (XACM)

4. ACM Framework Design

ACM framework is designed and implemented to fulfill objectives of ACM model. Its design is shown in Figure 3. ACM framework is composed of Access Enforcement, Access Decision Making modules wherein Access Enforcement includes Credential Authentication and Access Trigger. Prior to access request, user's credentials containing user's attributes are securely transmitted to Credential Authentication for verification (process 1).

In the design, we incorporate *Activity Recognition* (AR) module from our CAMUS middleware [4][5]. It provides user's activity information to ACM by gathering raw contextual data related to user activity, producing high level context, and then reasoning the user actions. As illustrated in Figure 3, *Activity Monitor* frequently monitors user activities and sends raw data to *CAMUS's AR Agent* (process 2). AR Agent deduces user activities and transmits to ACM (process 3). Access Enforcement maps user's activities to set of corresponding permissions. Then, user's credentials, user's activities, and permissions are sent to Access Decision Making (process 4). If accepted, Access Trigger automatically queries information from hospital information database (process 5).

5. Related Work

Access Control technology has a long history. Started in late 60s, Lampson [6] introduced a formal, mathematical description of access control based on an *access matrix*. In 1973, Bell and Lapadula [7] proposed the first *rule-based access control* model. Then in 1983, access control technology took a significant step forward when the U.S. Department of Defense (DoD) published its *Trusted Computer System Evaluation Criteria* (TCSEC) [8]. In the standard, two important access control modes were defined: *discretionary access control* (DAC) and *mandatory access control* (MAC). However, one of the big limitations of these models is that they manage privileges based on individuals instead of group of individuals. This brings high complexity and significant cost to manage in growing large-scale systems. Therefore, during early 90s, a new access control model based on *role* of user (*Role-based Access Control* - RBAC) was introduced by D. Ferraiolo [9] to tackle this problem. RBAC is conceptually simple: access to computer system objects is based on a user's role in an organization. The authorizations are not assigned directly to particular users, but to roles. A role denotes a job function describing the authority and responsibility conferred on a users assigned to that role. RBAC is a basis for many descendant models afterwards.

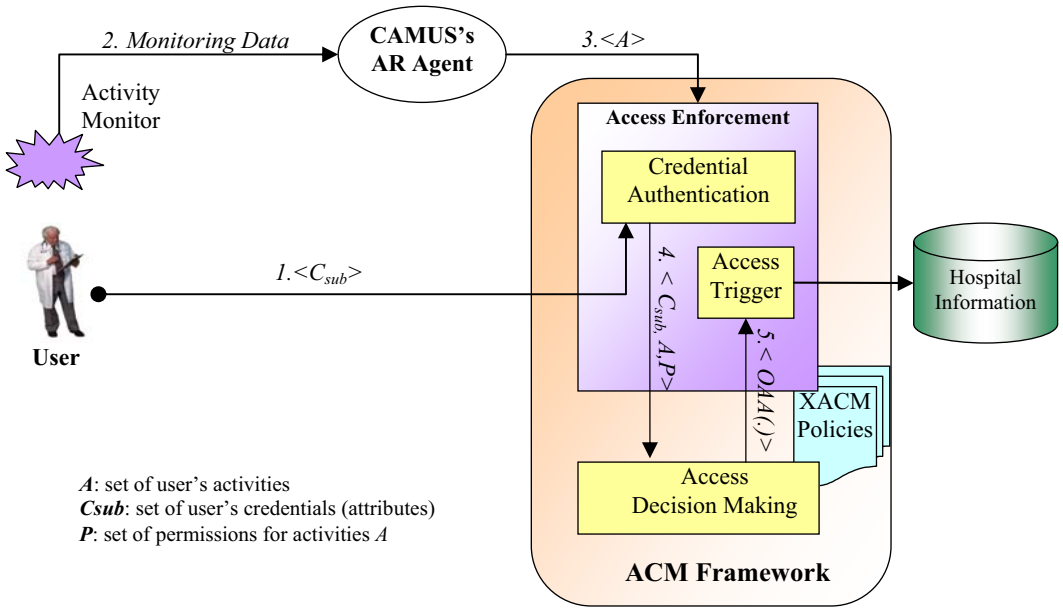

Figure 3. ACM Framework Design

Convington *et al* [10] proposes a *Generalized Role-Based Access Control* model (GRBAC). GRBAC is an extension of the traditional RBAC model for securing application in the highly connected home as well as in other environments. The major benefit of this model is its combination of usability and expressiveness. It solves the problems in RBAC approach by introducing three different kinds of role: *subject roles* (e.g. 'adult', 'child'), *object roles* (e.g. image, source code, streaming video), and *environment roles* (e.g. daytime, nighttime). The *Usage Control* ($UCON_{ABC}$) model [11] encompasses traditional access control, trust management, and digital rights management (DRM) to control the access to and usage of digital information objects. $UCON_{ABC}$ enables finer-grained control over usage of digital objects than that of traditional access control policies and models. On the other hand, it covers both centrally controllable environment and an environment where central control authority is not available. $UCON_{ABC}$ also deals with privacy issues in both commercial and noncommercial environments.

