A Trust Management package for Policy-Driven Protection & Personalization of Web Content

Juri L. De Coi, Daniel Olmedilla, Sergej Zerr
L3S Research Center & University of Hannover
Hannover, Germany
Email: {decoi,olmedilla,zerr}@L3S.de

Piero A. Bonatti, Luigi Sauro
Università di Napoli Federico II
Napoli, Italy
Email: {bonatti,sauro}@na.infn.it

I. Introduction

Open distributed environments like the World Wide Web offer easy sharing of information, but provide few options for the protection of sensitive information and other sensitive resources. Typically, this protection is based on the assumption that a requester is already known by the server (e.g., by means of previous registration and user/password authentication mechanisms). This way, the server is able to map the identity of the requester into a permissions table in order to grant or deny access to a resource.

Nowadays, due to the success of the WWW and therefore to the big amount of potential users a server might have, maintaining a table of authorizations based on identities is no longer desirable. Specifically, the Web provides an environment where parties may make connections and interact without being previously known to each other. In many cases, before any meaningful interaction starts, a certain level of trust must be established from scratch through an exchange of information between the two parties. However, the more a personal information is sensitive the more it cannot be candidly disclosed to an unknown party, thus trust establishment should proceed by means of bilateral steps, i.e. it should be negotiated.

In trust negotiation [1], [2], [3], [4], [5], [6], [7] a client and a server iteratively request (possibly verifiable) information (e.g., credentials) in order to satisfy each other's policies and be able to perform advanced authorization decisions. Examples of such pieces of exchanged information might be whether an entity is an European citizen, a student in a German university or a member of a seal program such as BBBOnline or TRUSTe. Policies used in trust negotiation provide a flexible and expressive way of specifying access control requirements. Although trust negotiation provides a promising approach for access control on open environments, no flexible integration on the Web has been provided to date.

Furthermore, many of the protected resources are not static, but rather generated dynamically, and sometimes the content of a dynamically generated web page might depend on the security level of the requester. Currently these scenarios are implemented directly on the script languages used to specify how dynamic web pages are to be built. This typically means that the access control decisions that can be performed are either simple and inflexible, or rather costs for its development and maintenance increase. Moreover it is commonly accepted that access control and application logic should be kept separate, as witnessed by the design of policy standards such as XACML and the WS-* suite.

This paper/demo presents an advanced approach to access control on the Web. It presents an easy deployable package that exploits emerging trust negotiation approaches [2], [4], [6] by integrating them in a Web scenario. In such a scenario advance decisions can be made based on expressive conditions, including credentials exchanged among entities in order to establish enough trust to be granted access to a resource, while preserving the privacy of information released [6]. In addition, policies can be used in scripting languages such as JSP in order to personalize dynamically generated content, based on locally stored information or requester information obtained through negotiations. Furthermore, using policies allows us to make use of many of the results in the area, including policy verification techniques and the use of our automatically generated natural language explanations [8] describing i.e. the requirements to be satisfied before access to a resource is granted or why a previous attempt has been denied. In combination with this paper we provide a live demo[1] and a screencast[2].

II. Web Content Generation and Protection

Bob is a web designer and developer of an organization and is also in charge of the development of the web pages that list all clients of the company. Such a list should contain different information depending on the requester:

- for members of the direction team all data including projects together with such a client and the budget of such projects should be included.
- for any employee of Bob's organization or any of its partners the list should only include the client name and the contact person.
- for any other person visiting the website, only the list of client names should be displayed.

Bob creates a server page which will be used to generate dynamic web pages. In order to decide which information is included in the list, he simply states the condition that must be fulfilled for each attribute of the client in order to be included in the page. This way, Bob specifies "name" to be always included. Then if the condition "member of direction team" is satisfied, the page will include name, project, budget and contact person. Else if the condition "consortium employee" is satisfied, then only contact person is added. The

[1] Available at `http://policy.l3s.uni-hannover.de/`

[2] Accessible at the page `http://www.viddler.com/olmedilla/videos/1/`. We recommend viewing it in full screen.

