The Image Protector - A Flexible Security Rule Specification Toolkit

Bechara Al Bouna, Richard Chbeir, Alban Gabillon

THE IMAGE PROTECTOR
A Flexible Security Rule Specification Toolkit

Bechara Al Bouna [1], Richard Chbeir[2] and Alban Gabillon[3]
[1]*Ticket Laboratory, Antonine University,Baabda, Lebanon*
[2] *LE2I Laboratory UMR-CNRS University of Bourgogne, Dijon, France*
[3] *Laboratoire GePaSud (Géosciences du Pacifique Sud), Université de la Polynésie Française*
bechara.albouna@upa.edu.lb, richard.chbeir@u-bourgogne.fr, alban.gabillon@upf.pf

Keywords: Multimedia Security, Content Protection, Privacy Preserving, Security Rules

Abstract: The tremendous sharing of multimedia objects on the web shed the light on several privacy concerns related in essence to the safe publishing of end users' personal data. Providing techniques to protect multimedia objects faces several difficulties due to multimedia objects' heterogeneous and complex structure on one hand, and on the other hand, the wide range of information that could be used to describe their content. In this paper, we present a flexible security rule specification toolkit for multimedia objects. Our toolkit is based on a security model and a core ontology in which we populate the model's related information and multimedia objects data. To specify security rules, we use the SWRL language in order to address both, the content and the context of multimedia objects.

1. INTRODUCTION

Web technologies have known an unprecedented growth in the last decade enabling easy publishing and sharing of multimedia objects (such as images, videos and audio). It is true that these multimedia objects are used in several application domains (medical, education, etc.) but mainly exploited for personal usage. However, their massive use caused several problems related to privacy and confidentiality preserving and current tools (operating systems, web-based sharing applications, DBMS, etc.) are still un-adapted and provide restrictive dedicated functionalities. Several attempts and studies were proposed recently by the research community to protect multimedia objects (and particularly photos and videos). (Fan, J., Luo, H., Hacid, M. S. and Bertino E. 2005) proposed a framework for privacy-preserving in video sharing dealing with owner-adaptive video privacy modeling, video content privacy protection and statistical inference control. The authors (Joshi, J. B. D., Kevin Li, Z., Fahmi, H., Shafiq, B and Ghafoor, A. 2002) presented a security model for distributed document management system. As for the approach discussed by (Kodali, N., Farkas, C. and Wijesekera, D. 2004), the adopted framework guaranteed the original security needs while allowing transparent access and flexible security rule manipulation. Despite the efficiency of the proposed approaches, multimedia objects content protection remains difficult due to multimedia objects complex structure. On the basis of the model proposed by Gabillon, A, and Capolsini, P, 2010a, Gabillon, A, and Capolsini, P, 2010b, we describe here a java-based prototype, called `Image Protector`, developed to handle image content protection and visualization. The originality of this model is that it includes a framework for specifying the access control mechanisms that should be used whenever a prohibition is derived from the security policy. Our prototype is divided into two main modules: 1) a BackOffice module providing the authorization manager the ability to specify flexible security rules based on the First Order Logic (FOL) and written in SWRL (Horrocks, I., Patel-Schneider, P. F., Boley, H. Tabet, S. Grosof, B. and Dean. M. 2004.), and 2) a FrontOffice module allowing end-users to visualize authorized multimedia objects[1].

[1] Our prototype handles images for now thus it will be extended to handle videos and audio segments. We

A web-based demonstration of the prototype can be found on http://www.upa.edu.lb/ImageProtector/

The reminder of this paper is organized as follows. In Section 2, we give a brief overview on the security model adopted. In Section 3, we present `Image Protector` and describe the core ontology used to store information related to the access control and multimedia objects. We also describe Back and Front office modules before concluding our work in Section 4.

2. SECURITY MODEL

In this section, we present the proposed security model based on first-order logic, a formal language and a set of inference rules to represent security constraints. Our security model is a 4-tuple defined as `<subjects, objects, actions, security rules>` and detailed in the following subsections

2.1.1 Subjects

Subjects are users or processes running on behalf of users. We define S as the set of subjects. Predicate `Subject(s)` reads "s is a user". We assume that each subject has associated attributes so to describe its identity, role, age, location and so on. Each attribute can correspond to a predicate in our model. It is to be noted that a subject s can be associated to one of several multimedia objects (picture, fingerprint image, etc.)