A. Corradi *et al* proposes a new model of context-based access control, UbiCOSM (*Ubiquitous Computing Context-based Security Middleware*) [12]. UbiCOSM uses the context as a foundation for security policy specification and enforcement processes. Unlike traditional access control models, permissions are directly associated with contexts, instead of user identities/roles. UbiCOSM is similar to our approach in the sense that it avoids exploiting user/role information to determine the set of user permissions. However, it differs from ACM that its context is general (e.g. location, time, etc). This does not directly specify actual task of users.

Several models have been introduced for controlling user's access to electronic patient records [13][14]. In [13], the authors supposed that hospital workers are highly mobile; they are constantly changing location to perform their daily work, which includes visiting patients, locating resources, such as medical records, or consulting with other specialists. The information required by these specialists is highly dependent on their location. The paper describes a location-aware medical information system that was developed to provide access to resources such as patient's records or the location of a medical specialist, based on the user's location. Their major contribution is based on trained back-propagation neural-network to estimate the user's location and a client to access information from the hospital information system that is relevant to the user's current location. Gustavo et al [14] presented a contextual role based access control authorization model aiming to increase the patient privacy and the confidentiality of patient data, whereas being flexible enough to consider specific cases. The basis of this model is RBAC with extension of role-tree hierarchy with authorization inheritance; positive and negative authorizations; static and dynamic separation of

duties based on weak and strong role conflicts. Additional extension is contextual authorization which uses environmental information available at access time, like user/patient relationship, in order to decide whether a user is allow to access and EPR resource. This model provides a more flexible and precise authorization policy, where permission is granted or denied according to the right and the need of the user to carry out a particular job function.

A recent development in policy languages relating to Electronic Healthcare Records (EHR) management is Cassandra [15]. It is a role-based trust management system with an elegant and readable policy specification language based on Datalog with constraints. The main security-specific aspect of Cassandra involves its role awareness; a number of specific predicates marshal role activation and deactivation. With file special predicates, Cassandra can easily express a wide range of policies including role hierarchy, role delegation, separation of duties, cascading revocation, automatic credential discovery, and trust negotiation.

Tees Confidentiality Model (TCM) [16] supports emergency overrides under certain circumstances. It provides a specific ordering designed to manage conflicts consistent with the proposed UK Patient Confidentiality Requirements. Confidentiality permissions are processed in a defined order connected with specific types of policy from least to most powerful (in terms of override) labeled: IRC, ISO, IGC, RSC, and RGC (I, R, S, G, and C representing Identity, Role, Specific, General, and Confidentiality, respectively). The idea is that these stages cover all the useful configurations for positive and negative permissions levied over individuals, groups of individuals, roles, groups of roles, particular data records, and groups of data records.

Another system that has been applied in healthcare domain is PERMIS (PrivilEge and Role Management Infrastructure Standards validation) [17]. PERMIS is a RBAC infrastructure that uses X.509 attribute certificates (ACs) to store the users' roles. All access control decisions are driven by an authorization policy, which is stored in an X.509 attribute certificate. PERMIS uses PMI (Privilege Management Infrastructure) certificates for its policy representation. These policies certificates indicate the prerequisites needed to acquire some given privileges.

As aforementioned, most of these works exploit user identity/role information to determine the set of access permissions. Therefore, they are not efficient to apply in activity-centered environments like hospitals where access permissions are tightly coupled with the tasks, rather than user identities or roles. Other works [12]-[14] use context in authorization rules more or less. However, the scope of context is so general. ACM applies user's activity as a center of the model to solve the problem. Instead of assigning permissions to users/roles, ACM grants permissions to user's activities so that they can accomplish the task.

6. Conclusion and Future Work

In this work, we have introduced an Activity-based access Control Model (ACM) aiming to support activity-centered environments like hospitals. Center of the model is user's activity. Unlike existing access control approaches where user identity/role triggers policy assessment, ACM model leverages user's activities to determine access permissions of that user. By adopting activity-centered strategy, ACM contributes a number of advantages over existing approaches:

- *Facilitates user's accesses*: Users don't need much concern on searching information related to their activity, log in and out, and sending access request repeatedly. All relevant information is automatically provided according to their actions.
- *Reducing complexity and cost of access management*: Foreseeing all types of mission that a user/role may carry out in the organization to grant proper access permissions to that user/role is non-trivial job. Moreover, this is more complex in cases the user/role mission is frequently changed. ACM deals with this problem by specifying access permissions for each activity. *Activity-User* assignment is left open to define dynamically. Thus, it brings more comprehensive way of policy administration and better management of change.

