Access Control Infrastructure for Digital Objects[*]

Javier López, Antonio Maña, Ernesto Pimentel, José M. Troya, Mariemma I. Yagüe

Computer Science Department
University of Málaga
email: {jlm, amg, ernesto, troya, yague}@lcc.uma.es

Abstract. Distributed systems usually contain objects with heterogeneous security requirements that pose important challenges on the underlying security mechanisms and especially in access control systems. Access control in distributed systems often relies on centralized security administration. Existing solutions for distributed access control do not provide the flexibility and manageability required. This paper presents the XML-based Secure Content Distribution (XSCD) infrastructure is based on the production of self-protected software objects that convey contents (software or data) and can be distributed without further security measures because they embed the access control enforcement mechanism. It also provides means for integrating Privilege Management Infrastructures (PMIs). Semantic information is used in the dynamic instantiation and semantic validation of policies. XSCD is scalable, facilitates the administration of the access control system, guarantees the secure distribution of the contents, enables semantic integration and interoperability of heterogeneous sources, solves the "originator retained control" issue and allows activities (such as payment) to be bound to the access to objects.

Keywords: Distributed systems security, secure content distribution, XML metadata, Privilege Management Infrastructure.

1 Introduction

The "digital object" concept has proven to be a valuable approach for different applications in distributed environments. Digital objects can be classified with respect to their contents. On the one hand, data objects encapsulate several logically related pieces of data along with some administrative information in a package intended to provide a uniform access. This is the case in systems for information commerce, digital libraries or eBooks. On the other hand, software objects encapsulate several services/operations. Software objects are found in object oriented middleware, web services or grid computing.

The security problems associated to digital objects also depend on the type of content encapsulated in the object. The protection for data objects is usually related to *Digital Rights Management* issues. In particular, access control, use-control, payment

[*] Work partially supported by Spanish *Ministerio de Ciencia y Tecnología* under project TIC2002-04500-C02-02

and copyright enforcement are relevant problems. The main problem for software objects is use-control, while access control (to operations), copy-protection, function analysis and runtime protection are also important issues. In both cases, the ability to retain control over the objects after they are accessed, known as “originator retained control”, is a desirable security property.

Distributed systems usually contain objects with heterogeneous security requirements. However, some of the new scenarios where distributed systems are emerging share some common problems. The most remarkable ones are the following. Firstly, it is usual that objects are accessed by previously unknown users. Therefore, subscription-based schemes are not appropriate in this case. Secondly, the execution of copyright agreements, payment or other activities must be bound to the access to the objects. Finally, the originator or owner of the object must retain control over it regardless of its physical location and even after it is accessed by users. Other requirements are: (i) that a high degree of flexibility is required because of the heterogeneous nature of the objects, (ii) that being able to change the access control parameters dynamically and transparently is also essential and, (iii) due to the large amount of objects, it is important to be able to establish access conditions in an automatic way based on information about objects.

Paradoxically, access control in distributed systems often relies on centralized security administration. Centralized control has important disadvantages: (a) the control point represents a weak spot for security attacks and fault tolerance, (b) it does not facilitate the deployment of owner retained control mechanisms, (c) it reduces system performance because it introduces a bottleneck for request handling, and (d) it usually enforces homogeneous access control schemes that do not fit naturally in heterogeneous user groups and organizations. On the other hand, systems for distributed security administration still have open problems. Solutions proposed so far do not provide the flexibility and manageability required. A system for distributed access control has been proposed based on the concept of mobile policies [1] to solve some of the limitations of *Role Based Access Control* (RBAC) schemes [2]. This improvement is limited by the requirement of executing the access control policies in trusted computers. Furthermore, when access to an object is granted, this object is sent to the client computer where it has no protection. Finally, because object and policy are compiled in a package, a change in the policy requires that the object-policy package is recompiled and distributed to all trusted servers.

