Proceedings of the Third IASTED International Conference
Communications, Internet, and Information Technology
November 22-24, 2004, St. Thomas, US Virgin Islands

Privilege Algebra for Access Control in Digital Libraries

Ron G. McFadyen[+], Yangjun Chen[*], Fung-Yee Chan
Department of Applied Computer Science, University of Winnipeg
515 Portage Avenue, Winnipeg, Manitoba, Canada, R3B 2E9
{r.mcfadyen, ychen2, f.chan}@uwinnipeg.ca

[+]Supported by NSERC 105709-03 (139988) (Natural Sciences and Engineering Council of Canada)
[*]Supported by NSERC 239074-01 (242523) (Natural Sciences and Engineering Council of Canada)

Abstract

The Web has become an important source of information. XML has been proposed as a way to encode information organized in digital libraries. In some cases, access to information needs to be controlled to prevent unauthorized access or update. As the number of users of a digital library can be enormous, it has been proposed that credentials, rather than user identifiers, be used to control access. Determining the roles, for a user, is shown to be equivalent to performing a partial-match query on credentials. The roles, credentials, and privileges are modelled according to RBAC (role-based access control), and so given a user's roles, a number of privileges are determined. As privileges may be complex, and as many roles may be associated with any one user, privileges may appear to conflict. We propose in this paper a privilege algebra for evaluating privilege expressions. To simplify privilege expressions and to resolve conflicts that may arise, the privileges in a role/object matrix are algebraically combined.

Keywords: access control, credential, digital library, privilege, RBAC, XML

1. Introduction

A Digital Library (DL) is a system that provides global access to information to users, typically through the Internet. A DL is not only a library, but also an information repository serving many domains: healthcare, education, e-commerce, etc. Users connect to the DL to retreive, publish and update information in the DL. In this paper, we focus on the security of documents for a DL.

For our purposes, we consider a DL to be a collection of DTDs and XML documents [1] available to end-users irrespective of their own location, and concept hierarchies [2] used for document classification. In the healthcare area for example, the use of XML DTDs for clinical information is being investigated as part of the HL7 SGML/XML SIG initiative [3]; a preliminary draft of the HL7 Document Patient Record Architecture is reported in [4].

An interesting aspect of DLs is that the potential number of end-users is extremely large, more than would be encountered in a single organization. It is impractical to issue and maintain user identifiers for such a vast collection of users. Instead, other ways of managing and identifying users (the use of credentials) have been proposed in the literature [5]. Credentials are *digital credentials* that are the digital analogues of the paper credentials we carry in our wallets. Credentials can be used to describe the characteristics (e.g. age, gender) and qualifications (e.g. medical doctor, valid credit card holder) of users.

To manage users and their access privileges, we adopt the role-based access control (RBAC) model [6, 7]. RBAC allows us to relate DL privileges (browse, author) to roles associated with users, instead of directly assigning privileges to users. RBAC allows for flexible policies [7, 8] and is suitable for web-based applications [9, 10]. Also, RBAC is easily extended for credentials [11].

By incorporating credentials, concepts, and RBAC, access to information may be controlled both on the basis of information content and/or context, and on the basis of user identity and/or characteristics [2, 11].

Our approach differs from others in two aspects. First, a privilege algebra is introduced, which can be used to present access rights exactly and to manage conflicts that may seem to be present. Second, our security control is matrix-based, simplifying the user's view of the security mechanism.

The rest of the paper is organized as follows. Section 2 is a brief description of a DL structure. In Section 3, we describe our role-based control model, including the privilege algebra and the credential specification. Then, in Section 4, we discuss the security control mechanism. Finally, in Section 5, a short conclusion is set forth.

2. Digital Libraries

We consider a Digital Library (DL) to be a collection of objects that include DTDs, XML documents, and concept hierarchies as shown in Figure 1. A DTD is a *document*

type descriptor used to specify the structure of some documents. A document is a text marked up with tags that may be defined in a corresponding DTD. A Concept Hierarchy is a collection of semantic definitions that apply to knowledge domains, which can be used to classify documents. In the following, we first show XML DTDs and DTD documents in 2.1. Then, in 2.2, the concept hierarchy is discussed.

