A Fine-Grained Document-based Access Control Model

Liliana Rosero, Jesus Aranda, Michel Riguidel, and Daniel Gidoin

***Abstract*—In this paper is presented a new access control method called *document-based access control* (*DuControl*). *DuControl* extends traditional access control methods in order to manage the document contents; it is done by means of a *rules set* that allows easy administration of the access policies, resources and users.**

***Index Terms*—Access Control, Authorization, Computer Security, Data Processing, Data Security**

I. INTRODUCTION

In information management systems, securing the access to structured documents and their parts is an important issue. This means, to ensure the availability of the data to the authorized users and at the same time to protect the information from intruders (unauthorized users). Classical security models and architectures are unable to meet the security requirements of a system that must provide a fine granularity of document protection. It is due to the recent developments in digital architectures (peer-to-peer distributed architectures, broadband, and multimedia content).

The aim of this paper is twofold. First, it proposes a new theoretical model of a fine-grained access control to documents called *DuControl*. The second aim of this paper is to show software implemented in order to verify the model.

One of the main novelties is the definition of *closed* and *open* authorizations, which provides a more precise way of controlling document access. It allows for an early control of possible conflicts between different kinds of accesses.

Formal definitions of *DuControl* actions and rules are presented in this paper, and the implementation of DuControl is discussed.

Manuscript received March 23, 2011. This work was supported in part by Thales, ThereSIS. Palaiseau, France.

L. Rosero, is a Ph.D. student at Telecom ParisTech. Paris, France. (e-mail: liliana.rosero@polytechnique.edu).

J. Aranda is a professor in the department of Computer Sciences at Universidad del Valle, Cali, Colombia; (e-mail: jesus.aranda@correounivalle.edu.co).

M. Riguidel is a professor in the department of Computer Sciences and Networks at Telecom ParisTech, Paris, France. (e-mail: michel.riguidel@telecom-paristech.fr).

D. Gidoin is the head of ICT Security Research Group of ThereSIS, Thales. Palaiseau, France. (e-mail: Daniel.gidoin@thalesgroup.com)

.

The paper is organized as follows: In Section 2 some related work is discussed. Section 3 shows a motivating example, which traditional access control models cannot support well. Section 4 presents the four types of rules and their properties in the proposed *DuControl* model. In Section 5, we give the implementation of *DuControl*. Section 6 hast the architecture and shows the Confia tool which was developed in order to verify the *DuControl* model. Section 7 concludes this paper.

II. RELATED WORK

Access control in an information system is set of rules determining which users are allowed to read, execute, share, or to modify specific elements in the system. The aim of an access control model is to ensure that only authorized access can take place. The evolution of access control models started with the research of Lampson [1], who introduced the basic ideas of discretionary access control (DAC) as a means of restricting access to objects according to the ownership of a resource. Each user in the system determines who can access the resources in their possession. In 1975, Bell and La Padula proposed the mandatory access control model (MAC) [4] that describes a set of access control rules which use security labels on objects and clearances for users. Security labels range from the most sensitive (e.g."Top Secret"), down to the least sensitive (e.g., "Unclassified" or "Public"). In the 1990s, Role-Based Access Control models [2] [3] [9] were proposed as an alternative to traditional discretionary and mandatory access controls for managing and enforcing security in large-scale enterprise wide systems. The main idea is that permissions are associated with roles and users are assigned to appropriate roles. It ensures that only authorized users are given access to certain data or resources. A role is a semantic construct forming the basis of access control policy. It is essentially a collection of permissions, and all users receive permissions only through the roles to which they are assigned.

Bertino and Ferrari [5] describe a method for securing the distribution of XML documents; they do this by encrypting file contents with different keys, and then distributing the keys selectively. They deal with the situation of multiple documents having the same policies and also of individual documents with unique policies. Their approach to access control is in terms of user characteristics rather than User Ids. Users are provided with credentials upon logging and then they are grouped into credential types.

Although many great contributions have been made to the progress of the field of access control, authorizations in these models are still static authorization decisions based on subjects' permissions on target objects. Once an access to the object is granted, the subject can access it repeatedly. To solve these problems, Park et al proposed the Usage Control (UCON) [8]. The UCON model unifies traditional access control models and temporal access model with the models called $UCON_{ABC}$ [6] which are based on Authorizations, oBligations and Conditions.

Our approach to solving these problems is different from that of the UCON model. With ***DuControl***, we extend traditional access control methods by adding a set of rules based on both privilege level and permitted extent of access within each document. We provide fine-grained access to parts of documents and we control the actions made with each user's privileges. We introduce two new concepts in the area of access control: *closed* and *open* authorizations. We believe that these features meet the requirements of modern access control systems, and that they can be easily applied to real-world applications. Whereas UCON is wide-ranging family of abstract models, DuControl is pragmatic solution ready to be implemented for operational use.

III. MOTIVATING EXAMPLE

This section introduces an example from [7] to motivate the applicability of *DuControl* in real world collaboration that takes place between two European Union (EU) administrative entities called *Europol* and *Eurojust*, and authorities of 27 countries member.

Example 1. Europol and Eurojust have their representatives (Europol National Member and Eurojust National Member) for each country in the European Union. Each country has its national contact points called National Authority for both Europol and Eurojust.