This paper presents the *XML-based Secure Content Distribution* (XSCD, pronounced “exceed”) infrastructure that provides distributed access control and enforcement and secure content distribution. We address the integration of a separate *Privilege Management Infrastructure* (PMI) by defining mechanisms for the semantic description of its components. We introduce *Semantic Policy Language* (SPL), an XML-based policy definition language designed to specify policies in a simple way, to be evaluated by processors with limited capabilities such as smart cards and to facilitate semantic policy validation processes. SPL policies are modular and can be composed without ambiguity. We also address the problem of the association of policies to objects in a flexible and automated way and the combination of policies. To achieve our goals we have extended the concept of mobile policy by allowing their execution in untrusted systems and used XML metadata technologies extensively.

The rest of the paper is organized as follows. Section 2 summarizes some relevant related work. Section 3 presents the building blocks of XSCD. Section 4 describes the infrastructure. Finally, section 5 summarizes the conclusions and presents ongoing and future work.

2 Related work

Regarding access control, several proposals have been introduced for distributed heterogeneous resources from multiple sources [3][4]. Unfortunately, these proposals do not address the specific problems of access control in distributed systems. Traditional access control schemes such as mandatory access control (MAC), discretionary access control (DAC) or RBAC are not practical for scenarios where the users are previously unknown or with a very large number of registered users.

Static grouping of users can suffice in many situations but it is not flexible enough to cope with the requirements of more dynamic systems where the structure of groups can not be anticipated by the security administrators. In these scenarios new resources are frequently incorporated to the system and each resource may need a different group structure and access control policy. Furthermore, the policy for a given resource may change frequently. A different approach is required in order to solve the scalability problems of these systems, facilitate access control management and provide means to express access conditions in a natural and flexible way.

Some systems based on the idea of the self-secured package of information have been proposed for secure content distribution. None of these systems has achieved a representative use because the security of these systems depends heavily on the security of the client software. IBM's *Cryptolope* [5] is one of the most elaborated alternatives. A Cryptolope is a package that includes the protected content and all necessary administrative information. As noted in [6], the fact that the opener component (known as Cryptolope Player) runs on the end user's PC, introduces the possibility to produce software emulators. In addition, an infrastructure of *trusted clearing houses* and online connection with these entities is needed. A similar scheme is Intertrust's *Rights|System platform* [7] where digital content is protected even when it is resold. This platform is designed for high-value digital goods but is actually limited to three data formats. Both schemes are platform dependent (need specific client software), offer a set of closed possibilities for the contents, and have no integrated payment scheme. Moreover, they are designed for high-value digital goods, but are not adequate for low-value transactions and occasional business relations.

In the context of policy specification, several XML based languages such as XACL, XrML, ODRL, ebXML, SAML or XACML have been developed for access control, digital rights management, authentication and authorization. Many similarities and interesting features can be found among these languages. Nevertheless, they do not support some relevant properties such as policy parameterisation and composition. Moreover, many features provided by those languages are not necessary in our application scenarios [8].

Two related proposals are the Author-X system [9] and the FASTER project [10], which propose similar systems specific for access control to XML documents. Both systems define hierarchic access control schemes based on the structure of the document. But the structuring of XML documents does not necessarily match the security requirements of the nodes. As a consequence, for the general case the number of different authorizations (positive and negative) that have to be defined grows up rapidly. Author–X policy language uses DTDs, while FASTER uses XML-Schema [11]. Scalability is very limited in both Author-X and FASTER systems. The FASTER system is described as completely server-side and Author-X is essentially centralized, although a distributed approach is proposed based on a set of XML federated sources relying on a central 'master source'. The design based on this central 'master source' has negative consequences on its scalability.

The content protection of Author-X is founded on the concept of "passive" secure container requiring a different key for each possible view of the document. This introduces important disadvantages related to the administration of the access control system and the security [12]. FASTER does not support any content protection mechanism, except the creation of the appropriate user view on the server. FASTER access control is based on user groups and physical locations defining a subject hierarchy. This scheme does not work well for scenarios where heterogeneous contents are frequent and the structure of groups can not be anticipated by the administrators. Author-X is based on credentials that are issued by the access control administrator. Therefore, in practice, each credential will be useful only for a single source, limiting interoperability. A consequence of this approach is that users are obliged to subscribe to sources before they can access their contents