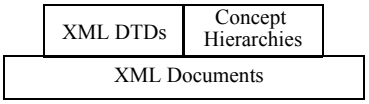

Figure 1. Main objects in a DL

2.1 XML DTDs and DTD Instances

In this subsection, we briefly show what is a DTD and what are DTD instances. Especially, how to embed security control in a DTD specification is discussed.

2.1.1. DTDs

A DTD is a document type descriptor used to specify the components and structure of XML documents. For a simple illustration, see a possible DTD for *Patient Care* documents shown in Figure 2, which shows the general

```
<!DOCTYPE Patient_Care [
<!ELEMENT Patient_Care
  (header, body?, radiology_report?)>
<!ELEMENT header
  (doc, event, patient, Doctor) >
<!ELEMENT doc>
<!ATTLIST doc
              id     ID       #REQUIRED
             date   CDATA #REQUIRED>
<!ELEMENT event (PCDATA)>
<!ELEMENT patient>
<!ATTLIST patient
              pid    ID             #REQUIRED
              bdate CDATA       #REQUIRED
              name CDATA       #REQUIRED
<!ELEMENT body
  (findings?) >
<!ELEMENT Doctor body findings radiology_report
               (PCDATA)>
]>
```

Figure 2. A DTD

structure of Patient Care records. If there are any instances of Patient Care records, they will follow the general structure outlined in this DTD. DTDs then, provide information about document structures in a DL. For instance, we know from the DTD that there is header component, a body component, and a radiology component. The header and body components have subcomponents, and so on. Obviously, such a DTD can be represented as a graph as shown in Figure 3.

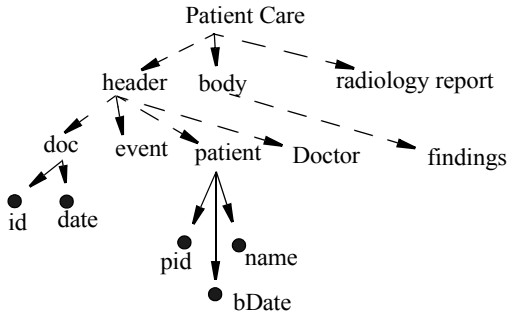

Figure 3. A DTD's graphical representation

For the purpose of security, each DTD may have its own security requirements. For instance, the above DTD can be annotated to specify that only Doctors and Nurses are allowed to update a Patient Care Record while other health care professionals are only allowed to browse its contents. In addition, each node in the graph can have its own access privileges, and these may differ from those specified elsewhere in the graph. If a node does not have access privileges explicitly specified for it, then it inherits the privileges of its parent node. We will discuss security requirements further in section 3.

2.1.2. DTD Instances

Figures 4 and 5 give an actual document and its graphical representation, which happens to correspond to the foregoing DTD.

```
<Patient_Care>
<header>
   <doc id="10" date="Oct. 2, 1999" />
     <event>
          Sept. 30, 1999
     </event>
   <patient>
     id="62144" name="Rob" bDate="Jan. 2, 1970"
   </patient>
   <Doctor>
      Dr. Jim Smith
   </Doctor>
</header>
<body>
    <findings>
       RLL nodule suggesting malignancy
    </findings>
</body>
</Patient_Care>
```

Figure 4. A document

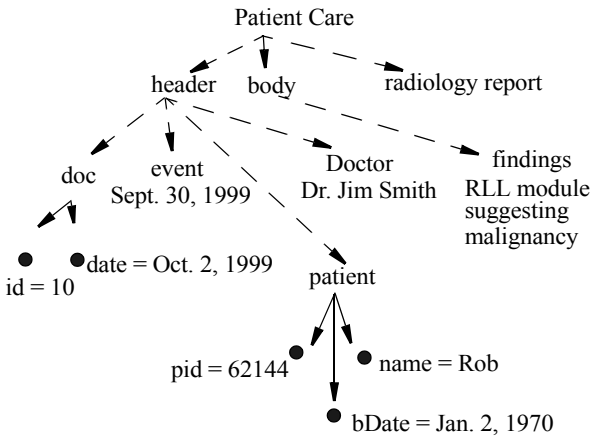

Figure 5. A document's graphical representation

In the document, a concrete document inherits the access privileges specified in the corresponding DTD. If deemed necessary, or if a concrete document does not have a corresponding DTD, a document may specify its own security and override specifications in the DTD.