Also each country has its law enforcement authorities; and their employees collaborate whenever there is an occurrence of cross border organized crime in the EU. This kind of collaboration entails a request for Mutual Legal Assistance (MLA). In this scenario, participants collaboratively define and work on a document called European Arrest Warrant (EAW).

The MLA scenario is structured as follows:

1. A Europol National Unit of country A (ENU A) makes a written request of assistance (for a witness protection) to a Eurojust National Member of country A (EJNM A).
2. The EJNM A opens a Temporary Work File in a local Case Management System (CMS).
3. The EJNM A contacts Eurojust National Member of country B (EJNM B) by forwarding the request of assistance.
4. The EJNM B contacts the responsible national authority of country B (NA B). Steps are taken by the responsible NA B to provide the requested assistance.

The MLA scenario depicts a clear scenario of collaborative activities on different parts of an XML document. Figure 1. shows the structure of the EAW document, and the labels for every element and attribute of the scenario.

Some requirements that characterize the scenario are the following:

Distributed Sources, Autonomy, and Access Policy. The different parts of the EAW document are structured according to the local regulations and policies of the authorities of the EU. For example, one country may require knowing the religious belief of a suspect. On the contrary, law in another country may prohibit the disclosure of one's religious belief.

Fine Grained Access. As the crime is cross border, sensitive information is possessed by different law enforcement authorities. One employee on EJMN of country A may need to access a deeply nested element containing useful information that is owned by ENU B. On the contrary, the information accessible for other employees could be limited.

As showed in the next sections, the proposed access control model handles detailed security policies that are able to define accesses to each part of a document. As multiple responsible authorities could originate the parts of the document, the rules of the model have being defined to be capable of expressing the particular policies for each part of the document.

Vulnerability situations and conflicts can arise when defining accesses to different parts of a document where each part can have a different owner. As a document is hierarchically organized, each part can be affected according to its own policies and the policies defined for the whole document or another part subsuming it. *Open* and *Closed* authorizations are defined to solve conflicts and prevent vulnerability situations.

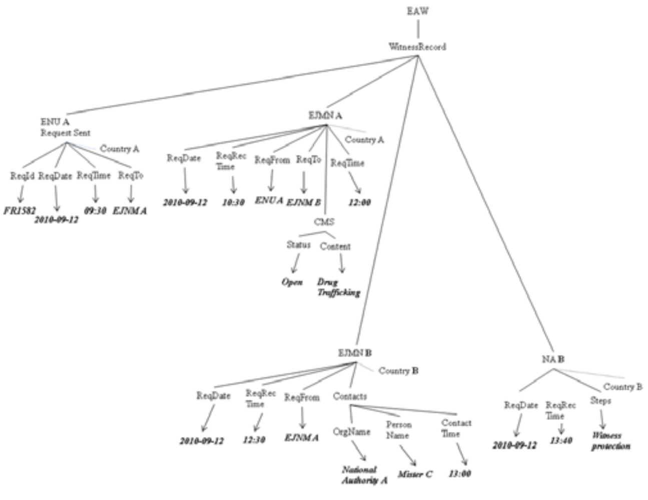


Figure 1: Graph representation of the XML document in Example 1.

IV. DuControl Model

A. Access

Definition 1 (*Access*). An *Access* is a 6-tuple *access(S, Ob, R, Content, Polarity*$_{right}$*, Polarity*$_{content}$*)* where *S* is the user who has the access, *Ob* is the object to be accessed, *r* is the privilege exercised on *Ob*, *Content* means the part of the Object to be accessed, and *Polarity*$_{right}$, *Polarity*$_{content}$ indicate whether or not the access rights or access scope (respectively) may be extended.

Example 2: Let us consider access(Alice, Itinerary.xml, Read, 1, Open, Closed). This indicates that the user Alice can access the document Itinerary.xml up to level 2 (i.e. the attributes and the nodes of the second level below the document root) with privilege Read. The level of access on the document can be potentially be increased (Open) but the scope cannot (Closed).

B. Access Rules

Now, we describe a set of rules suitable for controlling the access on documents. First the 4 types of rules are described, after that we explain and give formal definitions for each rule.

There are four types of rules:

1) Grant Rules:

These permit user to access an object according to some requirements (e.g. conditions, obligations).

2) Derived Rules:

These propagate a user's access rights within the document hierarchy according to the level of protection of each sub-tree, and the type of rights granted (read, write, add nodes).

3) Forced Rules:

These solve any conflict between different access rights that have the same user and object.

4) Non-vulnerability Rules:

These protect the entire content of a document from intrusion as a result of unauthorized access

a) Grant Rule

Rule 1, the grant rule, states that if a user satisfies requirements Tr_i before allowing access, they can obtain the privilege *R* on the part of the object *Ob* indicated by *Content*, on that rule, we have two parameters called Polarity. *Polarity*$_{right}$, *Polarity*$_{contentt}$ represent, respectively, whether or not the privilege R or the part of the object accessed can be extended.

Rule 1.

Access(S, Ob, R, Cont, Polarity$_{right}$*, Polarity*$_{content}$*)* $\leftarrow$
Tr_1 &Tr_2 ...& Tr_n $(0 \leq i \leq n)$

b) Derived Rules

Rule 2, a derived rule, states that if a user has access to an object *Ob* with privilege *R* and with content parameter +, then they also have access to any object Ob_2 that is part of *Ob* (described by the predicate *Ancestor* on the formal definition of the rule) with the same privilege, content parameter and polarities.