2.1.2 Objects

Objects are basically multimedia objects. We define $O \subset MO$ the set of objects. Predicate `Object(o)` reads "o is an object". We also assume here that each object has associated attributes and each attribute can correspond to a predicate.

2.1.3 Actions

Actions correspond to dedicated operations to visualize create, update, delete images/videos and to play, record, update sounds. We define A the set of actions. Predicate `Action(a)` reads "a is an action". We assume that each action has associated attributes to define its type, category and various parameters. Each attribute can also correspond to a predicate.

2.1.4 Security Rules

Security rules specify how subjects can execute actions on objects. Our model includes permissions (positive rules) and prohibitions (negative rules). We define a positive authorization rule as a logical rule having the following form:

$$\forall s \forall a \forall o(Condition \rightarrow Permit(s; a; o))$$

Permit(s; a; o) reads "s is permitted to execute action a on object o". We define a negative authorization rule as a logical rule having the following form:

$$\forall s \forall a \forall o(Condition \rightarrow Deny(s; a; o) \wedge Protect(o;M))$$

Deny(s;a;o) reads "s is forbidden to execute action a on object o". Protect(o;M) is optional and reads "o is protected by using method M". M denotes a protection method or one of the followings: `blur`, `spiral`, `access_denied`, `request rejected, etc.`

If, for a given prohibition, there is no specified protection mechanism then a default mechanism applies. This default mechanism depends on the application. For example, the following default rule says that the default mechanism is `blur`.

$$\forall s \forall a \forall o(Deny(s; a; o) \rightarrow Protect(o; \texttt{blur}))$$

In our study, authorizations propagate to sub-objects i.e., the following entailments should be considered as part of the security policy:

E1:
$\forall s \forall a \forall o_1 \forall o_2(Permit(s; a; o_1) \wedge Belong(o_2; o_1) \rightarrow Permit(s, a, o_2))$

E2:
$\forall s \forall a \forall o_1 \forall o_2(Deny(s; a; o_1) \wedge Belong(o_2; o_1) \rightarrow Deny(s; a; o_2))$

3. IMAGE PROTECTOR PROTOTYPE

`Image Protector` provides a simple and efficient authorization management toolkit for images and photos. Descriptions and properties of each image are stored in a core ontology and security rules are written in **Semantic Web Rule Language** (**SWRL**) on top of the specified ontology so to provide the user with high expressive querying possibilities.

will use the terms multimedia objects and images interchangeably

Before we detail `Image Protector` modules, we will first present the core ontology used to describe the security model and store images related data.

3.1 Ontology-Based Security Model

When multimedia data come to play, the use of ontologies is required to fill the semantic gap between the users and the provided application (particularly located in the semantic descriptions and their mapping into low-level features of multimedia objects such as colors, texture, shapes, etc.). This has been proven in many situations (Masoumzadeh, A and Joshi, J.: 2010) (Chen, T.Y. 2008) For this reason and also to provide extensible framework, we adopt in our work an ontology-based storing approach.

Images features and semantic descriptions are stored in the core ontology (MO class) whereas their content (i.e., raw data) are either stored locally or on the Web (while referred to using URIs or URLs).

Figure 1 illustrates how instances related to the access control model and the multimedia objects are interconnected.

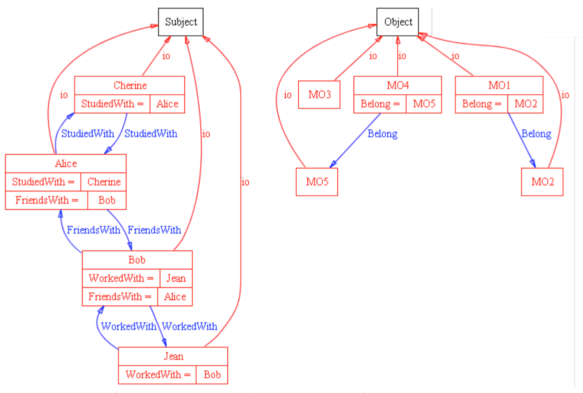


Figure 1: Sample core ontology

In the following, we will describe Back and Front Office modules along with their components.