2.2 Concept Hierarchies

A Concept Hierarchy, or ontology [12], is a collection of semantic definitions that apply to a domain. These may be manually assigned, or system generated [13, 14]. Each document may be related to several concepts that are entries in one or more concept hierarchies. Logically, we consider that a document (or a portion of a document) or a DTD can be related to various concepts by listing those concepts for the DL object in question, and/or listing the DL objects that pertain to a particular concept. For instance, some Patient Care records might be related to the concept of Released Patient. Logically, we can view the ontologies as a layer on top of our XML documents.

Based on ontological concepts, we could restrict a class of users (say researchers) to view those records for patients that have been treated and released. Similarly, we could restrict some users from examining whole classifications of documents. When a privilege is related to a concept, then that privilege will propagate to any documents recognized as having that concept.

We are not concerned with the internal structure of an ontology, rather we are only concerned with the idea that ontologies comprise concepts and these concepts can be related to documents in a many to many association. That is, there is an association between ontology concepts and DL objects. This can be represented as a pair of the form <c,o> where *c* is a concept and *o* is a DL object.

3. Role-based Access Control

In this section, we discuss our access control model. First, we extend the traditional role-based access model to a credential-role-based model in 3.1. Then, the concept of roles and role hierarchies are discussed in 3.2. Next, we define a privilege algebra in 3.3, which can be used to express compound privileges asigned to users. Finally, 3.4 is devoted to credential specification.

3.1 Role-Based Access Control Model

In our system, a Role-based Access Control (RBAC) Model is introduced for maintaining and implementing security. It involves the concepts of Users, Roles, and Privileges. Each user has a unique user identifier and is assigned to various roles; the association between Users and Roles is many-to-many. That is, a user identifier can be assigned to different roles and a same role can be shared by multiple users.

With RBAC, individual users are not granted privileges directly; instead, privileges are granted to roles. This provides a layer of indirection. Privileges may be positive or negative. Positive privileges represent allowable operations, such as updating a document, browsing a document (or even part of a document), but negative privileges explicitly disallow operations and are called negative privileges. For instance, the privilege administrator may allow (or grant) browse on the whole DTD such as the Patient Care Record, but disallow (or revoke) the browse privilege on some subset, say the findings subtree, of the DTD tree.

Privileges are assigned to roles in a many-to-many fashion. In addition, roles may be organized into a hierarchy, indicating that one role is more specific than another. Therefore, the relationships among users, roles and privileges can be represented using an UML class diagram as follows.

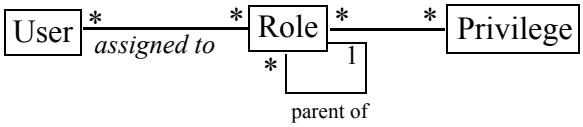

Figure 6. UML diagram: Users, Roles, and Privileges

The above method works well in a local environment. However, it cannot be used for a Web-based application since users may be unknown. For this reason, the concept of credential is utilized, which may (or may not) provide for the identity of users. Each credential is associated with one or more roles and then each user who is able to provide the credentials can be assigned the corresponding

roles. Thus, our UML class diagram is changed as shown in Figure 7.

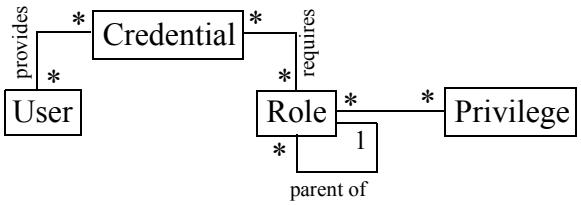

Figure 7. UML diagram including Credentials

3.2. Roles and role hierarchy

Typically roles are structured hierarchically similar to the way organizations structure themselves. We expect roles to correlate with the job functions that are assigned to employees of an organization. This correlation is represented using a reflexive association *parent of* for Roles in the UML diagram shown in Figure 7.