3 XSCD Overview

The main motivation has been to design an access control scheme that is scalable to a large number of previously unknown users and solves the originator retained control issue. Starting from a design based on attribute certificates and taking into account the characterization of different scenarios as well as the analysis of previous proposals. The following are the main goals for our secure distribution infrastructure:

- *Scalability.* The centralized approach adopted in current access control systems introduces many drawbacks in terms of efficiency, manageability and security. The access control system must be designed to suit scenarios where the number of users, attributes and policies are very large. Therefore, distributed access control enforcement is essential. Moreover, distributed management of the security policies is also important.
- *Originator-retained-control.* This issue deals with enabling the originator to retain control over the protected object, not over the contents. The latter, called 'full use-control', is a digital rights issue that is out of the scope of this work.
- *Distributed access control management.* Administrators should be able to manage their resources regardless of the resource location. Attribute certificates issuers must be independent of the application and the system must support the secure interoperation with those entities.
- *Interoperability*. The integration of different heterogeneous object sources is hindered by traditional access control systems because each source defines a

specific access control scheme. The integration of an external PMI represents a step towards the solution of this problem.

- *Distributed access control enforcement.* Access control enforcement mechanisms must be distributed in order to avoid bottlenecks in request processing.
- *Ease of management.* The distributed approach must not introduce complexity of management. Tools to help security administrators should be provided.

The separation of the access control and authorization functions (credential issuance or attribute certification in our case) is universally accepted as a secure and scalable approach. XSCD is based on the integration of an external PMI supported by semantic information about the certification entities. We describe now the basic building blocks of the XSCD infrastructure.

3.1 Building Blocks

Privilege Management Infrastructure. Traditionally, the study of authorization issues has focused on access control. However, when considering Internet applications, authorization adopts a wider meaning, including group membership, role identification (collection of permissions or access rights, and aliases for the user's identity), limits on the value of transactions, access time for operations, security clearances, time limits, etc. Attribute certificates provide, for those applications, the means to carry authorization information, which in this way becomes "mobile".

The mobility feature of attributes is not a new issue. In fact, extensions of identity certificates as specified in ITU-T 1997 recommendation [13] tried to address this problem. However the use of the corresponding extension, *subjectDirectoryAttributes*, does not make entity attributes independent from identity. To be more precise, when using that solution, the change of privileges indirectly force a costly revocation of the identity related information. Besides, that solution does not solve delegation and impersonation issues, which are especially relevant in many of actual applications. The ITU-T 2000 recommendation [14] provides a more suitable solution because it clearly defines a framework where identity and attribute certificates, although related, can be independently managed. That recommendation defines new types of authorities, *Attribute Authorities* (AA), for the assignment of privileges. It also defines the *Source of Authority* (SOA) as the ultimate authority to assign a set of privileges. Additionally, the ITU-T framework provides a foundation to build a PMI that contain a multiplicity of AAs, SOAs and final users.

Usually, each SOA issues certificates for a small number of semantically related attributes. With this approach security administrators do not have control over some elements of the access control system. Consequently, a mechanism to establish the trust between these administrators and the PMI is required. We have addressed this problem using semantic information about the certifications issued by each SOA to assist the security administrators in the creation and semantic validation of access control policies.

Software Protection. In order to provide secure content distribution and originator control, the access control system must provide means to protect the contents not only while in transit through the network but also when they arrive to the destination. The problem with systems based on passive secure information containers is that users must install specific client software to access the protected data. This software controls access to the data and enforces the appropriate actions (e.g. payment) before access is granted. Because the client software is not protected, it becomes the weak spot in terms of security. Opposed to these proposals we use "active" containers (software instead of data) in order to avoid some problems of the latter. XSCD uses protected mobile software elements named *Protected Content Objects* (PCO) to convey the contents and force the user to fulfil the applicable policy before access is granted. By "protected software" we mean that it is neither possible to discover nor to alter the function that the software performs and it is also impossible to impersonate the software. In our solution, this is achieved using a variant of the SmartProt system [15]. SmartProt partitions the software into functions that are executed by two collaborating processors. One of those processors must be a trusted computing device that enforces the correct execution of the functions and avoids that these functions are identified or reverse engineered. We are currently using smart cards for this purpose although other alternatives are possible.

