Building a Distributed Semantic-aware Security Architecture

Jan Kolter, Rolf Schillinger, Günther Pernul

Department of Information Systems, University of Regensburg, D-93040 Regensburg
{jan.kolter,rolf.schillinger,guenther.pernul}@wiwi.uni-regensburg.de

Abstract. Enhancing the service-oriented architecture paradigm with semantic components is a new field of research and goal of many ongoing projects. The results lead to more powerful web applications with less development effort and better user support. While some of these advantages are without doubt novel, challenges and opportunities for the security arise. In this paper we introduce a security architecture built in a semantic service-oriented architecture. Focusing on an attribute-based access control approach, we present an access control model that facilitates semantic attribute matching and ontology mapping. Furthermore, our security architecture is capable of distributing the Policy Decision Point (PDP) from the service provider to different locations in the platform, eliminating the need of disclosing privacy-sensitive user attributes to the service provider. With respect to privacy preferences of the user and trust settings of the service provider, our approach allows for dynamically selecting a PDP. With more advanced trusted computing technology in the future it is possible to place the PDP on user side, reaching a maximum level of privacy.

1 Introduction

Over the last years information systems developed from large monolithic systems to dynamic distributed networks. Aside from performance factors, expected economic benefits are the main reasons for this development. More and more companies outsource parts of their production chain, which is only possible with flexible communication infrastructures that provide techniques for distributed processing. Such infrastructures may be based on the emerging service-oriented architecture paradigm [1] that allows the registration and discovery of remote applications which are wrapped into web services.

Along with the development of distributed systems comes the demand for a flexible security infrastructure which suits the complex concepts of the underlying distributed architecture. Apart from privacy, integrity, availability and non-repudiation, access control plays a central role in information systems. Due to the heterogeneous and open character of distributed architectures, access control is not only a major security component; it emerges to a decisive factor in developing a trustworthy architecture.

In this paper our focus is on the development of a distributed security architecture that facilitates flexible semantic access control. Our goal is to employ the existing attribute-based access control (ABAC) and enrich it with semantic components residing in the architecture. The architecture further provides the option of moving the privacy and trust-sensitive Policy Decision Point to a suitable location.

Our work is carried out in the project Access-eGov[1]. The project's goal is to employ Semantic Web and Peer-to-Peer technologies to build a service-oriented e-Government architecture that provides distributed registries and semantic discovery of annotated web services. An integral part of this semantic architecture is a security infrastructure, providing a flexible access control component and protecting citizens' privacy.

The remainder of this paper is organized as follows: In Sect. 2 we discuss the development and existing approaches of ABAC. We continue with introducing the concept of semantic service-oriented architectures in Sect. 3. In Sect. 4, we present a dynamic, semantic-aware security architecture, the main contribution of this paper. Finally, Sect. 5 describes the building blocks and implementation details of our prototype system. Section 6 concludes this paper and gives an outlook on our future work.

2 Attribute-based Access Control

As the attribute-based access control (ABAC) component is the key component of our security architecture, we lay out the development and give an overview of ABAC in this section. Furthermore, we pinpoint existing approaches of ABAC in service-oriented and semantically-enriched settings.

2.1 Access Control

Lopez et al. define access control as "the prevention of unauthorized use of a resource, including the prevention of use of a resource in an unauthorized manner" [2]. Generally, the question of which subject owns permissions to access a set of objects occurs whenever a number of subjects have to use a number of objects to fulfill certain tasks. From the early days of multi-user computing access control emerged to a central security issue and is still a major concern of modern distributed information systems.

A general access control architecture is introduced by Sandhu and Samarati [3]. Main actors of this architecture are a subject wanting to access an object, a database with access policies called the authorization database, and a reference monitor, that either allows or disallows a subject's access to an object, based on data from the authorization database. This architecture is generic enough to serve as basis for every modern access control system and generally follows one

[1] Supported by the European Union, IST Programme, Contract No. FP6-2004-27020

of the three following security models, namely the Discretionary Access Control (DAC) model, the Mandatory Access Control (MAC) model or the Role Based Access Control (RBAC) model [2].

All three of these access control models, however, bear shortcomings that conflict with certain requirements of modern information systems.