Rule 2.

Access(S, Ob2, R, +, Polarity$_{right}$*, Polarity*$_{content}$*)* $\leftarrow$
Ancestor(Ob, Ob$_2$ *)* & *Access(S, Ob, R, +, Polarity*$_{right}$*, Polarity*$_{content}$*)*

Rule 3, a derived rule, states that if a user has access to an object *Ob* with privilege *R* and with content parameter *k,* then they also have access to any object Ob_2 that is an element of the object *Ob* at level *i* (where $i \leq k$) (described by the predicate $Ancestor_i$ on the definition of ***Rule 3***) with the same privilege and polarities but with content parameter *k-i*.

Rule 3.

Access(S, Ob2, R, k-i, Polarity$_{right}$*, Polarity*$_{content}$*)* $\leftarrow$
Ancestor(Ob, Ob$_2$ *)* & *Access(S, Ob, R, k, Polarity*$_{right}$*, Polarity*$_{content}$*)*,
$(0 \leq i \leq k \leq n)$

Rule 4, a derived rule, states that if a user has a privilege *R* on the object *Ob* then also has any privilege R_2 weaker than *R* on the object *Ob*, where the privilege R_2 is extensible (i.e. $\text{Polarity}_{content}$ = Open). The other parameters remain unchanged.

Rule 4.

Access(S, Ob, R$_2$*, Content, Open, Polarity*$_{content}$*)* $\leftarrow$
$R_2 \leq R$ & *Access(S, Ob, R, +, Polarity*$_{right}$*, Polarity*$_{content}$*)*

Rule 5, a derived rule, states that if a user has access to the area of the object *Ob* represented by *Content* then they also has access to any subsumed area *Content*$_2$ where the latter is extensible (i.e. $\text{Polarity}_{content}$ = Open). The other parameters remain unchanged.

Rule 5.

Access(S, Ob, R, Content$_2$*, Polarity*$_{right}$*, Open)* $\leftarrow$ *Content*$_2$≤*Content*
& *Access(S, Ob, R, Content, Polarity*$_{right}$*, Polarity*$_{content}$*)*

Rule 6, a derived rule, states that if a user has access to an object *Ob* that is part of an object Ob_2 then the same user can access Ob_2 at level 0 (i.e. the root of the object) with the same rights and where both polarities are open.

Rule 6.

Access(S, Ob$_2$*, R, 0, Open, Open)* $\leftarrow$ *Ancestor(Ob*$_2$*, Ob)*
& *Access(S, Ob, R, Content, Polarity*$_{right}$*, Polarity*$_{content}$*)*

c) Forced Rule

Rule 7, the forced rule – and the most complex one – is used to resolve the conflict when there are two different accesses with the same user and object. Below we describe this resolution rule that determines how to get a definitive access:

The access rights are decided by considering whether or not the access in conflict with the more restrictive rights has $\text{Polarity}_{\text{right}}$ = Closed or not; if so, the definitive access is the more restrictive one, otherwise it is the more permissive one.

The content depends on whether or not the access in conflict with the more limited content has $\text{Polarity}_{\text{right}}$ = Closed; if so, the content of the definitive access is the more limited one, otherwise it is the less limited one.

The polarity is Open if the polarities of both accesses in conflict are Open, otherwise the polarity is Closed.

Rule 7.

Access(S, Ob, R_3, $Content_3$, $Polarity_{right3}$, $Polarity_{content3}$) <—
Access(S, Ob, R_2, $Content_2$, $Polarity_{right2}$, $Polarity_{content2}$)
& Access(S, Ob, R_1, $Content_1$, $Polarity_{right1}$, $Polarity_{content1}$)

Where

R_3 = Min(R_1, R_2) if $\text{Polarity}_{\text{right2}}$= $\text{Polarity}_{\text{right}}$= Closed,
Max(R_1, R_2) if $\text{Polarity}_{\text{right2}}$= $\text{Polarity}_{\text{right}}$= Open,
Max(R_1, R_2) if $R_i, \geq R_j$ & $i \neq j$ & $\text{Polarity}_{\text{righi}}$= Closed & $\text{Polarity}_{\text{righj}}$= Open,
Max(R_1, R_2) if $R_i, \geq R_j$ & $i \neq j$ & $\text{Polarity}_{\text{righi}}$= Open & $\text{Polarity}_{\text{righj}}$= Closed.

$Content_3$ = Min(Content_1, Content_2)
if $\text{Polarity}_{\text{content2}}$= $\text{Polarity}_{\text{content1}}$= Closed
Max(Content_1, Content_2)
if $\text{Polarity}_{\text{content2}}$= $\text{Polarity}_{\text{content1}}$= Open,
Max(Content_1, Content_2)
if $R_i, \geq R_j$ & $i \neq j$
& $\text{Polarity}_{\text{contenti}}$= Closed
& $\text{Polarity}_{\text{contentj}}$=Open,
Max(Content_1, Content_2)
if $R_i, \geq R_j$ & $i \neq j$ & $\text{Polarity}_{\text{contenti}}$= Open
& $\text{Polarity}_{\text{contentj}}$= Closed

$Polarity_{right3}$ = Closed
if $\text{Polarity}_{\text{right1}}$ = Closed
or $\text{Polarity}_{\text{right2}}$ = Closed,
Open otherwise

$Polarity_{content3}$ = Closed
if $\text{Polarity}_{\text{content1}}$ = Closed
or $\text{Polarity}_{\text{content2}}$ = Closed
Open otherwise

d) Non-vulnerability Rule

Rule 8, the non-vulnerability rule, is used to protect the content of an object. It states that if a user has been granted access up to a certain extent of the object with *$Polarity_{content}$ = Closed* then it is forbidden to allow the same user to access document contents beyond that extent.