3.2 BackOffice Module

This module helps the authorization manager to specify who has the right to manage a multimedia object and/or its content. In its current version, `Image Protector` allows to associate only `view` action. Figure 2 shows the architecture

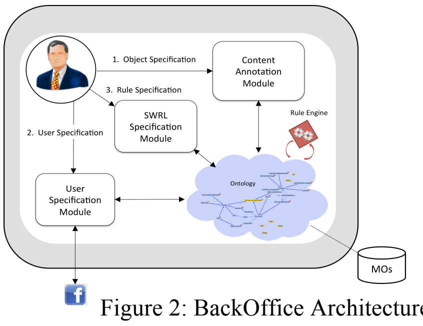


Figure 2: BackOffice Architecture

of this module.

Using the **SWRL specification module**, the authorization manager can define flexible security rules written in SWRL by allowing him/her to: 1) map the multimedia objects to the core ontology and to map semantic descriptions to their content using the **content annotation module**, 2) define subjects (users wishing to visualize images using the front office module) using the **user specification module**, and 3) specify actions and protection mechanisms (in the current version, we provide three main mechanisms such as pixelize, blur, and spiral).

3.2.1 Content Annotation Module

The content annotation module is used to annotate images in order to enrich their content with semantic descriptions. Our annotation module includes the possibility to manually specify any object of interest (drawing minimum bounding rectangles) and assign their corresponding description. In order to facilitate the annotation process, we provide also an automatic blob and face detection functions. For that, we used the OpenCV API (Bornet, OpenCV. 2000) to auto-detect objects of interests contained in an image. Furthermore, we used a face recognition function which is based on Eigen face (Turk, M. and Pentland, A. 1991) in order to recognize possible faces already annotated.

3.2.2 SWRL Specification Module

The SWRL specification module helps managing rules logically using facts and relations. It is an FOL based language formed by an implication between an antecedent (body) and consequent (head). In our prototype, SWRL is used to specify security rules and enrich the ontology with inferred common knowledge. We used Protégé[2] SWRL tab to provide an easy way to administrate rules and handle syntactical errors. In fact, we integrated the SWRL tab as an embedded component in the authorization panel.

In the following, we present some sample rules used to handle authentications and authorizations.

3.2.2.1 Authentication rules

These rules are used to guarantee safe access to the frontOffice module. Below, we provide some sample ones.

[2] http://protege.stanford.edu/

Rule 1: Allow all users to access the image viewer of the FrontOffice

User(?x) → Authentication(?x, true)

Rule 2: Prevent user `Bob` from accessing the front office

UserName(?x, `Bob`*) → Authentication(?x, false)*

Rule 3: Deny young users (under 12) to access the front office

Age(?x,?age) swrlb:lessThanOrEqual(?age,12) →Authentication(?x, false)

Rule 4: Deny `bob` from accessing the image viewer if he is not located in one of the university lab with a 0.7 threshold

UserName(?x, `Bob`*) ∧ swrlb:notLocatedIn(*`InUPALab`*, 0.7) → Authentication(?x, false)*

We note that *swrlb:notLocatedIn(*`InUPALab`*,0.7)* is a user defined predicate used to verify whether the authenticating user is located in the controlled lab at the authentication time. The predicate accepts as input a captured image and become valid if this image is similar to the predefined one representing the related lab (*`InUPALab`*).

3.2.2.2 Authorization rules

Authorization rules guarantee safe management of multimedia objects. Below, we provide some sample rules related to controlling the visualization of images and their content (since the current version of our prototype only handles images). For each of the following rules, if the conclusion does not include the Protect predicate then the default protection mechanism applies.

Rule 5: Prevent `Bob` from seeing all images annotated with a term “Jean”

Username(?x, `Bob`*) ∧ Annotation(?y,* `Jean`*) → Deny(?x,View,?y)*

Rule 6: Prevent `Bob` from seeing all images annotated with “Alice” using spiral as a protection type

Username(?x, `Bob`*) ∧ Annotation(?y,* `Alice`*) → Deny(?x,View,?y) ∧ Protect(?y,* `Spirale`*)*

Rule 7: Prevent `Bob` from seeing images of the friends of “Dupond”

Username(?x, `Bob`*) ∧ Annotation(?y,* `Dupond`*) ∧ FriendsOf(?y,?z) → Deny(?x, View,?z)*

Rule 8: Prevent `Bob` from seeing `Dupond`’s annotated images if he is not located in one the University Lab.

Username(?x, `Bob`*) ∧ Annotation(?y,* `Dupond`*) ∧ swrlb:notLocatedIn(*`InUPALab`*, 0.7) → Deny(?x, View,?y).*

Rule 9: Prevent `Bob` from seeing all objects similar to a given picture (img.jpg).

