Design and Implementation of Distributed Access Control Infrastructures for Federations of Autonomous Domains

Petros Belsis[1], Stefanos Gritzalis[1], Christos Skourlas[3], and Vassillis Tsoukalas[2]

[1] Department of Information and Communication Systems Engineering University of the Aegean, Karlovassi, Samos, Greece
{pbelsis,sgritz}@aegean.gr
[2] Department of Informatics, Technological Education Institute, Athens, Greece
cskourlas@teiath.gr
[3] Department of Industrial Informatics, Technological Education Institute, Kavala, Greece
vtsouk@teikav.edu.gr

Abstract. Federations of autonomous domains allow resource sharing in a highly dynamic manner, improving organizational response times and facilitating cooperation between different information systems. To accomplish this, it is essential to provide a scalable and flexible mechanism that allows security management and acts at application level independently of operating system or platform. In this paper we present a scalable solution that enables interoperation between different systems participating in a dynamic federation, while it also allows the participating systems to retain their autonomy; we present the software architecture of this distributed access control enforcement mechanism and describe our implementation choices.

1 Introduction

Over the last decades we have experienced a major shift towards the decentralized, distributed computing paradigm. The benefits from the realization of distributed infrastructures are manifold; among else, many challenges have attracted considerable attention in distributed computing, such as: implementation of sophisticated knowledge extraction techniques that enable utilization of assets from different domains; achievement of interoperability between different platforms; performance issues and last but not least, advances in distributed security models. Most of the developed security techniques apply at operating system level; other solutions apply by embedding at each application a customized security mechanism that enables access to authorized users, before logging in. As a consequence, in order to utilize resources in distributed infrastructures, a user has to undergo several independent authorization procedures. This task creates a considerable overhead on each domain, while it also makes more difficult any attempt for Information System's integration. Another parameter that has to be considered is the immediate drop in the degree of user satisfaction, which can prove to be detrimental in business application scenarios.

While decentralization of administrative control requires that all participating domains specify their policies in an interoperable manner, there are a number of challenges related with the ability to transfer the credentials of users in the federated

C. Lambrinoudakis, G. Pernul, A M. Tjoa (Eds.): TrustBus 2007, LNCS 4657, pp. 125–134, 2007.

environment across organizational boundaries [1]. In order to achieve this, there is a requirement to establish interoperable protocols and to provide support for composite policy evaluation.

One additional concern regarding the management of distributed systems is related to heterogeneity, due to the presence of resources of diverse nature. In this paper we describe a distributed infrastructure utilizing XML technologies for access control enforcement. The system's modular components communicate using the Java Remote Method Invocation (RMI) model. The developed prototype is characterized by its scalability potential and its platform independency. The contribution of this paper relies on the following: (i) We present a technique that enables cooperation and resource sharing between multiple autonomous domains; (ii) we present techniques that enable user authentication through a single sign-on procedure for all domains, simplifying thus the authentication procedures to a high degree; (iii) We enable ease of integration of our access control mechanism with existing platforms, while we retain platform and operating system independency.

The remainder of the paper is organized as follows. After the brief introduction, we present the motivation for our research in Section 1; related work and background literature is studied in Section 2. Section 3 analyses the requirements placed on the system design and Section 4 raises and discusses issues related to the system's design and provides example usage scenarios; Section 5 provides concluding remarks and directions for future work.

2 Related Work

The problem of defining access control models for multi-domain environments has recently attracted considerable interest. A number of solutions have been proposed towards this direction. So far, more emphasis has been placed on implementing models, than for creating mechanisms that enable secure interoperation between different domains. In [2] the notion of secure virtual enclaves is being introduced, where domains complying with the Role Based Access Control (RBAC) model share resources. In this work the roles and shared resources are specified in advance and agreed without using technological means, providing thus little support towards the formation of dynamic coalitions.

Bonatti et al [3], propose an algebra for the synthesis of an access control policy out of simpler policies. In their model their language's expressiveness is analyzed with respect to first order logic. They show that their language's formal semantics are equivalent to first order logic formulations. Even though this work provides a tool for preliminary feasibility analysis, the exact implementation details to provide support for coalition formation are missing [3].