Rule 8.

Access(S, Ob, R_2, $Content_3$, $Polarity_{right2}$, Closed) ←
$Ancestor_j$(Ob, Ob_2)
& Access(S, Ob, R_2, k, $Polarity_{right2}$, Closed) & k < j
& Access(S, Ob_2, R_3, $Content_1$, $Polarity_{right1}$, $Polarity_{content1}$)

Rules 1 to 6 are conservative, i.e. the accesses in the rule head are kept. In Rules 7 and 8 – the resolution rule and non-vulnerability rule, respectively – the accesses in the head rule are replaced with the definitive accesses, which are calculated by each rule.

C. Some Properties

Now, we describe some properties that are held for the set of rules.

Definition 2 (*Conflict*). A conflict occurs when there are at least two different accesses: Access(S, Ob, R_1, Content_1, $\text{Polarity}_{\text{right1}}$, $\text{Polarity}_{\text{content1}}$) and Access(S, Ob, R_2, Content_2, $\text{Polarity}_{\text{right2}}$, $\text{Polarity}_{\text{content2}}$) where the users and the objects are the same.

Definition 3 (Conflict-free rule). A rule is said to be conflict-free if after applying the rule there is no conflict.

Theorem 1 (Conflict-free property). The set of rules described above generates a conflict-free set of accesses.

Proof.

By construction of these rules, Rule 7 can be used to resolve the conflicts generated by the other rules. If there are a set of accesses in conflict Access_1, Access_2, ... Access_n, one can apply repeatedly Rule 7 as follows: Rule 7(Rule 7(Rule 7(Access_1, Access_2), Access_3), ... Access_n). Eventually only one access will remain.

Definition 4 (Vulnerability). It is said that a system is vulnerable if it is possible to give a user certain access rights to an object when the user is either not allowed to access that object or they are not allowed to access that object with those rights.

Theorem 2 (Non-vulnerability). The set of rules described above is not vulnerable.

Proof.

By construction of these rules, Rule 7 and 8 explicitly resolve any possible vulnerability concerning rights or content, respectively.

D. Applying the model to the motivating example

A situation is modelled in order to show how the model here proposed can be useful in controlling access to documents. The situation comes from the scenario described in Section III.

In the situation it is contemplated that one country may require knowing the religious belief of a *suspect*. On the contrary, law in another country may prohibit the disclosure of one's religious belief. As the religious belief can be seen as an object it is possible to define the following accesses:

Access_1: Access(Subject, $Ob_{personal\ information\ suspect}$, Right, 1, $Polarity_{right}$, Closed)

Acccess_ 2: Access(Subject, $Ob_{personal\ information\ suspect}$, Right, 1, $Polarity_{right}$, Open)

Access_1 prohibits explicitly the disclosure of the personal information of the suspect; meanwhile *Access_2* could allow disclosing that information up to depth 1 if needed. Let us consider that $Ob_{personal\ information\ suspect}$ is subsumed in the Object $Ob_{suspect}$. and there is an access as follows:

Access_3: *Access(Subject, $Ob_{suspect}$, Right, +, $Polarity_{right}$, Open)*

Access_3 is saying that the whole information of the suspect could be disclosed, clearly it means that the religious belief, being part of suspect personal information, could also be disclosed. It can be derived by Rule 2:

Access_4: Access(Subject, $Ob_{personal\ information\ suspect}$, Right, +, $Polarity_{right}$, Open)

Access_1 and *Access_4* could be in conflict as *Access_1* is saying that the parts of the personal information can be accessible, whereas *Access_4* means the opposite. This kind of conflict leaves open possible vulnerabilities. ***Rule 7*** solves this conflict by stating the following resolution access from Access 1 and Access 4:

Access_5: Access(Subject, $Ob_{personal\ information\ suspect}$, Right, 1, $Polarity_{right}$, Closed)

Access_5 states that the personal information of the suspect cannot be disclosed.

However, if *Rule_7* is applied from *Access_2* and *Access_4* the resolution access would be the following:
Access_6: Access(Subject, $Ob_{personal\ information\ suspect}$, Right, +, $Polarity_{right}$, Open)

Access_6 means that the whole personal information of the suspect is disclosed.

V. Implementation of DuControl

A. Algorithms

In implementing an access control model, the administration of access requests is fundamental. All accesses in DuControl are derived from the grant, derived, forced and non-vulnerability rules.

A set of accesses is called a *document-based access base* (DAB). A DAB contains accesses which are not in conflict with each other and without introducing vulnerability.