Username(?x, `Bob`*) ∧ swrlb:Sim(?y,* `img.jpg`*, 0.6) → Deny(?x View,?y)*

We note that *swrlb:Sim(?y,* `img.jpg`*, 0.6)* is a user-defined predicates based on the Eigen face recognition function.

3.2.3 Rule Engine

A rule engine is used to process predefined SWRL rules on the ontology and infer potential information. Currently, several existing engines are provided in the literature such as Jess, Bossam[3], Hoolet[4], Pellet[5], KAON2[6], RacerPro[7], FaCT++[8], etc. In our prototype, we adopted jess API (Sandia National Laboratories) which is a java-based rule engine free for academic purpose.

The inferred result of security rules is stored in the Deny and Permit *object properties*. That is authorized (denied respectively) multimedia objects for a given user would be stored as a triplet of <user, action, object> in the object properties Permit (Deny respectively).

3.2.4 User Specification Module

The user specification module allows the authorization manager to add users to the core ontology. Using this module, the authorization manager is able to manually add new users and assign their related attributes or automatically retrieves users and their attributes from FaceBook. Furthermore, we enrich the core ontology with semantic relations between subjects determined using users’ profile. We defined five different semantic relations:

- areFriends: both users are friends on the social network
- StudiedTogether: users have studied at the same university or school

[3] http://bossam.wordpress.com/
[4] http://owl.man.ac.uk/hoolet/
[5] http://clarkparsia.com/pellet/
[6] http://kaon2.semanticweb.org/
[7] http://www.racer-systems.com/
[8] http://owl.man.ac.uk/factplusplus/

- WorkedTogether: users worked in the place
- LivedTogether: users have lived in the same region or area
- hasCommonAffiliation: users are members of the same network

Each of the semantic relations is generated by comparing fragments (see Figure 3) of information gathered from users' profiles using the edit distance and the N-Grams functions for string similarity. If both fragments match a score higher than a predefined one (which could be modified according to users' need), we consider that the related semantic relation exists between both users.

```
<work_info>
  <location>
  <country>
     France
   </country>
  </location>
 <company_name>
  LE2I
 </company_name>   <position>
  PHD Student
 </position>
 <description>
System and Network issues
  </description>
</work_info>
```

```
<work_info>
  <location>
        <country>
        France
        </country>
   </location>
  <company_name>
       LE2I
  </company_name>
  <position>
  PHDStudent
  </position>
  <description>
    Multimedia And Access
Control issues
 </description>
</work_info>
```

Figure 3: Profile information fragments describing the work history of two different users

In the next section, we will present the FrontOffice Module in which images are displayed and filtered out according to the authentication and authorization rules.

3.3 FrontOffice Module

The FrontOffice module is used to manage (query, visualize, etc.) stored multimedia objects. The architecture of this module is displayed in Figure 4

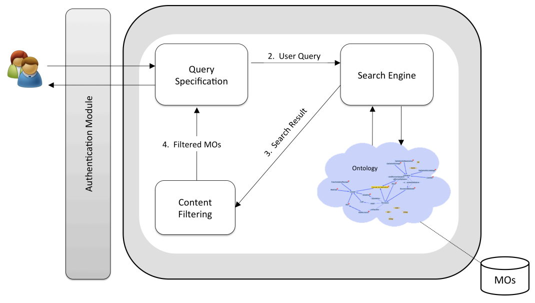


Figure 4: FrontOffice Architecture

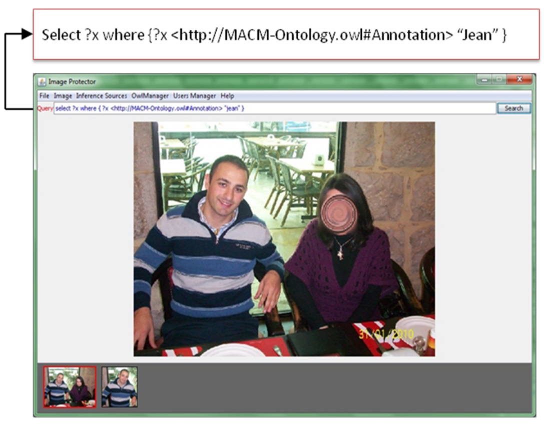
In order to access this module and use provided functionalities, users should be authenticated. Before displaying requested multimedia objects (currently only images), a filtering engine is used to remove and/or protect confidential content according to the predefined authorization rules.