Though the design of ACM first aims to support clinical works in hospitals, it can be applied in other activity-centered environments.

We are at the beginning stage of design and developing ACM model. A lot of work remains to be done. To begin with, we plan to make ACM design and implementation in great details. The policy specification language XACM is to be completed in the most expressive way. Next, we plan to build a concrete application scenario and deploy in real hospital environments

conjunction with TBSI security infrastructure and CAMUS context-aware middleware.

Acknowledgements

This research was supported by the MIC (Ministry of Information and Communication), Korea, Under the ITFSIP (IT Foreign Specialist Inviting Program) supervised by the IITA (Institute of Information Technology Advancement).

Corresponding Author: Young-Koo Lee.

References

[1] Jacob E. Bardram and Henrik B. Christensen. Pervasive Computing Support for Hospitals: An Overview of the Activity-Based Computing Project in Pervasive Computing Vol. 6, No. 1 January-March 2007

[2] Tentori, M., Favela, J. and Gonzalez, V. Towards the Design of Activity-aware Mobile Adaptive Applications for Hospitals. The 4th International Workshop on Ubiquitous Computing for Pervasive Healthcare Applications (UbiHealth 2006). September 17-21, 2006 USA

[3] Le Xuan Hung, Tran Van Phuong, Yonil Zhung, Pho Duc Giang, Sungyoung Lee, and Young-Koo Lee. Security in Ubiquitous Computing: Problems and Proposed Solution. 12th IEEE Embedded Real Time Computing Systems and its Applications (RTCSA 2006), Sydney , Australia, Aug 2006

[4] Middleware Infrastructure for Context-aware Ubiquitous Computing Systems. Technical Report , Ubiquitous Computing Lab, Kyung Hee University, Korea. http://uclab.khu.ac.kr/camus/go.php?cmd=tr Feb 2005.

[5] Hung Quoc Ngo, Anjum Shehzad, Kim Anh Pham Ngoc, Sungyoung Lee, Manwoo Jeon, "Research Issues in the Development of Context-aware Middleware Architectures", The 11th IEEE International Conference on Embedded and Real-time Computing Systems and Applications (RTCSA 2005), 17-19 August 05, HongKong

[6] Lampson, B. W., "Dynamic Protection Structures," AFIPS Conference Proceedings, 35, 1969, pp. 27–38

[7] Bell, D. E., and L. J. LaPadula, Secure Computer Systems: Mathematical Foundations and Model, Bedford, MA: The Mitre Corporation, 1973

[8] DoD, Trusted Computer System Evaluation Criteria (TCSEC), DoD 5200.28-STD.

[9] D. Ferraiolo, R. Sandhu, S. Gavrila, D.R. Kuhn, R. Chandramouli, Proposed NIST Standard for Role-Based Access Control, ACM Transaction on Information and System Security, Vol. 4, No. 3, August 2001, pages 224-274

[10] M. J. Covington, M. J. Moyer, and M. Ahamad, "Generalized Role-Based Access Control for Securing Future Applications," 23rd National Information Systems Security Conference, 2000.

[11] J. Park and R. Sandhu. The UCONABC usage control model. ACM Transactions on Information and System Security (TISSEC). Volume: 7 Issue: 1 p. 128 – 174. 2004

[12] A. Corradi, R. Montanari, and D. Tibaldi, "Context-based access control management in ubiquitous environments," Proc. Third IEEE International Symposium on Network Computing and Applications, (NCA'04), pp.253–260, Aug. 2004.

[13] Marcela D. R et al, Location-aware Access to Hospital information and services. IEEE Transactions on Information Technology in Biomedicine 2004.

[14] Gustavo H. M.B.M et al. A Contextual Role-Based Access Control Authorization Model for Electronic Patient Record. IEEE Transactions on Information Technology in Biomedicine 2003.

[15] Becker, M.Y, and Sewell, P. Cassandra: distributed access control policies with tunable expressiveness. Proc. Seventeenth IEEE Computer Security Foundation Workshop, Pacific Grove, USA, June 2004

[16] Longstaff, J.J, Lockkyer, M.A, and Nicholas, J. The Tees Confidentiality Model: an authorization model for identities and roles. Proc. Eighth ACM Symposium on Access Control Models and Technologies, Como, Italy, 2003

[17] Chadwick, D.W., and Otenko, A.. The PERMIS X.509 role based privilege management Infrastructure. Proc. Seventh ACM Symposium on Access Control Models and Technologies, Monterey, USA (ACM Press, 2002), pp. 135–140