SMARXO: Towards Secured Multimedia Applications by Adopting RBAC, XML and Object-Relational Database

Shu-Ching Chen[1], Mei-Ling Shyu[2], Na Zhao[1]
[1]Distributed Multimedia Information System Laboratory, School of Computer Science
Florida International University, Miami, FL 33199, USA
[2]Department of Electrical & Computer Engineering
University of Miami, Coral Gables, FL 33124, USA
[1]{chens, nzhao002}@cs.fiu.edu, [2]shyu@miami.edu

ABSTRACT

In this paper, a framework named SMARXO is proposed to address the security issues in multimedia applications by adopting RBAC (Role-Based Access Control), XML, and Object-Relational Databases. Compared with the other existing security models or projects, SMARXO can deal with more intricate situations. First, the image object-level security and video scene/shot-level security can be easily achieved. Second, the temporal constrains and IP address restrictions are modeled for the access control purpose. Finally, XML queries can be performed such that the administrators can proficiently retrieve useful information from the security roles and policies.

Categories and Subject Descriptors

H.2.0 [**Database Management**]: General – *Security, integrity and protection*

General Terms

Management, Security

Keywords

Multimedia Security, Role-based Access Control (RBAC), XML, Object-Relational Databases

1. INTRODUCTION

With the rapid development of various multimedia technologies, more and more multimedia data are generated in the medical, commercial, and military fields, which may include some sensitive information which should not be accessed by or can only be partially exposed to the general users. Therefore, user-adaptive multimedia data access control has become an essential topic in the areas of multimedia database design and multimedia application development for the national security purpose. RBAC (Role-Based Access Control) is a good candidate for user authorization control. However, most of the existing RBAC models mainly focus on document protection without fully considering all the possible environmental constraints. Although it is claimed that some extended models are able to offer the protection on multimedia files, there are still some problems not solved. For instance, Figure 1(a) shows an image which can be accessed but the "plate" object inside should not be displayed. That is to say, if a user requests this image, he/she can only view the partial image as shown in Figure 1(c).

The focal goal of our research can be outlined as constructing a framework to control the access to multimedia applications, files, and furthermore the visual/audio objects or segments embedded in the multimedia data. In this paper, we architect a framework named SMARXO (**S**ecured **M**ultimedia **A**pplication by adopting **R**BAC, **X**ML [6] and **O**RDBMS). Several significant techniques are proficiently mixed in SMARXO to satisfy the complicated multimedia security requirements. First, efficient multimedia analysis mechanisms can be utilized to acquire the meaningful visual/audio objects or segments. Second, XML and object-relational databases are adopted such that proficient multimedia content indexing can be easily achieved. Third, we upgrade and embed a dominant access control model which can be tailored to the specific characteristics of multimedia data. Moreover, XML is also applied to organize all kinds of security related roles and policies. Finally and most importantly, these techniques are efficiently organized such that multi-level multimedia access control can be achieved in SMARXO without any difficulty.

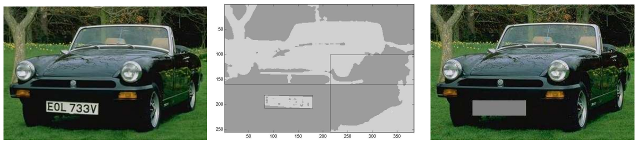

Figure 1. Example on image object-level security
(a) original image (b) segmentation map (c) hiding a portion of the image

2. RELATED WORK

A fundamental feature of RBAC is to support the administration of large numbers of privileges on system objects, and reduce the effort to define and manage complex security policies. Traditional RBAC models [5] have a lot of restrictions on access control modeling. Therefore, numerous extended RBAC models are emerged to handle those unsolved issues. In [1], the Temporal Role-Based Access Control (TRBAC) model which brings the basic temporal dependencies was proposed, but it cannot handle several useful temporal variables including the constraints on user-role and role-permission assignments. The Generalized Temporal Role-Based Access Control (GTRBAC) model [3] was proposed later to solve this problem. However, these two models only improved the control capability on temporal constraints. [4] proposed the Generalized Role-Based Access Control (GRBAC) model which leverages the traditional RBAC by incorporating subject roles, object roles, and environment roles. But they only introduced the temporal constraints in the environment roles, and it can only handle access control on multimedia files without taking care of multimedia contents. Another Generalized Object-

MM'04, OCTOBER 10–16, 2004, NEW YORK, NEW YORK, USA.