Some specific sections of the unprotected software are translated by SmartProt into functionally equivalent sections of card-specific code. The translation process also identifies the dependencies between these protected sections, reorganizes the code and introduces fake code and data to confuse the attacker. These sections are then encrypted with a unique, randomly produced key using a symmetric cryptosystem. This key will be later included in a license (for a specific smart card) that is required to be able to run the software. Each license is encrypted using the public key of the client smart card. The last step substitutes the original code sections by calls to a function that transmits the respective equivalent protected sections, including code and data, to the card. Some additional support functions are also included. Therefore, the protected sections of the software do not reside in the cards; instead, during the execution of the software, these sections are transmitted dynamically as necessary to the card where they are decrypted using the installed license and executed. When finished, the card may send back some results. Some other partial results will be kept in the card in order to obtain a better protection against function analysis and other attacks. As each piece of software has its own key, we can manage them individually, which is not possible in other software protection proposals where the protected sections of all applications share the same key. For XSCD we integrate the access control policy and the license in a structure that we call Mobile Policy (MP).

Metadata. Most of times, metadata (information about data) are designed to support people or programs in locating and retrieving information resources. XML metadata technologies such as XML-Schema, RDF and RDF Schema [16] provide the foundation for the description of semantic information in our proposal. Metadata are applied at different levels in XSCD. On one hand, access control policies benefit from metadata for its creation and semantic and contextual validation. Likewise, digital objects have metadata associated that are used for the dynamic policy assignment and parameter instantiation. Additionally, metadata are used at the mobile policies creation level, for the specification and acquisition of certification rules. On the other hand, metadata is an essential tool for the integration of the external PMI.

Authorization language. Although other XML-based languages have been developed for access control and authorization, there are specific requirements found in our infrastructure that make their use difficult or inadequate. The main reason is that they do not support some relevant properties such as policy parameterisation and composition. On the other hand, policies must be processed inside smart cards with limited storage and processing capabilities. Therefore, the specification language must be simple enough to be translated into a more compact form in order to be processed by the smart cards. For these reasons, we have developed a specific XML-Schema based language, *Semantic Policy Language* (SPL), in order to specify the access control policies. The keys to the high flexibility of SPL are the extensive use of metadata, the modular composition of policies that separates declaration of each policy component and the parameterisation of the policies.

The SPL system uses several components to define policies. *SPL Policies* specify the conditions that must be satisfied to gain access. Each *Policy Applicability Specification* (*PAS*) links some policies to a series of resources. Finally, *Secured Resource Representations* (*SRRs*) and other contextual metadata are used to provide semantic information that is used in policies and PAS. Additionally, *Source of Authorization Descriptions* (*SOADs*) provide a semantic description of the PMI that is essential for the policy validation process. Fig. 1 shows examples of some of these components.

SPL policies are described following an XML-Schema template where we can declare access rules stating the set of certificates that must be presented for granting access. Optionally, we can declare a set of actions to be performed before access is granted. Examples of these actions are `Notify_To`, `Payment` and `Online_Permission`. This is known as provisional authorization. *Import* clauses can also be included to substitute some of the previous components of the policy and to allow the modular composition of policies based on the X-Path standard. Additionally, dynamic instantiation is enabled by the possibility to define parameters in the policies. The instantiation of parameter references is stated in the PAS. Metadata (SRRs and contextual information) is used for parameter instantiation.

The *Policy Applicability Specification* (*PAS*) provides an expressive way to relate policies to resources, either explicitly or based on metadata. PAS documents include declarations of the applicable policies, the target objects and, optionally the instantiation of the parameters of the policy. The object declaration includes the object location, the operations affected (defaulting to all operations) and some optional conditions. In this way, the PAS comprise all necessary information to relate policies to objects, and to instantiate policies. The PAS in Fig. 1 relates all objects in ‘http://www.lcc.uma.es/Research/VFwkProgramme’ of type ‘report’ to the policy defined in ‘VFrameworkProgram.xml’.