Below we outline an algorithm for calculating a DAB from a set of rules R:

A LGORITHM 1. *Document-based Access Base* (DAB)

```
Let be R= {$r_1$, $r_2$, $r_3$, $r_4$, $r_5$, $r_6$, $r_7$, $r_8$} a set of rules;
DAB is initialized to be empty;

CalculateDAB(R) {
  newAccesses = applyRules(r1, r2, r3, r4, r5, r6, DAB)
   while (newAccesses $\neq$ {})
         DAB = DAB union   newAccesses
     newRules = applyRules(r1, r2, r3, r4, r5, r6, DAB)

  DAB = resolveConflicts(DAB, r7)
  DAB = non-vulnerability(DAB, r8)
}
```

Notice that in the implementation is imperative that the DAB is calculated by using firstly Rules 1 to 6 – the conservative rules – and then the filtering rules Rules 7 and 8. In this way we can guarantee the identification of all the possible conflicts and inconsistencies w.r.t. access to the document, and thus eliminate them.

After the set of valid accesses, DAB, is determined from the set of rules, the next step is to judge whether or not the access requests are approved. So every access request is checked against the DAB to determine whether the access is authorized.

Definition 5 (Access Request). An access request is a 3-tuple (S, Ob, right) where S is the user who requires the access, Ob is the object to be accessed and right is the privilege exercised on object Ob.

A LGORITHM 2. *Access Request*

```
Let R be a set of rules;
Let DAB be a set that after calling CalculateDAB will contain
accesses which are not in conflict with each other and without
introducing vulnerability.


Access_control(R, right) {

  DAB = CalculateDAB(R)
  acc = retrieveAccess(DAB, right)
  If (acc)
    retrieveViewObject{acc.content, acc.Ob}
  else
    error{"Illegal access"}
}
```

The first step is to calculate the DAB, then to check if the request matches some access in DAB. If so, then the part of the object visible to the user is calculated from the Object per se (acc.Ob) and the content (acc.content).

VI. Architecture

In this section, are explained the main components in the architecture of our software tool called Confia. Figure 2 shows the functional architecture, there is a module called ***policies store***, which is the policies repository; the module called ***Rule Set***, implements the rules presented in Section IV; the policies store is invisible to ordinary users but is visible to the administrator.

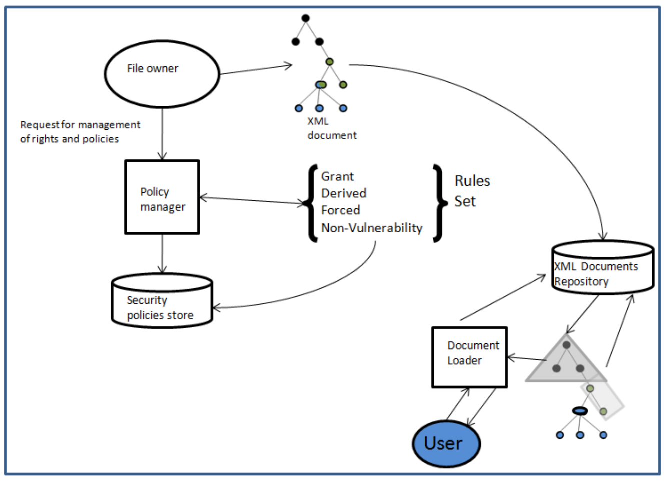


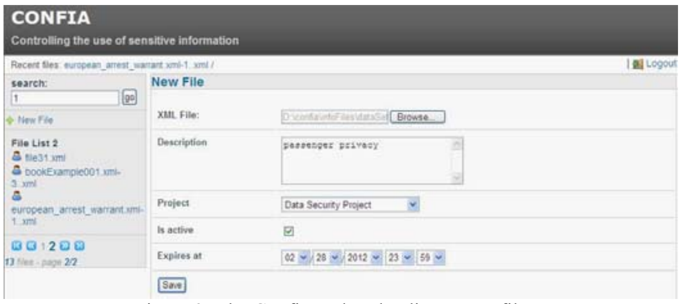
Figure 2. The architecture of Confia tool.The XML document is uploaded to the system.

- The document owner can decide to apply security attributes to the tree nodes, and to assign access rights to other users.
- The policy manager communicates with the rule set in order to solve any conflicts between the new accesses and those in the policies store.
- The Non-vulnerability rules are applied in order to protect the entire content of the document from intrusion.
- Derived rules are applied to propagate access rights and to ensure that the policies store is stable.
- When a user asks for a resource (i.e. a file, a subtree or a node), the *grant rule* permits him to access that object only if in the policies store there is a policy that explicitly says that the user is allowed to use that object.
- When the user does modifications into the file and/or leaves the system, the updated document is sent to the XML documents repository.

A. Software description

The Confia tool was developed in Java in order to verify the *DuControl* model, and more generally to have a self-contained environment in which to develop and test XML document access control algorithms. It is a multi-user document repository with a web-based interface for uploading, browsing, and modifying files. File contents are displayed as an expandable tree of XML nodes, to which security attributes can be assigned. The ability of users to view whole or partial documents in the repository is a function of these attributes and their access level. Both the document owner and the system administrator may assign security attributes to documents, while the system administrator alone may assign access levels to users. Internally the system calculates the DAB for every user using the algorithm 1. And the access are decided based on algorithm 2; then the user can access only the parts of the document to which they are granted.

When a user uploads a new file (see Figure 3), they must assign it to a project and decide whether or not it is active, i.e. visible to the other users. If so, the user must specify a date and time at which it expires and is no longer active.