Composition Petri-Net Model (GOCPN) [2] was proposed, which mainly focuses on the modeling of documents to allow secure accesses to a multimedia database management system. GOCPN utilizes a mandatory access control (MAC) approach which cannot fully perform complicated roles, role hierarchies, temporal constraints, and IP address restrictions.

In SMARXO, the RBAC model is enhanced and utilized to manage complicate roles and role hierarchies. Moreover, the multimedia documents are indexed and modeled such that access control can be facilitated on multi-level multimedia data. The comparison among SMARXO and these existing security models/approaches is depicted in Table 1.

Table 1. Comparison of Multimedia Security Techniques

Support	$RBAC_3$	TRBAC	GTRBAC	GRBAC	GOCPN	SMARXO
Access Control	Yes	Yes	Yes	Yes	Yes	Yes
Role Hierarchy	Yes	Yes	Yes	Yes	No	Yes
Temporal Constraints	No	Yes	Yes	Yes	No	Yes
IP address Restrictions	No	No	No	No	No	Yes
Security on Multimedia Data	No	No	No	Yes	Yes	Yes
Security on Multilevel Objects	No	No	No	No	Yes	Yes

3. SMARXO ARCHITECTURE

There are three phases available in order to build up the complete security verification architecture for multimedia applications. Figure 2 illustrates the SMARXO architecture. The multimedia data, extracted features, and furthermore the XML documents are all organized in the ORDBMS. Once a user (including the administrator) logs in the system and requests the multimedia data, the security checker verifies user's identification and the related permission. The multimedia manager responds based on the security checking results. The source multimedia data may need to be processed in order to hide the object-level or scene/shot-level information. In addition, through this framework, the administrators are capable of creating, deleting, and modifying the user roles, object roles, temporal roles, IP address roles, and security policies. Since all the protection related information is managed by XML, security information retrieval becomes very convenient.

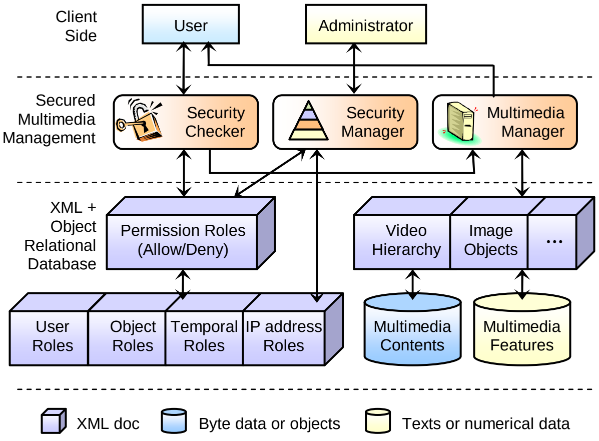

Figure 2. SMARXO architecture

4. MULTIMEDIA ACCESS CONTROL

The traditional RBAC methods need to be extended to perform superior access control functionalities such as the temporal and IP address control, object-level and scene/shot-level access control, etc. Based on the formal definition of traditional RBAC in [5], the extended formal definitions are given in Figure 3. Compared with the traditional RBAC model, we also introduce the object roles, temporal roles, and IP address roles. The associated rules are defined such that these advanced roles can be combined to perform the inclusive access control.