The following diagram illustrates potential roles for a health care organization. As we descend the hierarchy, the roles become more and more specialized. For instance, a Person is classified as either a Visitor, a Patient, or an Employee. An Employee is specialized as either an Administrator, Doctor, Nurse or Technician. In general, it is possible for a user to have more than one role, but along any one path, a user will assume just one role. Normally a user will be assigned to a role at the leaf level of the hierarchy.

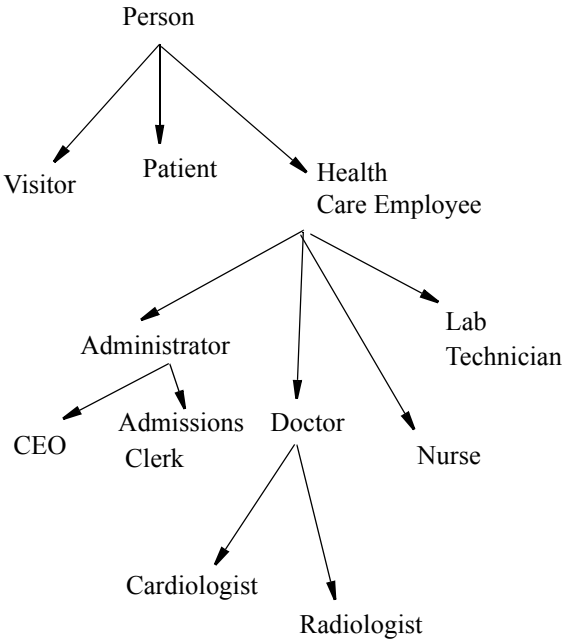

Figure 8. Hierarchical organization of roles.

As discussed previously, our users are unknown but our users provide credentials that may (or may not) provide identity. Most importantly, their credentials give us information we can use to assign users to roles.

3.3 Privileges and the Privilege Algebra

Privileges are operations that users can perform on DL objects. For our purposes, we consider two operations: browse (b) and update (u). The update privilege permits a user to alter the portion of the document to which the privilege applies. The browse privilege permits a user to view the portion of the document to which the privilege applies. Note that if the browse privilege applies to an IDREF attribute, the user can see the IDREF value, but cannot access the linked document, or document portion, if access to the document portion is denied due to privilege assignments. From the above discussion, we can see that a privilege can be represented as a pair (o, d), where $o \in$ {b, u} and d is a document (or a subtree of a document), or a set of documents. More generally, d can be a sub-structure in a DTD tree, or a concept in a concept hierarchy. If d is a sub-structure D in a DTD tree, then o can be applied to all the documents conforming to D. If d is a concept C, then o can be applied to all the documents associated with C.

In the following, we propose a privilege algebra to represent compound privilege expressions. Especially, it can be used to simplify an expression and resolve privilege conflicts.

A privilege algebra is a pair $<P, Q>$, where P is a set of all the possible privileges involved in the system and Q = {+, -}. Both '+' and '-' are binary operations defined as follows.

Definition 1. Let $p_1 = (o_1, d_1)$ and $p_2 = (o_2, d_2)$ be two privileges, where d_1 and d_2 are two single documents. The sum $p = p_1 + p_2$ is defined as follows.

(i) if $o_1 = o_2$, $p = (o_1, d_1 \cup d_2)$,

(ii) if $o_1 \neq o_2$, $p = (o_1, d_1) \wedge (o_2, d_2)$,

where '$\wedge$' represents that two priviliges are assigned simultaneously.

Definition 2. Let $p_1 = (o_1, D_1)$ and $p_2 = (o_2, D_2)$ be two privileges, where $D_1 = \{d_1^1, d_1^2, ..., d_1^n\}$ and $D_2 = \{d_2^1, d_2^2, ..., d_2^m\}$ are two document sets. The sum of $p = p_1 + p_2$ is defined as follows.