Khurana et al [4], define a model for the dynamic management of coalitions based on a Restricted First Order Predicate Logic (RFOPL) RBAC compliant language RCL 2000. In their model, domains take turns in making proposals about the management of shared coalition assets resources. A coalition access control matrix is being formulated keeping records of allowed accesses, while the matrix is being modified during the negotiation process and as intermediate system states are formed. Their work also builds upon a negotiation process that defines membership upon roles with

predefined access permissions instead of negotiating the permissions according to the role classified for every specific user, as defined instead in our work.

Another notable approach that builds upon an XML policy language is the X-GTRBAC framework [1]. This framework provides support for most of the RBAC concepts, such as Separation of Duty Constraints; it also has an integrated mechanism for resolving conflicts emerging from ambiguities or conflicting requirements from the domain specific policies. Unfortunately there are no supporting software tools for this framework so far. Instead of defining a new language, we have decided to utilize evolving standards in access controls and extend them appropriately and develop suitable software tools for multi-domain environments security management. Our work in addition develops a scalable infrastructure built upon independent modules that interoperate using evolving standards in access control.

3 Requirements Analysis

Among the basic requirements when developing distributed access control enforcement infrastructures is the preservation of autonomy. The requirement for decentralization of administrative control in multi-domain environments poses major challenges when specifying the framework for access control policy definition. Decentralization in our framework is achieved by implementing multiple autonomous domains each one of which is responsible for enforcing local access control policies. Each policy enables determination of access privileges for role-access-object pairs, in accordance to the generic Role Based Access Control Model (RBAC) compliant policy definition.

Our framework builds upon the main principles of the XACML [6] policy framework which focuses on enabling distributed management of resources. XACML is an XML based framework for specifying and applying access control for Web-based resources that supports prohibitions, obligations, and resolution of conflicts. Our extended authorization framework has the following strong points:

- It is built using standardized technologies, thus providing support for extensions and enables interoperation between various platforms
- It allows extensions as to support the needs for a variety of environments.
- It allows context-based authorization, by enabling authorization upon examination of domain related predicates (see also section 4).

Our work extends this single-domain authorization framework to provide support for role and privilege assignment for users belonging to remote domains. This is necessary when users from one domain need to be assigned privileges to access data from other federated domains. In order to achieve this interconnection between different domains, several issues need to be taken under consideration:

- Access to data should be regulated by specific generic guidelines, applicable for all the cooperating environments.
- While the data access guidelines should be uniform, enforcement points should be autonomous and have a large degree of freedom in managing their IT infrastructure.

- Dynamic nature of the coalition. The number of units who participate in the cooperating schema is not stable. Units can join or depart at any time, increasing thus the complexity of the overall management.
- Absence of centralized authorization architecture. Security policies can be defined locally without the necessity for central management which would endanger the system's performance by introducing a single point of failure. It would also not be consistent to the distributed nature of the system.
- Transparency to the users. The procedures for retrieving i.e. medical-record details, whether retrieved locally or from a remote domain should be of no difference to the user.

3.1 Generic Access Control Enforcement Model

The basic operational principles of our framework can be divided in two major categories: authentication-related and authorization-specific. Authentication is performed by implementing a mechanism that allows interpretation using SAML [7] compliant assertions for authenticating credentials. The SAML standard provides support for various types of authentication information; a SAML assertion provides information that the requester's credentials match predefined policy requirements. In order to provide an efficient and robust mechanism to verify the user's identity we have utilised X.509 certificates. Thus, the first task for a user is to provide appropriate credentials that will allow him/her identification within the domain he/she belongs to. The SAML assertion issued by the authentication module can be further used by the access control framework in the presence of multiple policies, eliminating the necessity for a user to undergo multiple authentication procedures within the context of the federated environment.