Sets:

U:	Users	(*) *O*:	Objects
Ru:	User Roles	(*) *Ro*:	Object Roles
S:	Sessions	(*) *Rt*:	Temporal Roles
P:	Permissions	(*) *Ri*:	IP address Roles

Rules:

1) $UA \subseteq U \times Ru$: user-role assignment
2) $RH \subseteq Ru \times Ru$: a partial order of role hierarchy
3) $PA \subseteq P \times Ru$: a basic permission-user role assignment
4) (*) $OA \subseteq O \times Ro$: object-role assignment
5) (*) $OP \subseteq P \times Ro$: a permission-object assignment
6) (*) $R \subseteq Ru \times Rt \times Ri$: an assembled role set with environmental constraints
7) (*) $OPA \subseteq OP \times R$: an advanced permission-role assignment
8) *user*: $S \rightarrow U$, a function mapping a session to a user
9) *u_roles*: $S \rightarrow 2^{Ru}$, a basic function mapping a session to a set of user roles
10) (*) *roles*: $S \rightarrow 2^{R}$, an advanced function mapping a session to a set of roles
11) *permissions*: $Ru \rightarrow 2^{P}$, mapping a user role to a set of permissions
12) *permissions'*: $Ru \rightarrow 2^{P}$, mapping a user role to a set of permissions with role hierarchies
13) (*) *permissions''*: $R \rightarrow 2^{OP}$, mapping an assembled role to a set of permissions
14) (*) *permissions'''*: $R \rightarrow 2^{OP}$, mapping an assembled role to a set of permissions with role hierarchies
15) $permissions(r) = \{p: P \mid (r, p) \in PA\}$
16) $permissions'(r) = \{p : P \mid \exists r' \leq r \cdot (r', p) \in PA\}$
17) (*) $permissions''(r) = \{p: OP \mid (r, p) \in OPA\}$
18) (*) $permissions'''(r) = \{p : OP \mid \exists r' \leq r \cdot (r', p) \in OPA\}$

(**Note:** the ones marked with * are advanced features of SMARXO)

Figure 3. Extended RBAC definitions in SMARXO

4.1 Multimedia Indexing Phase

In order to support multi-level security, the multimedia data are required to be stored hierarchically. For instance, by applying image segmentation techniques on Figure 1(a), the corresponding segmentation map (as shown in Figure 1(b)) can be achieved. Each extracted object is bounded with a rectangle. The extraction results may help people identify the meaningful objects and compute the associate bounding boxes. Both the original image and the image object information can be stored in the ORDBMS. If a specific security policy requires some portions of the target image to be hidden from the user, the system can retrieve the sub-object's attributes and process the original image to hide those portions (e.g., the protected "plate" in Figure 1(c)). XML can be adopted to index the image object information by a 6-tuple: <*o_id*, *o_name*, *o_x*, *o_y*, *o_width*, *o_height*>, which means the object id, object description, x and y coordinates of the top-left point, and the object width and height, respectively. Such an example can be found in Figure 4(a).

By utilizing video decoding, shot detection, and scene detection techniques, the specific video can be automatically segmented and diverse levels of the video objects can be achieved: *frame*, *shot*, *scent*, and *event*. For the purpose of video indexing, we can furthermore apply XML to store this kind of video hierarchy information. As shown in Figure 4(b), the start frame and end frame numbers of the shots are stored to mark the segmentation boundaries. In SMARXO, a "*shot*" is treated as the fundamental unit to store the video data for the efficiency purpose. Hence, shot-level security can be performed easily by displaying the accessible shots and skipping those prohibited shots. In addition, the users are allowed to manually identify their target multimedia objects or segments by giving the corresponding parameters.

```
(a)
<ImageObjects>
 <Image imgid='i001'>
 <Object o_id='i001o01'>
  <o_name>TAG</o_name>
  <o_x>40</o_x>
  <o_y>80</o_y>
  <o_width>8</o_width>
  <o_height>50</o_height>
 </Object>
 <Object o_id='i001o02'>
  <o_name>CAR</o_name>
  …
 </Object>
 </Image>
…
</ImageObjects>
```

```
(b)
<VideoHierarchy>
 <Video v_id='v01'>
  <Event e_id='e01'>
   <Scene c_id='c01'>
    <Shot s_id='s01'>
      <frame_s>1</frame_s>
      <frame_e>89</frame_e>
    </Shot>
    …
   </Scene>
   …
  </Event>
  …
 </Video>
 …
</VideoHierarchy>
```