(i) Denote $p_{ij} = (o_1, d_1^i) + (o_2, d_2^j)$,

(ii) $p = \sum_{i=1}^{n} \sum_{j=1}^{m} p_{ij}$.

This definition is just the extension of '+' operation over document sets.

Definition 3. Let $p_1 = (o_1, d_1)$ and $p_2 = (o_2, d_2)$ be two privileges. The difference $p = p_1 - p_2$ is defined as follows

(i) if o_1= b and o_2 = u, $p = (o_1, d_1)$,

(ii) otherwise, $p = (o_1, d_1 / d_2)$.

Note that understanding Definition 3 is critical to understanding how negative privileges are managed. p_2 represents a negative privilege and the expression, $p_1 - p_2$, is the situation where p_1 is granted and p_2 is disallowed. In practical terms for (ii) above, where d_1 is a document tree and d_2 is a subtree of d_1, the net effect is d_1 pruned of d_2 for the operation o_1.

Definition 4. Let $p_1 = (o_1, D_1)$ and $p_2 = (o_2, D_2)$ be two privileges, where $D_1 = \{d_1^1, d_1^2, ..., d_1^n\}$ and $D_2 = \{d_2^1, d_2^2, ..., d_2^m\}$ are two document sets. The difference $p = p_1 - p_2$ is defined as follows.

(i) Denote $p_i = (o_1, d_1^i) - \sum_{j=1}^{m} (o_2, d_2^j)$

(ii) $p = \sum_{i=1}^{n} p_i$,

where p_i is computed as follows.

(a) Let $q_1 = (o_1, d_1^i) - (o_2, d_2^1)$,

(b) $q_j = q_{j-1} - (o_2, d_2^j)$, $(1 < j \le m)$

(c) $p_j = q_m$.

This definition is just the extension of '-' operation over document sets.

Below are several examples, which are used to show how the algebraical expressions are constructed to represent compound privileges. We consider the following privileges:

p_1= update access to headers within Patient Care records = <u, Patient_Care.header>

p_2= browse access to findings within Patient Care records = <b, Patient_Care.findings>

p_3= browse access to doctors within Patient Care records = <b, Patient_Care.header.Doctor>

p_4= update access to doctors within Patient Care records = <u, Patient_Care.header.Doctor>

Example 1. $p_1 + p_2$.

$p_1 + p_2$ = <u, Patient_Care.header>

+ <b, Patient_Care.findings>

= <u, Patient_Care.header> ∧ <b, Patient_Care.findings>

Example 2. $p_1 + p_4$.

$p_1 + p_4$ = <u, Patient_Care.header>

+ <u, Patient_Care.header.Doctor>

= <u, Patient_Care.header>

Example 3. $p_1 - p_3$.

$p_1 - p_3$ = <u, Patient_Care.header / Doctor>

That is, the result is update privilege on the header subtree but pruned of the Doctor branch. The expression $p_1 - p_3$ arises in the situation where the privilege administrator assigns p_1 as a positive privilege and p_3 as a negative privilege to a role; that is the administrator intends to disallow the browse privilege on the Doctor subtree. Note that p_3 can be subtracted many times: p_1-p_3...-p_3, and the result is still the same.

Example 4. $p_1 - p_4$.

$p_1 - p_4$ = <u, Patient_Care.header / Doctor>

Again, note that the expression $p_1 - p_4$ arises in the situation where the privilege administrator assigns p1 as a positive privilege and p4 as a negative privilege to a role; that is the administrator intends to disallow update on the Doctor subtree.

Note the result in examples 3 and 4 is the same, even though the negative privileges differ. In this approach, a negative privilege will remove update access from a document component.

Concerning associativity and commutativity we have the following definition.

Definition 5. Let p_1, p_2 and p_3 be three privileges. We have

(i) $(p_1 + p_2) + p_3 = p_1 + (p_2 + p_3)$,

(ii) $(p_1 - p_2) - p_3 = p_1 - (p_2 + p_3)$, and

(iii) $(p_1 + p_2) - p_3 = (p_1 - p_3) + (p_2 - p_3)$.