Every solution attempting to enable intra-domain communication should be characterized by its interoperability and scalability features. Our approach in order to enable cooperation between different access policies, builds upon a policy mapping process, which enables roles from one domain to be mapped to another domain [1][9]. In a multi-domain environment, a requester usually originates from a different domain than the one that the requested resource belongs to. As we already stated in the previous paragraphs, a basic requirement is related with the credential management in the federated environment in such a manner that a single sign-on (SSO) mechanism is provided [12]. By integrating in our authentication mechanism SSO capabilities through signed SAML statements, different domains in the federated environment identify authorization decisions already issued by other domains. In addition, our framework provides support for context-enabled authorization and authentication; this is achieved by incorporating context related environmental attributes in role definitions (for example the domain where a user belongs, such as medical.administration.gov). In cases where a request does not originate from the same domain with the PDP, the PDP communicates with the coalition registry which stores information about the available mappings for the requester's role. Each PDP contains information about in-mappings consisting information about roles from remote domains associated with roles to its own jurisdiction and out-mappings for roles in other domains that its policy is associated with. Our approach thus results in a distributed implementation of the coalition registry, which only stores information on a domain-pair basis.

Table 1. Xpath based role mapping between roles in two domains

DOMAIN A	DOMAIN B
Minister/GenSecretaryB/SectorB2Manager	**Minister/GenSecretary/SectorBDirector**

Typically if we consider that the policy is encoded in XML compatible form, the coalition registry contains information about role equivalences between different role hierarchies, which can be encoded by means of XPath expressions [8]. XPath aims at addressing parts of XML documents. It represents location of data in an XML document correctly and efficiently, which makes it a suitable language for both XML query and access control [11]. An example mapping based on XPath is presented in Table 1. This provides an example of a mapping codification example, where the XPath expressions identify role equivalences between different role hierarchies. Therefore we define paths that allow the mapping of roles between different role schemata. Notice that due to the expressiveness of XPath, one can represent more complex role mappings in a very compact way, by grouping together equivalent roles in one XPath expression, without having to write separate rules for each role. The applicability of such a solution is apparent in case of organizations which operate under a common framework (example medical organizations, ministries in e-Government environments, etc).

We enable role mapping to be performed on single-direction basis i.e. a role in one organization could acquire the permissions of another role on the target domain, without the opposite being necessary valid. The next section discusses in detail our proposed approach and we underline the design decisions we undertook in respect to the system design issues raised in this section.

Upon authentication of the requester, the authorization framework works as follows: The administrator edits the policy in appropriate format and makes it available to the PDP. Each request is directed to the Policy Enforcement Point (PEP) which constructs a XACML request message and directs it to the Policy Decision Point (PDP). The PDP proceeds by loading the policy from the policy repository and evaluating the request according to the loaded policy. Accordingly the response is formulated in an XACML response message and is directed to the PEP which finally enforces the decision, authorizing or rejecting the request.

4 System Architecture and Implementation

The distributed policy authorization module is realized by means of object-oriented software architecture, using Java. The system design can be represented using UML class diagrams. Figure 2 depicts a UML based representation of the software architecture meta-model, which extends the single-domain XACML's generic model by introducing the multi-domain management classes.

The main classes of the model include the following: Rule, Policy and PolicySet. The Policy class manages those policies which refer to shared target objects. A target refers to a set of resources under request (Objects requested), the subject (requestor's role) and the action intended to be performed over the shared objects.