3.3.1 Authentication Module

The authentication process is based on the username, password, and environmental context predicates (e.g., *swrlb:notLocatedIn)*.

3.3.2 Image Viewer Interface

The Image Viewer interface is used to execute queries and visualize the returned filtered result. In fact, denied multimedia objects within the returned result are protected using the content filtering engine according to the protection mechanism adopted for the authenticated user.



```
Select ?x where {?x <http://MACM-Ontology.owl#Annotation> "Jean" }
```

Figure 5: FrontOffice Interface

3.4 Search Engine

As mentioned earlier, in order to query data stored in the ontology, we used SPARQL (Prud'hommeaux, E and Seaborne, A. 2008) (see Figure 5). The search engine provides the ability to search through images' content and attributes as defined in the core ontology. The queried result is forwarded to a content filtering engine in order to protect confidential multimedia objects.

3.4.1 Content Filtering Engine

Content filtering engine works by searching the core ontology for the denied multimedia objects related to the authenticated user. In fact, the module queries the “Permit” and “Deny” object properties, finds possible intersections, and returns the filtered result to the Image Viewer Interface for visualization.

4. CONCLUSION

In this paper, we presented a prototype called `Image Protector` which is a flexible toolkit to specify authorization and authentication rules. The image protector is composed of two main modules: a backOffice module and a frontOffice module. The backOffice module is used to manage multimedia objects and specify security rules whereas the frontOffice module supports querying multimedia objects and displaying their filtered content. Several future work will be explored. In the current version of the prototype, videos and audio segments are not supported and we are aiming to provide the users with related functionalities. Furthermore, we would like to include in our prototype a dedicated conflict resolution method in order to solve possible intersections between permitted and denied multimedia objects.

ACKNOWLEDGEMENTS

This prototype has benefited from the work carried out by Mr. Jean Khayrallah from the Antonine University (Lebanon) during his graduation project.

REFERENCES

Gabillon, A, and Capolsini, P, 2010. Rule-based Policy Enforcement Point for Map Services. *In Proc. of the 3rd ACM SIGSPATIAL International Workshop on Security and Privacy in GIS and LBS.*

Gabillon, A, and Capolsini, P, 2010. Security Mechanisms for Geographic data. *In Proc. of the International Conference on Management of Emergent Digital EcoSystems (MEDES)*

Masoumzadeh, A and Joshi, J.: 2010. OSNAC: An Ontology-based Access Control Model for Social Networking Systems. *SocialCom/PASSAT* pp.751-759.

Horrocks, I., Patel-Schneider, P. F., Boley, H. Tabet, S. Grosof, B. and Dean. M. 2004. SWRL: A Semantic Web Rule Language Combining OWL and RuleML. [online] Available at: http://www.w3.org/Submission/SWRL/ [Accessed 21 July 2010]

Fan, J., Luo, H., Hacid, M. S. and Bertino E. 2005 A novel approach for privacy-preserving video sharing. *CIKM* pp609-616

Joshi, J. B. D., Kevin Li, Z., Fahmi, H., Shafiq, B and Ghafoor, A. 2002. A model for secure multimedia document database system in a distributed environment. *IEEE Transactions on Multimedia* 4(2): pp.215-234

Turk, M. and Pentland, A. 1991. "Face recognition using eigenfaces". *Proc. IEEE Conference on Computer Vision and Pattern Recognition*. pp.586–591.

Kodali, N., Farkas, C. and Wijesekera, D. 2004. An authorization model for multimedia digital libraries. *Int. J. on Digital Libraries* 4(3) pp.139-155.

Sandia National Laboratories, The Rule Engine for the Java^{TM} Platform. [online] Available at: http://www.jessrules.com/ [Accessed 31 July 2010]

Bornet, OpenCV. 2000 [online] Available at: http://sourceforge.net/projects/opencv/, [Accessed 12 June 2010]

Chen, T.Y. 2008. Knowledge sharing in virtual enterprises via an ontology-based access control approach. *Computers in Industry* 59(5): pp.502-519

Prud'hommeaux, E and Seaborne, A. 2008. SPARQL Query Language for RDF, [online] Available at: http://www.w3.org/TR/rdf-sparql-query/ [Accessed 12 June 2010]