Figure 4. XML examples on multimedia hierarchy

(a) example for image objects (b) example for video hierarchy

4.2 Security Modeling Phase

In most of the multimedia applications, a request behavior can be briefly recognized by a 4-tuple: <*who*, *what*, *when*, *where*>. The meaning of this request is that some user requests some data at some time by using some computer. As we discussed before, most of the related research work can only control accesses by the "*who*" and "*what*" attributes. Few models can support security verification on the "*when*" attribute. By contrast, our framework supports all of them.

```
(a)
<SubjectRoles>
 <UserGroup default='Allow'>
  <Group g_id='Professor'>
   <User u_id='Bailey'>
    <Password>abc</Password>
   </User>
   …
  </Group>
</UserGroup>
 <UserGroup default='Deny'>
  <Group g_id='Student'>
   <User u_id='Smith'>
    <Password>321</Password>
   </User>
   …
  </Group>
  …
 </UserGroup>
</SubjectRoles>
```

```
(b)
<ObjectRoles>
 <o_group id='Shots_a' >
  <scene s_id='s02'>
   <shot>2</shot>
   <shot>3</shot>
   <shot>4</shot>
   …
  <scene>
  …
 </o_group>
 <o_group id='Shots_b'>
   <shot>6/shot>
   <shot>12</shot>
   …
 </o_group>

…
</ObjectRoles>
```

Figure 5. XML examples on the fundamental roles

(a) example on subject roles (b) example on object roles

User roles, also recognized as "Subject roles", are the most fundamental feature of RBAC. In addition to the basic requirements, SMARXO supports one more specific feature on user authorization. When the administrator creates a new user account, he/she can choose the default property of this user from two options. One is to initially grant all the access abilities to this user, and then assign the roles which deny this user's access to some object. The other option is to disable the user from accessing by default. Then the permission roles can be granted to this account. For instance, in Figure 5(a), the user "*Bailey*" in the "*Professor*" group is assigned the default value "*Allow*"; while the user "*Smith*" in the "*Student*" group is assigned the default property "*Deny*."

Sometimes, the user may not be able to access one or more segments/objects of a multimedia file. However, he/she should be able to access other parts of this file. The object roles are facilitated to satisfy this requirement. For instance, Figure 6 illustrates a video shot sequence stored in the database. User A cannot access shots 2, 3, 4; while User B cannot access shots 6 and 12. However, A and B should be allowed to view the other shots of this video except their prohibited segments. SAMRXO supports this kind of access control by modeling both the object roles and multimedia hierarchy information. Figure 5(b) depicts an XML example for the object roles. Furthermore, in order to efficiently organize plentiful object roles, we introduce the object-role hierarchy which is defined as follows.

***Definition* 1**: An Object Hierarchy $OH=(O,OG,\le_{OG})$, where O is a set of objects and $OG=O\bigcup G$ with G is a set of object groups. $\le$ is a partial order on OG called the dominance relation, and $O\subseteq OG$ is the set of minimal elements of OG with respect to the partial order. Given two elements $x,y\in OG$, $x\le_{OG} y$ iff x is a member of y.

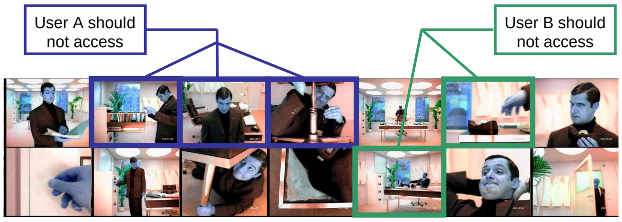

Figure 6. Example requirements for video scene/shot-level access control

```
(a)
<TemporalRoles>
 <tGroup e_id='Holiday'>
  <Holiday h_id='Thanksgiving'>
   <Month>11</Month>
   <WeekNo>4</WeekNo>
   <WeekDay>4</WeekDay>
  </Holiday>
  …
 </tGroup>
 <tGroup e_id='OfficeHour'>
  <H_interval>
   <H_start>9</H_start>
   <H_end>17</H_end>
  </H_interval>
 </tGroup>
 …
</TemporalRoles>
```

```
(b)
<SpatialRoles>
 <ipGroup ipg_id='University'>
  <ipUniv ipu_id='FIU'>
   <ipDept ipd_id='SCS'>
    <seg1_fix>131</seg1_fix>
    <seg2_fix>94</seg2_fix>
    <seg3_fix>133</seg3_fix>
    <seg4_start>1</seg4_start>
    <seg4_end>255</seg4_end>
   </ipDept>
   …
  </ipUniv>
  …
 </ipGroup>
…
</SpatialRoles>
```