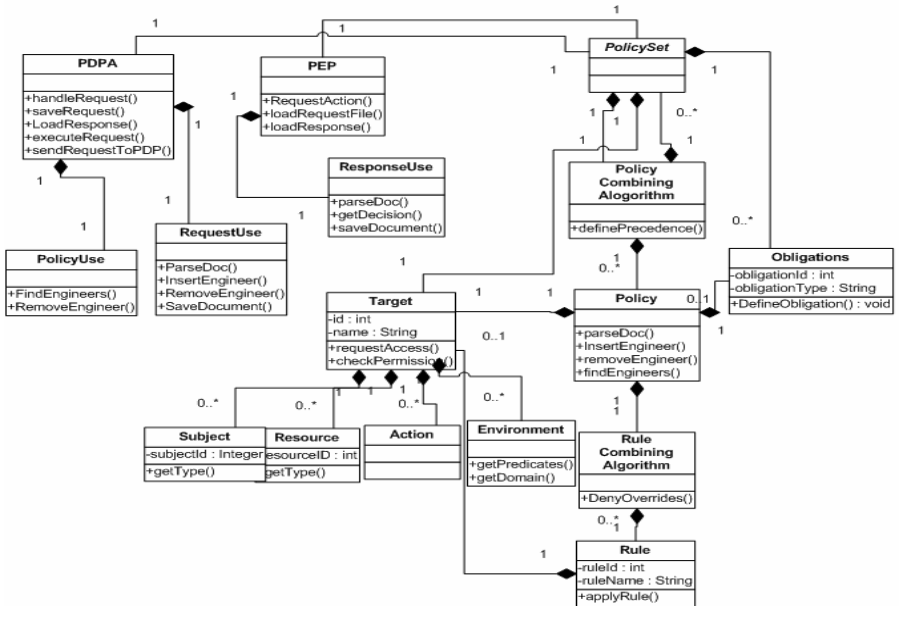

Fig. 1. The distributed access control infrastructure software architecture design in UML notation

The effect of a Rule indicates the result of a logical (i.e. true or false) evaluation of the rule. The allowed actions we have provisioned for are "Permit" and "Deny". A policy <Target> element specifies the subsets resources, actions and environment to which the policy applies. Obligation policies may be supported but their existence is not deemed as necessary, considering our requirements. Obligation policies are likely to be defined by administrators and their characteristic is that there may be less strict controls on modifying an obligation policy. For example, a negative obligation policy may act as a restraining guideline in cases where it is not practical or feasible to issue a negative authorization policy. Policy interoperation is ruled by a Policy combining algorithm, implemented by an appropriate class, responsible for resolving conflicts and ambiguities; depending on the criticality of shared resources, a deny overrides mechanism specifies the priority of access denial criterion in case of a conflict. Subject and Resource classes enable including constraint determination and manipulation in the role-specification schema; for example temporal constraints (determination of activating and deactivating times for a session) or environmental constraints that facilitate role management and enable defining a set of actions for a group of users characterized by common attributes. The distributed PEP and PDP which enable interoperation in a federated environment have been implemented by means of appropriate classes.

The PEP handles authorization enforcement and is responsible for formulating the request for a resource in a XACML compliant message and subsequently forwarding it to the PDP. Furthermore, the PDP except from reasoning over a specific access request provides through its interface the ability to edit and load available policies

Table 2a (left). An Excerpt from an XACML request. The requester's attribute is highlighted, as well as the requested resource. **Table 2b (right)** XACML response message.

```
<Request>
<Subject>  <Attribute >
   <AttributeValue>secretary@nsf.gov</AttributeValue>
</Attribute></Subject>
     <Resource><Attribute><AttributeValue>
file://record/ResearcherlRecords/PeterDoe
</AttributeValue></Attribute></Resource>
    <Action><Attribute><AttributeValue>read
</AttributeValue></Attribute></Action>
</Request>
```

```
<Response>
        <Result>
<Decision>NotApplicable
</Decision>
        </Result>
</Response>
```

from the domain's policy repository. Our PDP's interface allows loading policies from the policy repository and editing them invocating the PolicyUse class. In a similar manner, the PEP constructs the XACML compatible request (Table 2a) and also extracts the response from the XACML response (Table 2b) by invoking the ResponseUse class. All the main modules of the developed prototype represented in UML notation are represented in Figure 2.

4.1 System Usage Scenario - Implementation Details

When a request for a resource appears, it is directed towards the PEP of the domain that contains the requested objects. The request includes the requested object, the subject (requester) and the action (permission) over that resource. Imagine the following scenario: a doctor who works as a general practitioner in two different hospitals while located in hospital B, wants to access some files that he/she has created in hospital A. Since there is a request for files to a remote domain, the authorization process works as follows: the authentication server issues a signed credential which will be also recognized by the corresponding module in hospital B; thus domain B's authentication module is invoked, evaluating the provided by domain A's SAML assertions, allowing a single sign-on procedure for all the participating domains in the coalition. Accordingly domain A's PEP identifies the address of all the cooperating PDP's and forwards the request to them. Each PDP maintains records of