Recent files: european_arrest_warrant.xml-1..xml /

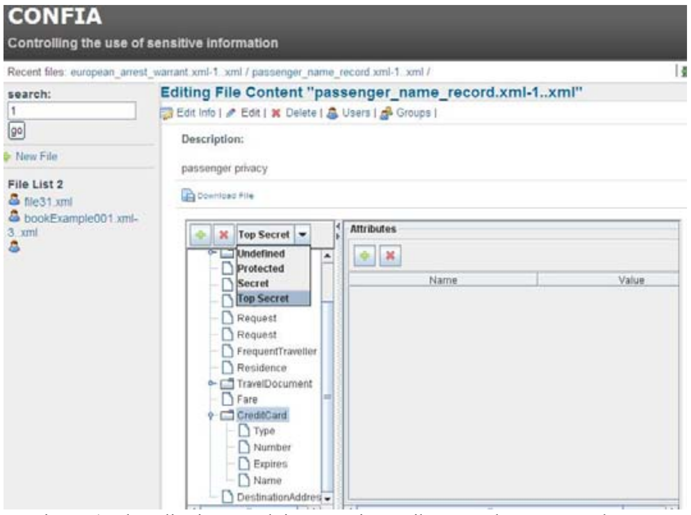
File List 2
file31.xml
bookExample001.xml-3..xml
european_arrest_warrant.xml-1..xml
1 2
13 files - page 2/2

New File

XML File: D:\confia\infoFiles\dataSet
Description: passenger privacy
Project: Data Security Project
Is active: ✓
Expires at: 02 / 28 / 2012 23 : 59

Figure 3: The Confia tool, uploading a new file

Once the user has uploaded a new file, they can use the *edit* view to examine the document tree and optionally apply security attributes to one or more nodes. The list of possible security attributes is defined by the system administrator; Figure 4 shows an example where the user may choose from the attributes *Undefined*, *Protected*, *Secret*, and *Top Secret.*


CONFIA
Controlling the use of sensitive information
search:
go
New File
Edit Info | Edit | Delete | Users | Groups |
Download File
Attributes
Name
Value

Recent files: european_arrest_warrant.xml-1..xml / passenger_name_record.xml-1..xml /

File List 2
file31.xml
bookExample001.xml-3..xml

Editing File Content "passenger_name_record.xml-1..xml"

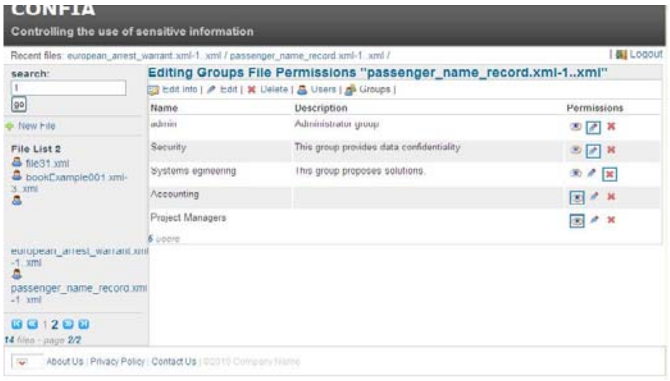
Description:

passenger privacy

Top Secret
Undefined
Protected
Secret
Top Secret
Request
Request
FrequentTraveller
Residence
TravelDocument
Fare
CreditCard
Type
Number
Expires
Name
DestinationAddres

Figure 4: The edit view, applying security attributes to document nodes.

In addition, the *groups* view (Figure 5) can be used to specify the overall access rights (read only, read / write, none) to the document for each group of users.


CONFIA
Controlling the use of sensitive information

Recent files: european_arrest_warrant.xml-1..xml / passenger_name_record.xml-1..xml / | Logout

search: 1 go

New File

File List 2
file31.xml
bookExample001.xml-3..xml
european_arrest_warrant.xml-1..xml
passenger_name_record.xml-1..xml

1 2
14 files - page 2/2

Editing Groups File Permissions "passenger_name_record.xml-1..xml"

Edit info | Edit | Delete | Users | Groups |

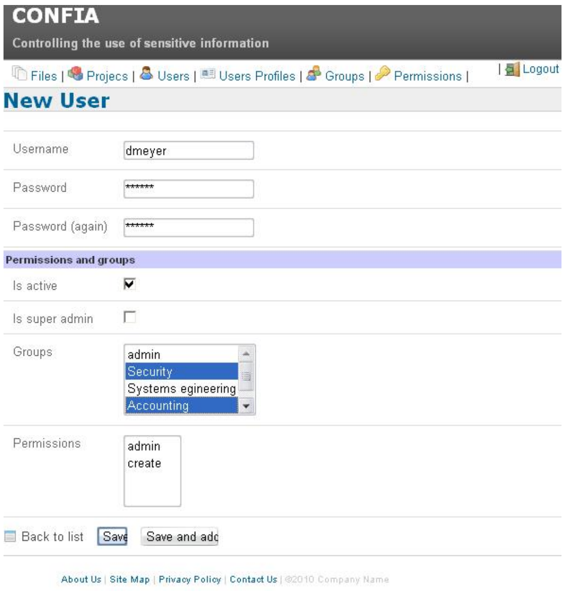
Name	Description	Permissions
admin	Administrator group	
Security	This group provides data confidentiality	
Systems egineering	This group proposes solutions.	
Accounting		
Project Managers		

6 users

About Us | Privacy Policy | Contact Us | ©2010 Company Name

Figure 5: The groups view

Each user's access level, and the list of groups to which they belong, are specified by the system administrator when their account is created (see Figures 6 and 7).