Figure 7. XML examples on the optional roles

(a) example on temporal roles (b) example on IP address roles

In a multimedia application, data may be available to the users at certain time periods but unavailable at others. In order to achieve this target, the temporal constraints can be generally formalized with the following attributes: *year*, *month*, *week number*, *week day*, *hour*, *minutes*, etc. As shown in Figure 7(a), "*Thanksgiving*" is depicted with three attributes, which means that Thanksgiving is the fourth Thursday of November. The other temporal role named "*OfficeHour*" illustrates that the office hours are from 9 o'clock to 17 o'clock every day.

Even for the same user, he/she may be able to access the multimedia data only by using some specific computers. The IP addresses can be utilized to embed this kind of constraints by identifying the different networks and clients. Usually, an IP address appears in the equivalent dotted decimal representation such as 10.0.0.1 and each octet in it ranges from 0 to 255. By checking the associated IP address, the server can judge whether this access is allowed. For this purpose, we define the IP address segment for the related role modeling.

***Definition* 2:** Given the octets name I_1, I_2, I_3, I_4, the IP address segment expression A can be defined as $A=\sum_{j=1}^{n} x_j \cdot I_j \rhd y_j \cdot I_d$, where n = 4, $0\le x_j \le 2^8-1$, $0\le y_j\le 2^8-1$, $x_j, y_j \in N$, $x_j + y_j \le 2^8-1$ for j = 1, …, 4, $I_d \in \{I_1, I_2, I_3, I_4\}$.

The symbol $\rhd$ identifies the set of starting points of the intervals. For example, $131\cdot I_1 + 94\cdot I_2 + 133\cdot I_3 + (1\cdot I_4 \rhd 254\cdot I_4)$ stands for the segment between 131.94.133.1 and 131.94.133.255. It can be modeled by XML as shown in Figure 7(b), which also means this segment is under the role of University→FIU→SCS.

The security policies in the traditional policies are basically classified into two categories. One is "*Allow Policy*" which means that some user can access some object; the other is "*Deny Policy*" which means that some user cannot access some object. SMARXO introduces "*Partial Allow Policy*" which means that the user can only access partial data of this object. The definition of security policy is given in Figure 8(a) with a 5-tuple. Figure 8(b) gives a policy example which means that the "*Student*" can access "*Shots_a*" in "*Holiday*" by using the machines of "*SCS*."

(a)

A security policy can be a 5-tuple:
<*Ru, Ro, Rt, Ri, Acc*> Where:
Ru: a user role;
Ro: an object role;
Rt: a temporal role;
Ri: an IP address role;
Acc: accessibility, the value can be *Allow*, *Deny*, or *PartiallyAllow*.

(b)

```
<PolicyRoles>
 <policy p_id='p01'>
  <Ru>Student</Ru>
  <Ro>Shots_a<Ro>
  <Rt>Holiday<Rt>
  <Ri>SCS</Ri>
  <Acc>Allow</Acc>
 </policy>
 …
</PolicyRoles>
```

Figure 8. Security policies

(a) formalized security policy (b) XML example on policy roles

4.3 DBMS Management Phase

In this framework, the multimedia features, XML documents, and the multimedia contents are stored into an ORDBMS. By efficiently managing the XML segments in the ORDBMS, the XML documents can be easily updated when editing the security policies or the multimedia hierarchy information. Moreover, all the contents prepared in XML can be searched easily and accurately. In other words, it is very convenient for the administrator to retrieve the security policies by performing XML queries in the ORDBMS. Furthermore, ORDBMS provides some valuable functionality to store the byte data and large objects. Therefore, the images as well as the video shots can be professionally managed.

4.4 Security Verification

Based on an access request, the system will first check user ID and password, and then check the user roles, object roles, temporal roles, and IP address roles consequently. After that, the security policy checks are performed on the "Object Entity Set" (OES) of the request object *o* that includes both the object itself and all the entities *s* (segments or sub-objects belong to *o*).