The *Secured Resource Representation* (*SRR*) is a simple and powerful mechanism to describe properties about resources. Properties described in SRRs are used to instantiate policies and PAS, and to locate the applicable policies. The SRR in Fig. 1 declares that object ‘http://www.lcc.uma.es/Research/VfwkProgramme/WP3.pdf’ is a ‘report’ and belongs to the project with ID ‘IST_2001-32446’.

```
<?xml version="1.0" encoding="UTF-8"?>
<spl:policy xmlns:spl="http://www.uma.es/ICICS"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.uma.es/ICICS Policy.xsd">
   <spl:parameter>Partner</spl:parameter>
   <spl:parameter>Project</spl:parameter>
   <spl:access_Rules>
      <spl:access_Rule>
         <spl:attribute_Set>
            <spl:attribute>
               <spl:attribute_Name>Member</spl:attribute_Name>
               <spl:attribute_Value>*Partner[@name]</spl:attribute_Value>
               <spl:SOA_ID>*Partner[@SOA]</spl:SOA_ID>
            </spl:attribute>
            <spl:attribute>
               <spl:attribute_Name>Project</spl:attribute_Name>
               <spl:attribute_Value>*Project</spl:attribute_Value>
               <spl:SOA_ID>*Partner[@SOA]</spl:SOA_ID>
            </spl:attribute>
         </spl:attribute_Set>
         <spl:attribute_Set>
            <spl:attribute>
               <spl:attribute_Name>Role</spl:attribute_Name>
               <spl:attribute_Value>Commisary</spl:attribute_Value>
               <spl:SOA_ID>EU_SOA</spl:SOA_ID>
            </spl:attribute>
            <spl:attribute>
               <spl:attribute_Name>Supervisor_Of</spl:attribute_Name>
               <spl:attribute_Value>*Project</spl:attribute_Value>
               <spl:SOA_ID>EU_SOA</spl:SOA_ID>
            </spl:attribute>
         </spl:attribute_Set>
         <spl:attribute_Set>
            <spl:attribute>
               <spl:attribute_Name>External_Reviewer</spl:attribute_Name>
               <spl:attribute_Value>*Project</spl:attribute_Value>
               <spl:SOA_ID>EU_SOA</spl:SOA_ID>
            </spl:attribute>
         </spl:attribute_Set>
      </spl:access_Rule>
   </spl:access_Rules>
</spl:policy>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<spl:PAS xmlns:spl="http://www.uma.es/ICICS"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.uma.es/ICICS pas.xsd">
<spl:policy>VFrameworkProgram.xml</spl:policy>
   <spl:object>
      http://www.lcc.uma.es/Research/VfwkProgramme
   </spl:object>
   <conditions>
      <condition>
         <property_Name>object_Type</property_Name>
         <predicate>equals</predicate>
         <property_Value>report</property_Value>
      </condition>
   </conditions>
   <spl:instantation>
      <spl:formal_Parameter>Project</spl:formal_Parameter>
      <spl:actual_Parameter path="Project_ID">
         <!—empty because is instantiated from the SRR -->
      </spl:actual_Parameter>
   </spl:instantation>
   <spl:instantation>
      <spl:formal_Parameter>Partner</spl:formal_Parameter>
      <spl:actual_Parameter path="//Members/Partner">
  http://www.lcc.uma.es/Research/VFwkProg.xml
      </spl:actual_Parameter>
   </spl:instantation>
</spl:PAS>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<spl:SRR xmlns:spl="http://www.lcc.uma.es/ICICS"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.lcc.uma.es/ICICS SRR.xsd"
resource="http://www.lcc.uma.es/Research/VFwkProgramme/WP3.pdf">
   <spl:property>
      <spl:property_Name>Object_Type</spl:property_Name>
      <spl:property_Value>report</spl:property_Value>
   </spl:property>
   <spl:property>
      <spl:property_Name>Project_ID</spl:property_Name>
      <spl:property_Value>IST_2001-32446</spl:property_Value>
   </spl:property>
</spl:SRR>
```

Fig. 1. Example Policy, its corresponding PAS and the SRR for a report

The *Source Of Authorization Description* (*SOAD*) documents are digitally signed [17] RDF instances expressing the different attributes certified by each SOA, including their names, descriptions and relations. SOADs convey information that is essential for the semantic validation of the policies such as metadata about the different attributes certified by the SOA and its certification procedures. The set of SOADs represents the semantic description of the PMI. Full integration of the PMI can be achieved transparently for the rest of the system based on this description.