CONFIA
Controlling the use of sensitive information

Files | Projecs | Users | Users Profiles | Groups | Permissions | | Logout

New User

Username	dmeyer
Password	******
Password (again)	******

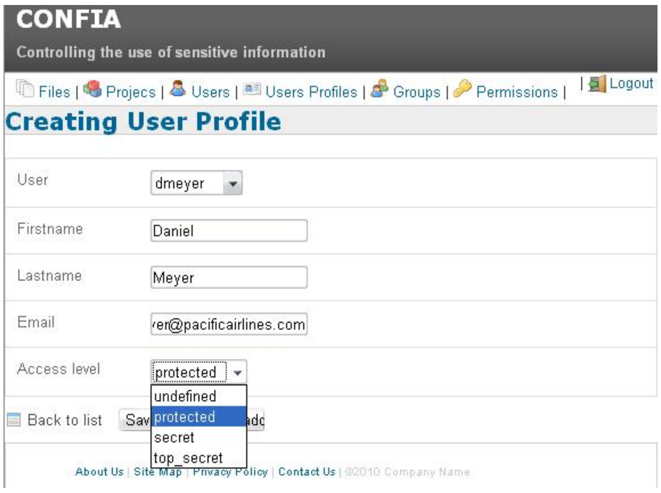
Permissions and groups

Is active	✓
Is super admin	
Groups	admin, Security, Systems egineering, Accounting
Permissions	admin, create

Back to list | Save | Save and add

About Us | Site Map | Privacy Policy | Contact Us | ©2010 Company Name

Figure 6. Adding a user


CONFIA
Controlling the use of sensitive information

Files | Projecs | Users | Users Profiles | Groups | Permissions | | Logout

Creating User Profile

User	dmeyer
Firstname	Daniel
Lastname	Meyer
Email	er@pacificairlines.com
Access level	protected

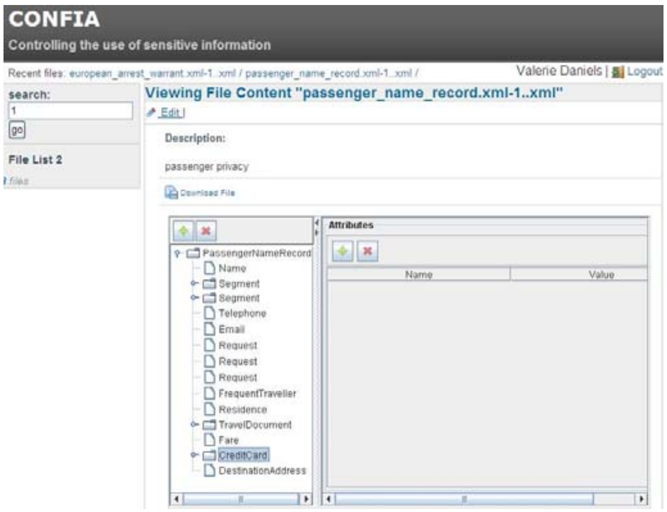
undefined
protected
secret
top_secret

Back to list | Save | Save and add

About Us | Site Map | Privacy Policy | Contact Us | ©2010 Company Name

Figure 7. Creating a user profile and setting the level of security clearance

Once a file has been uploaded and annotated, other users that have been granted read or modify access to it are able to view the parts of the document tree corresponding to their access level. If one or more document nodes have been annotated with a higher security classification than a user's access rights, those nodes will not appear in the version of the document presented to that user (see Figures 8 and 9).


CONFIA
Controlling the use of sensitive information

Recent files: european_arrest_warrant.xml-1..xml / passenger_name_record.xml-1..xml / Valerie Daniels | Logout

search: 1 go

File List 2

Viewing File Content "passenger_name_record.xml-1..xml"

Edit |

Description:

passenger privacy

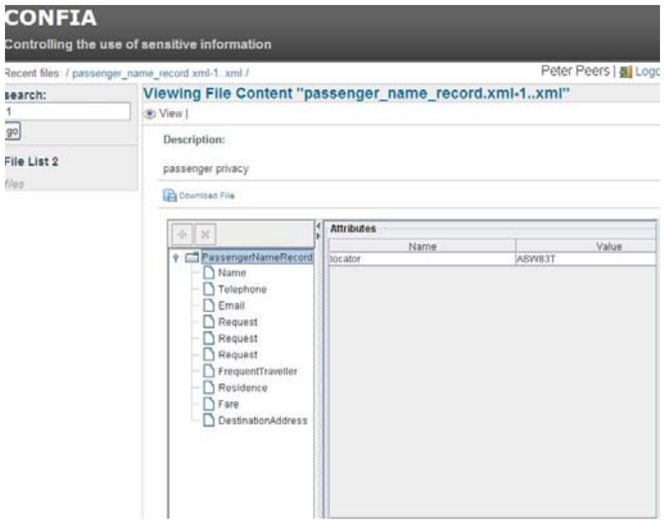
Download File

PassengerNameRecord
- Name
- Segment
- Segment
- Telephone
- Email
- Request
- Request
- Request
- FrequentTraveller
- Residence
- TravelDocument
- Fare
- CreditCard
- DestinationAddress