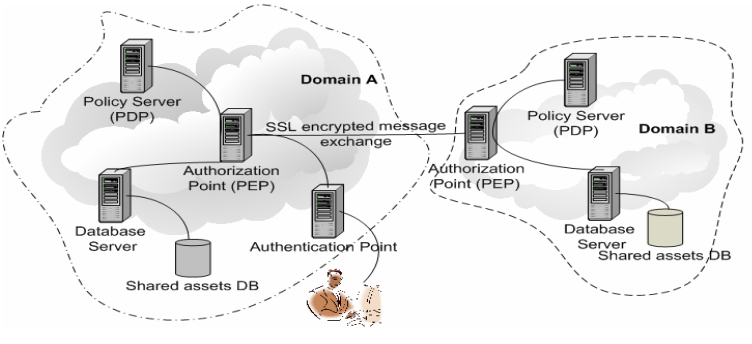

Fig. 3. Multi-domain access control enforcement

the role equivalences from other domains in its coalition registry; thus hospital's B PDP will identify the doctor as one of the roles that should be authorized to access hospital B resources. The invocation also of the context handler integrated in the authentication module and using XACML's context enabled role definition, allows easily authentication evaluating domain specific attributes for a role (such as the domain that the request originates from; for example we authenticate all users that originate from a specific domain like: medical.admin.gov). These attributes can be easily included in the generic XML-based role definition schema. The issued credential along with the request is directed to the domain's PEP which upon receiving an access request, formulates a XACML compliant message indicating the requester, the object to be accessed and the permission under request and directs it to the PDP.

From a technical perspective, there were several issues to consider: first, the need to provide a means to authenticate all users with a single sign-on mechanism; second, the necessity to provide a technique to allow efficiently a mechanism for policy interoperation; third, to provide a technique to reflect easily policy updates, while retaining the security features of the system.

Communication between the different modules from the remote domains is achieved using Java's Remote Method Invocation Model; the reason for selecting this is that it allows to reflect easily updates in both domains authentication-authorization models and to reflect also easily policy updates. Figure 3 gives an overview of the generic architecture of the distributed access control framework. The authenticating module functions in a way that was presented in the beginning of the current section. The authorization framework implemented for our experimental federated environment which consisted of 3 subnets, functions as follows: Each PDP (one for each domain) through the developed for our evaluation purposes prototype interface provides the ability to edit and modify policies. The PEP provides through the interface the ability to formulate requests, and then constructs an appropriate message in XACML format. Through an RMI call the PEP identifies the PDPs of the cooperating domains and directs an XACML message. Accordingly the message is parsed by the parsing module and the original request is identified from the message's payload. Then the policy is loaded from the policy repository and finally the request is evaluated against the available policies. Finally a response message is sent to the PEP which enforces the decision. For the overall system, the potential impact on the PEP's performance is small since there is absence of a centralized PEP; on the contrary, the PEP is implemented in a distributed manner. We have implemented an experimental topology comprising of three different domains with different role hierarchies. Each domain comprises of a different sub-network each one with its own PEP and PDP; these independent modules communicate using Java RMI. For our evaluation scenario we have directed several concurrent requests from each domain towards the other, measuring the capability of our prototype to correctly evaluate those different access requests.

5 Conclusions

In this paper we presented a distributed authorization framework that supports federated autonomous environments. Among its more distinctive features are: (i) its

distributed nature that allows maintenance of autonomy of participating domains; (ii) credential management using single authentication procedures by means of SAML assertions (iii) incorporation of context-related parameters in role specification schemas that effortlessly allow for context-based authentication and authorization. Moreover we can distinguish its scalability support due to the low complexity of the role mapping mechanism; we presented its salient features that support interoperability, since it utilizes XML-based technologies for role specification and role mapping codification. The fact that the coalition registry is also implemented in a distributed manner facilitates its deployment as it demands fewer resources and avoids the existence of a single point of failure as in the case of deploying it in a centralized manner.

We have presented a prototype implementation as part of an ongoing research work; throughout the paper we have presented a generic software architecture using UML notation as well as an operation scenario explaining in detail the role of each module. So far we have tested our prototype using an experimental setting of three different domains and the initial findings are promising. Our framework provides the possibility to apply access controls at application level, providing platform and operating system independency.

Our architecture supports the satisfaction of the requirements recorded in section 3 by: a) providing access to data for users in the federated domain using the presented architecture which applies the policy rules for each domain, while it facilitates autonomy maintenance for all the participating domains; b) by not restricting the number of domains that join or leave the federation since maintenance of coalition related information adds only a small amount of information overhead to the coalition registry; c) there is absence of centralized management. Each domain may cooperate with each other without intermediate management.