4 Architecture and operation of the infrastructure

Fig. 2 shows the main components of the XSCD system and their relationship. The first component is called *Policy Assistant*. This component uses the SOADs to produce and validate SPL Policies and PAS. The second component is the *SmartProt* protection system. This component transforms unprotected content objects in the server into *PCOs* generating also their corresponding *Licenses*. A PCO is a software object that encapsulates and protects the original object and enforces the access control mechanism. PCOs can be freely distributed to untrusted servers. It can be noticed that policies are not included in the PCO. The third component, called *Mobile*

Policy Generator, attends requests from end users producing *MP*s dynamically. The *Object Metadata* database, containing SRRs, is used to determine the set of applicable policies for the corresponding PCO.

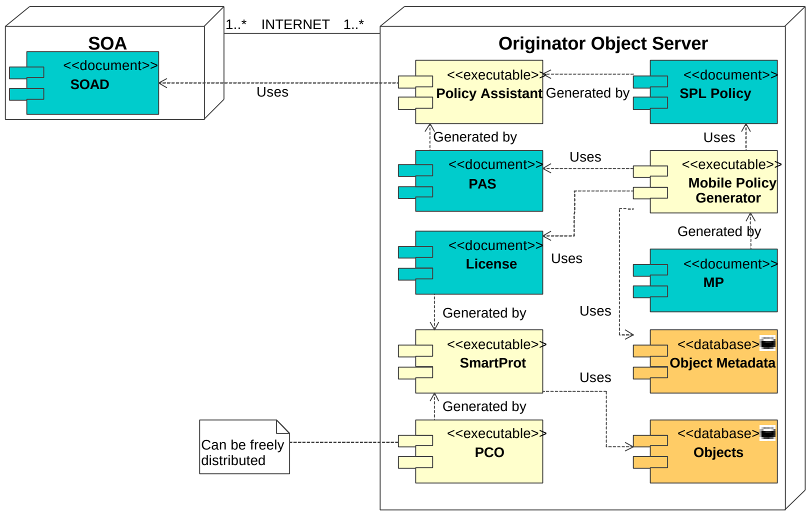

Fig. 2. Overview of the XSCD Infrastructure

Fig. 3 shows the system operation. When the client requests some object from a server it receives the PCO containing it, which runs in the client computer. Before the protected sections of its code can be executed the PCO has to retrieve the corresponding MP (which includes the license that allows the decryption and execution of those protected sections). To do this it sends a request containing the certificate of the public key of the smart card. In case the server from where it was retrieved is the originator of the PCO, it produces the MP for that PCO. Otherwise the server just forwards this request to the originator. Once received, the MP is installed and used to run the protected sections of the PCO.

Policy Specification and Validation. The creation and maintenance of access control policies is a difficult and error prone activity. The Policy Assistant component (which includes the Policy Editor and Semantic Policy Validator) is designed to help security administrators to specify those policies and validate them to find errors. For this purpose, the Policy Assistant provides the administrators with information about the attribute certificates that can be included in the policies, their sources and relation. This information is gathered from SOADs.

The Policy Assistant includes components for the automated validation of policies at different levels. SPL policies are validated syntactically using XML-Schema. Semantic validation is made possible by the use of a specific Semantic Policy Validator that uses the DOM API to parse the document validating it. Finally, policies can be validated taking into account the context where they will be applied.

For instance, consider the case of a research network among several institutions that participate in a common research project. Each participant establishes the access control parameters over the contents they share. Additionally, membership is certified by each institution. Fig. 1 showed an example policy granting access to members assigned to this project of any participant institution, to the commissioner of the project and to the external reviewers assigned to it by the European Union. The institution parameter is instantiated from contextual metadata about the project. On the other hand, the project parameter is instantiated from the SRR corresponding to the object to be accessed. A parameter can represent a complex XML element, as is the case of the institution parameter.