***Definition* 3**: Object Entity Set: $OES(o) = \{o\} \bigcup \{s : s \in o\}$.

Figure 9 depicts the security verification algorithm. A brief function "*p_check*(*o*)" is presumed to check if the user can access object *o* in the specified time from some specified computer. Three kinds of results can be formalized as follows:

1. The access will be denied *iff* $p\_check(o) = FALSE$.
2. A user can access the original multimedia data *o iff*

$$\forall t \in OES(o)[p\_check(t) = TRUE],$$

where t can be any entity including *o* and all *o*'s sub-objects.

3. A user can access the processed multimedia data *o'* where the prohibited sub-objects are removed from *o iff*

$$(p\_check(o) = TRUE) \wedge (\exists s \in o[p\_check(s) = FALSE]),$$

where *s* can be any sub-object or segment which belongs to *o*.

Input: An Access Request <*id*, *pwd*, *time**, *ip_addr**, *object*>

Output: (1) *FALSE*: Access is denied;
(2) *object*: Complete multimedia data as requested;
(3) *object'*: Processed multimedia data without the protected objects.

Algorithm *security_check*(*id*, *pwd*, *time**, *ip_addr**, *object*):

```
BEGIN
if (id, pwd) ∉ U                          //Verify user identity
    return FALSE;
else
    if (get_user_role(id))                //Check user-role assignment
        u_role = get_user_role(id);
    else u_role = id;
    if (get_object_role(object))          //Check object-role assignment
        o_role = get_object_role(object);
    else o_role = object;
    if (get_temporal_role(time))          //Check temporal-role assignment
        t_role = get_temporal_role(time);
    else t_role = time;
    if (get_IPaddr_role(ip_addr))         //Check IP address role assignment
        ip_role = get_IPaddr_role(ip_addr);
    else ip_role = ip_addr;
    if (check_permission(u_role, o_role, t_role, ip_role)=DENY)
        return FALSE;
    else
        for all sub_object ∉ object       //Check permission on the sub-objects
        if (check_permission(u_role, sub_object, t_role, ip_role)=DENY) {
            object' = security_process(object)      //Process multimedia data
            return object'; }             //User can access the processed object
        else
            return object; //User can access the complete object
END
```

(**Note:** Features marked with * are advanced ones but optional in SMARXO.)

Figure 9. Algorithm for security verification in SMARXO

5. CONCLUSIONS

This paper proposes a practical framework – SMARXO that can be employed in multimedia applications to perform multilevel multimedia security. RBAC, XML and ORDBMS are efficiently combined to achieve this target. Currently, we are in the process of applying, testing, and optimize this framework in our distributed multimedia management system.

6. REFERENCES

[1] Bertino, E., Bonatti, P. A. and Ferrari E. TRBAC: A Temporal Role-Based Access Control Model. *ACM Transaction on Information and System Security (TISSEC)*, Vol. 4, No. 3, August 2001, 191-233.

[2] Joshi, J., Li, K., Fahmi, H., Shafiq, B. and Ghafoor, A. A Model for Secure Multimedia Document Database System in a Distributed Environment. *IEEE Transaction on Multimedia*, Vol. 4, No. 2, June 2002, 215-234.

[3] Joshi, J., Bertino, E., Shafiq, B. and Ghafoor, A. Dependencies and Separation of Duty Constraints in GTRBAC. In *Proceeding of 8th ACM Symposium on Access Control Models and Technologies (SATMAC 2003)*, Como, Italy, June 2003, 51-64.

[4] Moyer, M. J. and Ahamad, M. Generalized Role-Based Access Control. In *Proceedings of the 21st International Conference on Distributed Computing Systems (ICDCS 2001)*, April 2001, 391-398.

[5] Sandhu, R.S., Coyne, E.J., Feinstein, H.L. and Youman C.E. Role Based Access Control Models. *IEEE Computer*, Vol. 29, No. 2, February 1996, 38-47.

[6] Extensible Markup Language (XML) 1.0 (Second Edition) – W3C Recommendation 6 October 2000. http://www.w3.org/TR/2000/REC-xml-20001006.pdf