The meaning of (i) and (ii) is quite intuitive. We have (iii) since we think that negative privileges should always take precedence over positive privileges for safety purpose.

3.4. Credentials

A credential serves the purpose of describing an individual; a user may have several credentials. When analysing the needs of a DL and the characteristics of potential users we will uncover a number of credentials. For instance, an employee has a name and works at a certain position within an organization, a doctor holds educational documents and professional society memberships, a person holds a valid credit card, etc. A more complete example of credentials that may be held by a Radiologist are:

- employment: {name: John Smith; birthDate: Oct 2, 1970; startDate: Sept 3, 1995; position: Radiologist; employer: HealthCo}
- medicalDegree: {grantedBy: Pacific University; yearGranted: 1990; specialy: Radiology}

Traditionally, paper forms of these credentials have existed, but digital credentials that are the digital analogues of paper credentials are appropriate when access to information is provided in a distributed fashion.

Before a user may assume a certain role, the user must provide all the credentials specified in the role/credential matrix (discussed in detail in section 4). Users can have any number of credentials and so, when their credentials are processed, many roles may be granted.

Each role has a set of credentials specified for it. These credentials are specified in the form of a credential expression [2]. For example, suppose the credentials required to assume the role of Radiologist comprise holding a Medical Doctor degree with a speciality of Radiology, then the credential expression is:

$$medDegree \wedge medDegree \bullet speciality = rad$$

Credential specifications are arbitrary logical expressions and so may be complex:

- if a and b are credential expressions, then $a \vee b$ and $a \wedge b$ are credential expressions

Example 5.

- A Cardiologist is someone with a Medical Doctor degree and a specific speciality:
  $medDegree \wedge medDegree \bullet speciality = car$
- A group consisting of only Radiologists and Cardiologists can be expressed as:
  $medDegree \wedge (medDegree \bullet speciality = rad) \vee medDegree \wedge (medDegree \bullet speciality = car)$
- A group of Doctors, irrespective of speciality, could be represented as: $medDegree$
- As in [2], credentials can be organized into a hierarchy to facilitate establishment of new credentials in a credential scheme.

4. Control Mechanisms

In this section, we discuss the security control mechanism, which is a process of three phases:

(i) determine the roles according to the credentials,
(ii) determine the privileges according to the roles,
(iii) simplify the privilege algebraic expressions.

In the following, we discuss this process in great detail.

4. 1. Credentials and Roles

As shown in the following diagram, a role hierarchy can be annotated to indicate the required credentials and can be provided as an XML-based specification [11]. Credentials are cumulative: as you progress down a path of the tree, the credentials for a descendent include the credentials of all ancestor nodes. For instance, as shown in Figure 9, for someone to be an Admissions Clerk, that person must be an employee and must have the position of adminClerk in the company.

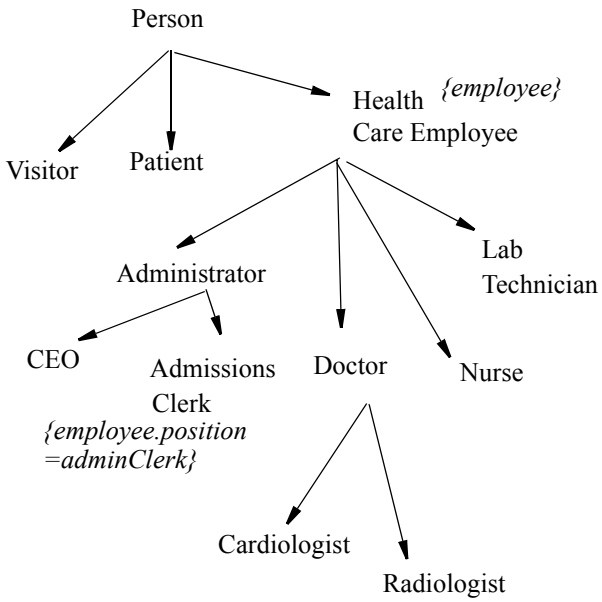

Figure 9. Roles for persons, annotated with credentials for employees and clerks.