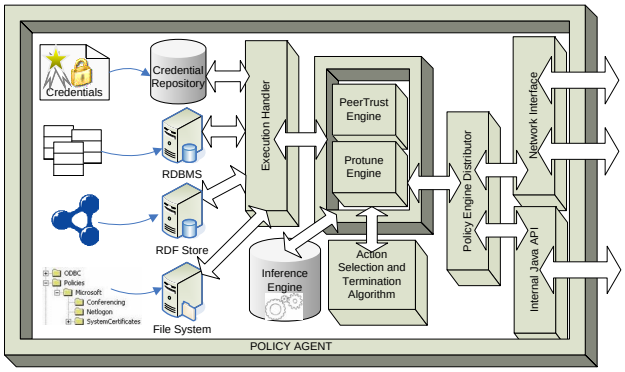

Fig. 1. Policy framework architecture

definition of who is member of the direction team or who works for a consortium partner is not required by Bob to develop the server page, and he is not necessarily allowed to access such information. Instead, a policy manager will specify that information, including that any external requester should certify with a credential that she is an employee of a partner company. The policy manager specified that a credential proving the status of the employee is required in order to satisfy the policy. Since a requester may protect her credentials with policies too, then a negotiation takes place at run-time in order to establish trust by iteratively satisfying each other policies.

III. Implementation

We have built our implementation on top of the PROTUNE policy framework, which provides negotiations and policy reasoning. PROTUNE is entirely developed in Java (architecture depicted in Figure 1) and provides an API that permits its easy integration in applications such as applets, Swing/RCP applications or web servers. It also includes integration of legacy systems, such as relational databases, RDF stores and LDAP servers, and access to local files and extraction of content based on regular expressions, which can be used within the policies.

We have developed a component that is easily deployable in web servers supporting servlet technology (we currently support Apache Tomcat), which adds support for negotiations and policy reasoning. It allows web developers to protect static resources by assigning policies to them. In addition to protection of static content, it also allows web developers to generate parts of dynamic documents based on the satisfaction of policies (possibly involving negotiations). This is done by the developer by adding tags of the form

```
<poljsp:policycondition policyname="protunePolicy">
    <poljsp:iftrue>
        Policy satisfied
    </poljsp:iftrue>
    <poljsp:iffalse>
        Policy not satisfied
    </poljsp:iffalse>
</poljsp:policycondition>
```

Furthermore, we integrated these tags within the Macromedia Dreamweaver framework in order to help web designers to easily

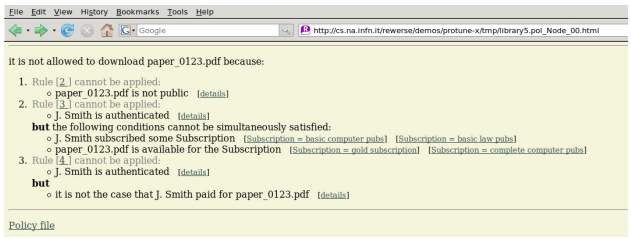

Fig. 2. Example of an HTML based generated explanations

and visually assign policies to their dynamic web pages[3].

On the client side, we intended to provide a solution that would work out of the box, without the user having to install any software. Therefore, we developed an applet (signed by a trusted authority) that is downloaded to the client browser and controls the subsequent HTTP requests generated by the user. We use javascript to capture user links and pass them to the applet. The inverse process is used when the applet receives the answers from the web server in order to display the content, be it the requested page or an access denied page with a link to the HTML based generated explanations. Using an applet allows to provide a browser-independent solution that keeps policies locally, and therefore private. A live demo and a screencast are provided at the aforementioned URLs.