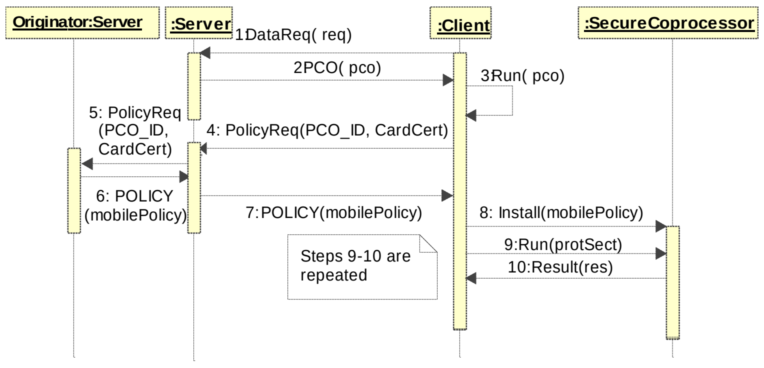

Fig. 3. A scenario for information access

The *Mobile Policy Generator* analyses the semantic metadata available for the target resource contained in the SRR along with other contextual metadata, finds the appropriate PAS and retrieves the necessary SOADs. Using this information, the *Mobile Policy Generator* is able to find the applicable policies. All applicable policies are then analysed and instantiated. Finally, all policies are combined and translated to produce the MP. For the semantic and contextual validation of policies the *Policy Assistant* performs the same process. In this case, after the combination of policies the *Policy Assistant* analyses the resulting policy and run some test cases in order to check its consistency. SOADs play a key role in this semantic validation because they provide the information needed to infer all the possible situations where access is granted, enabling the detection of semantically incomplete or incorrect policies.

Protected Content Object Generation and Access. The SmartProt system is used to protect software applications. In our prototype system, the protected application (PCO) is a Java applet responsible for the transport of the contents. In other environments PCOs will be implemented accordingly (e.g., CORBA objects, a proxy for web services, etc.). Consequently, the PCO includes the (encrypted) contents to be accessed, the access control enforcement mechanism and a cryptographic link to the MP. PCOs can be distributed and copied freely.

The PCO generation process is independent of the customer secure coprocessor (smart card) and will be performed just once for each piece of mobile software. The first step of the PCO generation consists in the production of a Java applet containing the original object. Then, SmartProt is used to protect this applet. The key generated by SmartProt will be used afterwards to produce the MP.

Mobile Policy Generation. In order to allow that originators of the contents are able to dynamically change the applicable access control policy regardless of the storage location, policy and PCO must be separated. In this way, policies are retrieved from the originating server during the execution of the PCO. This allows a high degree of flexibility giving the originator more control over the application of the policies. For efficiency and flexibility, validity constraints in MPs can be used to control the need for an online access to the originator server. Originators can define certain validity constraints for each policy (based on number of accesses, time, etc. depending on the smart card features). Hence, policies can be cached by clients and used directly while they are still valid. The generation of MPs is a reasonably fast process while the generation of PCOs is slower. Furthermore, PCOs are much more stable than policies. Finally, opposed to PCOs, each MP is specific for a smart card.

Following a user request, the new MP is produced linking the combined SPL policy and the PCO. The MP is obtained and loaded in the card as part of the PCO. When the MP is received by the client smart card, it is decrypted, verified and stored inside the card until it expires or the user explicitly decides to extract it. Once the MP is correctly installed in the card the protected sections of the PCO can be executed, which requires the cooperation of the card containing the MP. The Mobile Policy Generator retrieves and combines all applicable policies to produce the MP. The combination of different policies is usually a difficult task that can result in inconsistent or contradictory policies. To deal with this problem the most common solution is to establish a series of general rules to solve the ambiguous cases. But, in practice, each situation is different. Therefore general rules do not produce good results. Our infrastructure enables the administrator to define rules governing the combination of policies. Additionally, the modular approach and the tools for policy creation, composition and validation facilitate the detection and correction of wrong policies. Because MPs must be processed by smart cards, they are specified using SPL and are later translated into a compact format to be included in MPs. The binding between PCO and the corresponding MP is established by cryptographic means.