4.2. Role/Credential Matrix

In general, the expression of the required credentials for a role, the role/credential expression, is an arbitrary boolean expression. Any such expression can be reformulated in Disjunctive Normal Form (DNF). In DNF, each role/credential expression is a disjunction of the form $c_1 \vee c_2 \vee \ldots \vee c_n$ where each disjunct, c_i, is a conjunction of terms $d_1 \wedge d_2 \wedge \ldots \wedge d_m$ where each d_i is of the form $\neg d_i$ or d_i.

This permits us to represent credentials required for a role in a simple matrix. Consider the following matrix (in Table 1) where each column corresponds to a simple credential expression (i.e. to a conjunction). The role/credential matrix has at least one row for each role (i.e. one row for each disjunction). The entries in the row indicate simple credential expressions that, if all are provided by a user, will result in the user being granted that role. When a user provides their credentials, we can determine using this matrix the roles to be assigned.

Such a matrix can be stored as a set of bit strings with each representing a row. Then, finding the roles for a specific user amounts to a partial match search of the table using a partial match query string with *'s for the credentials provided and 0's for credentials not provided. For example, if the user has credentials matching an Admissions Clerk, then the partial match query is "000**". The *hits*, or matches, are the two rows for Employee and Admissions Clerk.

It is possible for a user's credentials to trigger several roles, which may conflict with respect to privileges. This is an issue we discuss in the next subsection.

TABLE 1. Role/Credential Matrix

	medDegree	**medDegree. speciality= rad**	**medDegree. speciality= car**	**employee**	**employee. position= adminClerk**
Doctor	1	0	0	1	0
Radiologist	1	1	0	1	0
Cardiologist	1	0	1	1	0
Admissions Clerk	0	0	0	1	1
Employee	0	0	0	1	0

4.3. Role/Object privilege matrix

The privileges associated with each role can also be summarized in a matrix, called the Role/object privilege matrix, as shown in Table 2. In the matrix, each row represents a role and the columns specify the privilege the role has, or does not have, depending on whether the privilege is positive or negative for a DL object. Next, we consider examples to illustrate the privilege algebra.

TABLE 2. Role/object Matrix

	Patient_Care .header	**Patient _Care**	**Patient_Care .findings**
CEO		b	
Doctor		b	u
Admissions _Clerk	u	b	-b

Example 6. First, consider the privileges of the Admissions Clerk given in Table 2. In this case there is browse privilege for the whole Patient Care record and that privilege is over-ridden for two components of the record: the clerk is allowed to update the header, but the clerk is not allowed to browse the findings component. We express the privileges of the Admissions Clerk as the following algebraic expression:

<b, Patient_Care> + <u, Patient_Care.header>
- <b, Patient_Care.findings>

Using the algebra, we simplify the expression as:

= <b, Patient_Care / findings> + <u, Patient_Care.header>

This simplified expressions shows exactly the access the clerk has to various components of a Patient Care record; this expression has positive permissions.

Example 6 shows a simplification for one role. When a user has more than one role, conflicts may arise which are resolved using the privilege algebra. The privilege algebra specifies the effect of combining privileges for different roles.

Example 7. Now, using Table 2, consider a person who is both the CEO and a doctor in the organization. The CEO has the privilege to browse the Patient Care record; a doctor can also browse a Patient Care record, and can update the findings section. The privileges that result when these two roles are combined in one person are determined by adding, according to the privilege algebra, the collection of privileges:

<b, Patient_Care> + <b, Patient_Care>
+ <u, Patient_Care.findings>

= <b, Patient_Care> + <u, Patient_Care.findings>

Example 8. Next, using Table 2, we consider a person with the credentials for Doctor and for Admissions Clerk. In this case we have an apparent conflict because a Doctor has update privilege for findings, but an Admissions Clerk is not allowed to browse that component. Combining the two sets of privileges (positive privileges appear first, followed by negative privileges) we have:

<b, Patient_Care> + <u, Patient_Care.findings>
+ <b, Patient_Care> + < u, Patient_Care.header>
- <b, Patient_Care.findings>

We eliminate the redundant expression of <b, Patient_Care>:

= <b, Patient_Care> + <u, Patient_Care.findings>
+ < u, Patient_Care.header> - <b, Patient_Care.findings>

We add the positive terms:

= <b, Patient_Care> ∧ <u, Patient_Care.findings ∪
Patient_Care.header> - <b, Patient_Care.findings>

Now we subtract <b, Patient_Care.findings> from both positive privileges:

<b, Patient_Care / findings> ∧ < u, Patient_Care.header>

The result is an expression specifying the privileges of a Doctor/Admissions Clerk.

5. Conclusion

In this paper, we have considered that users of a digital library may be anonymous, but recognized according to the credentials they provide when they access or update information in the library. A user may have credentials related to several roles, and so the privileges a user is granted may be several and this collection of privileges may lead to apparent conflicts. We have proposed a simple privilege algebra to control the evaluation of a privilege expression and to eliminate any ambiguity. The result is an unabiguous determination of the access rights appropriate for a provided set of credentials.

References

[1] J. Clark, Comparison of SGML and XML, http://www.w3.org/TR/NOTE-sgml-xml-971215, December 1997.

[2] Nabil R. Adam, Vijay Atluri, Elisa Bertino, Elena Ferrari, A content-based authorization model for digital libraries; *IEEE Transactions on Knowledge and Data Engineering, Vol. 14*, No. 2, March/April 2002.

[3] HL7 SGML/XML Special Interest Group website, www.mcis.duke.edu/standards/HL7/committees/sgnl/.

[4] Robert H. Dolin et al; HL7 document patient record architecture: an XML document architecture based on a shared information model, *JAMIA Fall Symposium Supplement*, 1999.

[5] M. Winslett, N. Ching, V. Jones, I. Slepchin, Using digital credentials on the world-wide-web, *Journal of Computer Security, Vol. 5*, Issue 3, December 1997.

[6] David Ferraiolo, Richard Kuhn, Role-based access control, *Proceedings of 15th. National Computer Security Conference*, 1992.

[7] Ravi S. Sandhu, Edward J. Coyne, Hal L. Feinsten, Charles E. Youman, Role-based access control models, *IEEE Computer, Volume 29*, Number 2, February 1996.

[8] S. L. Osborn, R. Sandhu, Q. Munawer, Configiuring role-based access control to enforce manadatory and discrentionary access control policies, *ACM Transactions on Information and System Security (TIS-SEC), Volume 3*, Number 2, *ACM Press*, 2000, pp. 85-106.

[9] Joon S. Park, Ravi Sandhu; RBAC on the web by smart certificates, *Proceedings of the Fourth ACM Workshop on Role-based Access Control*, October 28-29, 1999, 1-9.

[10] J. B. D. Joshi, W. G. Aref, A. Ghafoor, E. H. Spafford, Security models for web-based applications, *Communications of the ACM, Volume 44*, Issue 2, February 2001.

[11] Rafae Bhatti, Elisa Bertino, Arif Ghafoor, James B. D. Joshi, XML-Based Specification for Web Services Document Security, *IEEE Computer*, April 2004.

[12] Deborah L. McGuinness; Ontologies come of age, in *Spinning the semantic web: bringing the world wide web to its full potential* (editors D. Fensel, J. Hendler, H. Lieberman, W. Wahlster, MIT Press, 2002).

[13] H. Chen, A machine learning approach to document retrieval: an overview and an experiment, *Proceedings of the 27th. Hawaii International Conference of System Sciences, Volume 3*, IEEE Computer Science Press, 1994, pp. 631-640.

[14] Salvador Nieto Sanchez, Evangelos Triantaphyllou, Donald Kraft, A feature mining based approach for the classification of text documents into disjoint classes, *Information Processing and Management 38* (2002) 583-604.