One of the main limitations of our approach is the fact that policy mappings have to be agreed by means of bilateral service level agreements between domain administrators; such a limitation though may not always be restrictive, since it is the case for most federated frameworks [10] such as e-Government alliances, or e-healthcare coalitions, to regulate under a common framework; moreover, the legal implications of an inappropriate access to sensitive personal data make automated coalition formation a risky process. In addition, it has been proved that the problem of automated negotiation for more than two policies is intractable [5]. In cases also that there is no direct equivalency in between the different role hierarchies, it is easy to create a new role on one of the hierarchies so as to provide support for a remote domain to access only specific shared resources. In addition the complexity of the approach is by far less than that of creating a global policy out of the component policies of the individual domains and requires less time to integrate a new role equivalency in the coalition registry.

The technical challenges that had to be overcome by the proposed approach are manifold: the architecture of the platform allows ease integration of a large number of domains, supporting thus scalability to a high extent; in addition the policy mappings have been implemented using a low cost technique by both means of technical feasibility and information overhead, something that makes it possible to integrate the platform over wireless infrastructures that lack hardware resources.

Our future work focuses on providing an automated framework to facilitate conflict resolution for the participating domains and on testing the validity of our framework by extensive experimentation for a large number of domains.

References

1. Bhatti, E., Bertino, E., Ghafoor, A.: A Policy framework for Access Management in Federated Information Sharing. In: IFIP Joint Working Conference on Security Management, Integrity, and Internal Control in Information systems, Fairfax USA, December 2005, pp. 95–120. Springer, Heidelberg (2005)
2. Shands, D., Yee, R., Jacobs, J.: Secure Virtual Enclaves: Supporting Coalition Use of Distributed Application Technologies. In: proceedings of the Network and Distributed System Security Symposium, NDSS 2000, San Diego, California, USA (2000)
3. Bonatti, P., De Capitani di Vimercati, S., Samarati, P.: An algebra for composing access control policies. In: ACM Tranactions on Inormation Systems Security (TISSEC) 5, 1, 1–35 (2002)
4. Khurana, H., Gligor, V.D., Linn, J.: Reasoning about Joint Administration of Coalition Resources. In: Proc. of IEEE International Conference on Distributed Computing Systems (ICDCS), Vienna, Austria, July 2002, pp. 429–439. IEEE press, Los Alamitos (2002)
5. Bharadwaj, V., Baras, J.: Towards automated negotiation of access control policies. In: Proc. of the 4th IEEE International workshop on Policies for distributed Systems and Networks (POLICY 03), pp. 77–86. IEEE press, Los Alamitos (2003)
6. Moses et al.: eXtensible Access Control Markup Language specification, v.2 Technical Overview (May 2004) Available: XACML Oasis TC Homepage, http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml
7. Hughes et al.: Technical Overview of the OASIS Security Assertion Markup Language (SAML) V1.1. OASIS (May 2004) http://xml.coverpages.org/saml.html
8. http://www.w3.org/TR/xpath (Accessed May 2006)
9. Belsis, P., Gritzalis, S., Katsikas, S.: A Scalable Security Architecture enabling Coalition Formation between Autonomous Domains. In: Proceedings of the 5th IEEE International Symposium on Signal Processing and Information Technology (ISSPIT'05), Athens, Greece, December 2005, pp. 560–565. IEEE Computer Society Press, Los Alamitos (2005)
10. Ao, X., Minsky, N.H.: Flexible regulation of distributed coalitions. In: Snekkenes, E., Gollmann, D. (eds.) ESORICS 2003. LNCS, vol. 2808, Springer, Heidelberg (2003)
11. Malatras, A., Pavlou, G., Belsis, P., Grtizalis, S., Skourlas, C., Chalaris, I.: Deploying Pervasive Secure Knowledge Management Infrastructures. International Journal of Pervasive Computing and Communications, Troubador Pub 1(4), 265–276
12. Mukkamala, R., Atluri, V., Warner, J.: A Distributed Service Registry for Resource Sharing among Ad-hoc Dynamic Coalitions. In: Proc. of IFIP Joint Working Conference on Security Management, Integrity, and Internal Control in Information Systems, December 2005, Springer, Heidelberg (2005)