Attributes

Name	Value

Figure 8: File view of a user with edition rights


CONFIA
Controlling the use of sensitive information

Recent files: / passenger_name_record.xml-1..xml / Peter Peers | Logout

search: 1 go

File List 2

Viewing File Content "passenger_name_record.xml-1..xml"

View |

Description:

passenger privacy

Download File

PassengerNameRecord
- Name
- Telephone
- Email
- Request
- Request
- Request
- FrequentTraveller
- Residence
- Fare
- DestinationAddress

Attributes

Name	Value
locator	ASW83T

Figure 9: File view of a user with read rights

Internally the system calculates the access rules applicable for every user using the implementation of the rule set described in Section 3. Then the user can access only the parts of the document to which they are granted. Any conflict is solved by calling recursively to rule 5, the Forced Rule.Conclusions

To meet new requirements in modern information systems, we present a new access control model: the document-based access control (DuControl) model. This model proposes open and closed accesses, allowing for a higher degree of control over the access to documents. In particular, the closed accesses impose a barrier that prevents the user from subsequently obtaining more privileges or gaining access to more of the document's content. A non-vulnerability rule is proposed, to prevent intrusion from possible unauthorized access.

DuControl has a vast range of applications in modern information systems, since its access control technology can be applied to all objects that have internal structure, such as films, military maps, and financial data. The protection of XML documents is just one such application.

REFERENCES

[1] Lampson Butler. Protection. *5th Princeton Symposium on Information Science and Systems, 1971. Reprinted in ACM Operating Systems Review, 8(1), 18-24*, 1974.

[2] Ferraiolo D. and Kuhn. D. Role-based access controls. *Proceedings of the 15th NIST-NSA National Computer Security Conference, Baltimore, Maryland*, 1992.

[3] Ferraiolo D., Cugini J.A., and Kuhn D. Role-based access control (rbac): Features and motivations. *11th Annual Computer Security Applications Proceedings*, 1995.

[4] Bell E. and La Padula L. Secure computer systems: Unified exposition and multics interpretation. T*echnical Report MTR-297, The Mitre Corp. Bedfor, Massachusetts.*, 1976.

[5] Bertino E. and Ferrari E. Secure and selective dissemination of xml documents. *ACM Transactions on Information and System Security*, ISSN 1094-9224:290–331, 2002.

[6] Park J. and Sandhu R. The ucon_abc usage control model. *ACM Transactions on Information and System Security*, 7:128–174, 2004.

[7] Rahaman M. and Roudier Y. A document-based dynamic workflow system. Institut Eurecom, France, 2009.

[8] Sandhu R. and Park. J. Usage control: A vision for next generation access control. *Lecture Notes in Computer Science. Springer Berlin / Heidelberg*, 2776/2003:17–31, 2003.

[9] R.S. Sandhu, E.J. Coyne, H.L. Feinstein, and C.E. Youman. Role based access control models. *IEEE Computer*, 29, 2:38–47, 1996.

Liliana Rosero is currently a PhD student in the Computer Science and Networks department at Télécom ParisTech in Paris, where she works under the supervision of Professor Michel Riguidel, and in the ThereSIS lab at Thales. She was awarded her master's degree in Complex Industrial Systems Engineering in 2007 by l'École Polytechnique de Paris, and obtained her engineering degree at the Universidad del Valle in Colombia. Her research interests include security in information systems, access control models and privacy protection.

Professor Jesús Aranda earned a PhD in Computer Science from l'École Polytechnique de Paris (2010), a PhD in Engineering from la Universidad del Valle-Colombia (2010), and a B.Sc. in Computer Science from la Universidad del Valle (2001). His main interests are concurrency theory and constraint satisfaction problems. Currently, he is a professor in the Department of Computer Science at la Universidad del Valle -Colombia. Contact him at jesus.aranda@correounivalle.edu.co

Professor Michel Riguidel was the Chair Department of Computer Science and Networks, at Télécom ParisTech, (2001-2010) where he lectures in security and advanced networks. His research is oriented towards security of Information Systems and Networks, Architecture of Communication Systems. He is the Head of the Multidisciplinary Thematic Network in security at CNRS (National Scientific Research Center). He belongs to the Executive Board of RNRT (National Network in Telecommunication Research). In the IST Integrated Projects of FP6 and FP7, he is Key Researcher of the SECOQC Integrated Project (Development of a Global Network for Secure Communication based on Quantum Cryptography), responsible of the Network Architecture. He has several patents in security (firewall, watermarking and protecting CD ROM).

Daniel Gidoin Graduated in 1973 from Engineer national school (ENSTEA) and in 1974 from a DEA of University Paris-SUD. He also graduated from a Master of security at College de l'Ecole Polytechnique (EN-STA) in 1993. He contributed to the e-Gov scenario conception, requirements specification and security and dependability architecture development and evaluation. He is the head of ICT Security Research Group of ThereSIS. He is also involved in NESSI European Technology Platform where he is co-leader of the Nessi TSPR Working Group (Trust Security Privacy and Resilience). Daniel has been involved in a number of European Security projects (e.g. Serenity, COMPAS, Nexof-RA) and PFC (System@tic). Daniel is also active participant to cross-ETP working group on Future Internet (focusing on Trust, Security and Privacy).