5 Conclusions and future work

We have presented the XSCD infrastructure for the secure distribution of objects in distributed systems. XSCD extends mobile policies by allowing their execution in untrusted systems and the dynamic and transparent modification of policies. XSCD is based on the SmartProt software protection scheme and the SPL access control scheme. We have introduced mechanisms to seamlessly integrate the external PMI in our infrastructure. The extensive use of XML metadata technologies facilitates the security administration in such environments, and enables important functionalities of the system such as the contextual validation of policies. Furthermore, the combination of policies and the association of policies to objects in a flexible and automated way have been considered. To summarize, XSCD represents a flexible solution for different distributed scenarios and any kind of content, solves the originator-retained-control problem, can be applied regardless of the attribute certification scheme, implements distributed access control management and enforcement mechanisms and allows dynamic modification of policies transparently and efficiently. To the best of our knowledge no other works have been done allowing the semantic validation of policies in distributed environments with separate authorization infrastructures.

A prototype of this system has been implemented for a Digital Library scenario. In such environment, PCOs are implemented using Java applets. *e-gate Cyberflex*^{TM} USB Java smart cards are used as secure coprocessors. The high capacity and the transfer speed of these cards makes possible that the performance of the PCO is very good. A set of techniques, such as temporary authorizations, is used to improve the performance. We are currently working on the application to the CORBA environment. More precisely, we have implemented a Resource Access Decision (RAD) facility based on the XSCD approach.

6 References

1. Fayad, A., Jajodia, S. *Going Beyond MAC and DAC Using Mobile Policies*. In Proceedings of IFIP SEC'01. Kluwer Academic Publishers. 2001.
2. Sandhu, R.S., E.J. Coyne, H.L. Feinstein and Youman, C.E. *Role-Based Access Control Models*. IEEE Computer, 1996. 29(2): pp. 38-47.
3. Thompson, M., et al., *Certificate-based Access Control for Widely Distributed Resources*. Proceedings of the Eighth USENIX Security Symposium. pp. 215-227. 1999.
4. Chadwick, D. W. *An X.509 Role-based Privilege Management Infrastructure*. Business Briefing. Global Infosecurity 2002. http://www.permis.org/
5. Cryptolope Technology Homepage. http://www-3.ibm.com/software/security/cryptolope/
6. Garcia-Molina, H.; Ketchpel, S.; Shivakumar, N. *Safeguarding and Charging for Information on the Internet*. Proceedings of the Intl. Conf. on Data Engineering. 1998.
7. Intertrust Technologies. http://www.intertrust.com/
8. Yagüe, M. *On the suitability of existing access control and DRM languages for mobile policies.* University of Málaga. Department of Computer Science TR. LCC-2002-7. 2002.
9. Bertino, E., Castano, S., Ferrari, E. On Specifying Security Policies for Web Documents with an XML-based Language. In Proceedings of ACM SACMAT'01. 2001.
10. Damiani, E., De Capitani di Vimercati, S., Paraboschi, S., Samarati, P. *A Fine-Grained Access Control System for XML Documents.* In ACM Transactions on Information and System Security (TISSEC), vol. 5, n. 2, May 2002, pp. 169-202.
11. W3C. *XML-Schema*. http://www.w3.org/XML/Schema
12. Bertino, E., Castano, S., Ferrari, E. *Securing XML documents with Author-X*. IEEE Internet Computing, 5(3):21-31, May/June 2001.
13. ITU-T Recommendation *X.509: Information Technology – Open systems interconnection – The Directory: Authentication Framework*, June 1997.
14. ITU-T Recommendation X.509: *Information Technology – Open systems interconnection – The Directory: Public-key and attribute certificate frameworks*, March 2000.
15. Maña, A., Pimentel, E. *An Efficient Software Protection Scheme*. In Proceedings of IFIP SEC'01. Kluwer Academic Publishers. 2001.
16. RDF Vocabulary Description Language 1.0: RDF Schema. W3C Working Draft 30 April 2002. http://www.w3.org/TR/rdf-schema/
17. W3C. *XML-Signature Syntax and Processing.* http://www.w3.org/TR/xmldsig-core/. 2002.