At runtime, when a client makes a request for a resource that requires the satisfaction of a policy (be it static or dynamic), the server evaluates the policy. If the policy is satisfied, the appropriate content is provided to the user. Otherwise, a natural language explanation is automatically generated from the policy [8] (see Figure 2), explaining the reasons why access to the resource is denied. During the process in which the server evaluates the request, if specified by the policy, the server may initiate a negotiation with the client policy agent, therefore starting the iterative exchange of policies and credentials among both of them.

IV. Summary

The World Wide Web offers easy sharing of information, but provide few options for the protection of sensitive information and other sensitive resources. Traditional protection mechanisms rely on the characterization of requesters by identity, which works well in a closed system with a known set of users. Trust negotiation protocols have emerged as a solution for open environments such as the Web. Our paper/demo will present an access control framework for the Web with the following features:

- provide powerful access control mechanisms in open distributed environments based on flexible and expressive declarative policies
- enables (possibly automated) interactions with human and software agents
- personalize the dynamic generation of web content based on information of the requester or other kind of sources and conditions.
- separate access control and application logic, therefore clearly distinguishing between web designers and developers from policy managers and content owners.
- help users understand access controls decisions and results of interactions, as part of the cooperative enforcement of server's policies.

As a side effect, our approach contributes to cost reductions. Factors such as improved readability/maintainability, declarative policy re-use, ontology re-use (say, roles and user categories of common interest), automated documentation (produced by the explanation facility) and automated interactions with human and software agents (through the negotiation agent) are expected to lead to better systems in shorter time with lower costs.

[3]As described in `http://skydev.l3s.uni-hannover.de/gf/project/protune/wiki/admin/?pagename=Integration+with+Dreamweaver`

REFERENCES

[1] W. H. Winsborough, K. E. Seamons, and V. E. Jones, "Automated trust negotiation," DARPA Information Survivability Conference and Exposition. IEEE Press, Jan 2000.

[2] P. A. Bonatti and P. Samarati, "Regulating service access and information release on the web." in *ACM Conference on Computer and Communications Security*, 2000, pp. 134–143.

[3] N. Li, J. C. Mitchell, and W. H. Winsborough, "Design of a role-based trust-management framework." in *IEEE Symposium on Security and Privacy*, 2002, pp. 114–130.

[4] R. Gavriloaie, W. Nejdl, D. Olmedilla, K. E. Seamons, and M. Winslett, "No registration needed: How to use declarative policies and negotiation to access sensitive resources on the semantic web," in *1st European Semantic Web Symposium (ESWS 2004)*, ser. Lecture Notes in Computer Science, vol. 3053. Heraklion, Crete, Greece: Springer, May 2004, pp. 342–356. [Online]. Available: 2004/2004_ESWC_PeerTrust-NoRegistration.pdf

[5] M. Y. Becker and P. Sewell, "Cassandra: Distributed access control policies with tunable expressiveness." in *5th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY 2004)*. Yorktown Heights, NY, USA: IEEE Computer Society, June 2004, pp. 159–168.

[6] P. A. Bonatti and D. Olmedilla, "Driving and monitoring provisional trust negotiation with metapolicies," in *6th IEEE Policies for Distributed Systems and Networks (POLICY 2005)*. Stockholm, Sweden: IEEE Computer Society, June 2005, pp. 14–23. [Online]. Available: 2005/2005_policy_protune.pdf

[7] A. J. Lee, M. Winslett, J. Basney, and V. Welch, "Traust: a trust negotiation-based authorization service for open systems." in *11th ACM Symposium on Access Control Models and Technologies*. Lake Tahoe, California, USA: ACM, June 2006, pp. 39–48.

[8] P. A. Bonatti, D. Olmedilla, and J. Peer, "Advanced policy explanations on the web." in *17th European Conference on Artificial Intelligence (ECAI 2006)*. Riva del Garda, Italy: IOS Press, Aug-Sep 2006, pp. 200–204. [Online]. Available: 2006/2006_ECAI_explanations.pdf