2.2 Development of Attribute-based Access Control

As opposed to aforementioned access control models with static mappings, concepts focusing on subjects' and objects' dynamic properties are of importance in large-scale distributed systems. Those attribute-based access control (ABAC) policies gained increasing popularity in the last years.

The basic idea of ABAC is based on the comparison of values of selected subject attributes and object properties (so called subject and object descriptors) [4]. Descriptors are a construct to "group" objects and subjects dynamically, not explicitly by an administrator but implicitly by their attribute or property values. As an example consider that access may depend on the age of a user. In this case, privilege assignments to the user cannot be done statically by a security administrator but have to be dynamically evaluated by the system based on the value of some of the attributes, e.g. "DateOfBirth". As the user gets older, his authorization state changes automatically. Access permissions might even depend on an external attribute, such as "physical location" of a user in a mobile environment.

While ABAC cannot be seen in many production environments, it has been taken up for research several times. Early work by McCollum et al. suggested a move away from discretionary access control and mandatory access control to "user attribute based access control" in the context of access to classified documents [5]. Throughout the 1990s, a number of researchers explained various forms of ABAC.

One widespread approach, the attribute certificate-based access control, encompasses work with practical implementations which are usually very close to efforts related to the X.509 / ITU-T [6] notion of attribute certificates, as presented by Farrell and Housley [7]. Most of the work in this field does not move too far away from a RBAC approach. Certified attributes are mainly used as assertions of the presence of certain roles. The main difference to RBAC is the decentralized management given by the concept of privilege management infrastructures as is for example explained in [2]. It is important to note that those approaches do usually not include the concept of the accessed objects bearing attributes. The interested reader is referred to [8, 9, 10] for a selection of research in this area.

A further reaching approach in the generic attribute-based access control propose the move away from the strong RBAC foundations, also including subject and object attributes in the access decision. Adam et al. showed that digital libraries need an access control which is not based on roles but on "qualifications

and characteristics of users" [11]. The XACML specification [12] is a policy language already supporting both subject and resource attributes. This idea was also taken up in [13] with the ultimate goal of allowing access control decisions without creating rules for every single resource.

3 Semantic Service-oriented Architecture

The security infrastructure presented in Sect. 4 is built in a semantically-enriched service-oriented architecture (SOA). As we employ the semantic components of this architecture, this section points out the weaknesses of plain SOAs and introduces the idea of the Semantic Web, resulting in the Semantic SOA (SSOA) paradigm.

SOAs are considerably flexible with regards to composition of different services. The discovery of services, however, is still a tough task. The service user has to search for relevant services using keywords, if the exact location of a needed service is not known in advance. Only in rare cases the location of the service is known from the start, e.g. if it is cached from previous executions or if it is known at implementation time of a SOA-based application. Consequently, if the location of a service is neither cached nor agreed upon, the architecture has to provide means to find matching services.

This discovery process in SOAs is equivalent to information retrieval in the World Wide Web or other networks, with all its problems and proposed solutions [14]. The current state of the World Wide Web shows that the widespread mechanisms of using spiders to fetch web pages, indexing them and calculating a matching score for each query, bears many problems. Most of them are related to the issue of choosing the correct index terms. While many advanced matching scores and techniques already exist, the selection of the index terms remains a big issue. There are many vocabularies to choose from; different organizations are prone to having different vocabularies.

That is where Tim Berners-Lee's notion of the Semantic Web comes into play [15]. His idea was to annotate human-readable web resources with metadata bearing a precise semantic. This enables so called web agents to process web content automatically without any interaction with the user. Aside from metadata models and a common syntax, a main collar of Tim Berners-Lee's Semantic Web was the definition of a standardized vocabulary. Such a vocabulary can be a plain list, a taxonomy - already capable of representing a hierarchy - or an ontology which features a set of elements with complex relationships among them.

Applied to the SOA paradigm the Semantic Web approach leads to a semantic service-oriented architecture. The underlying idea is the same as already proposed by Berners-Lee. Web services are annotated using certain taxonomies or ontologies to make them machine-readable. Without this machine readability, there would be no possibility for a matching based on semantics instead of plain keywords.

4 Security Architecture

In the following we introduce our distributed security architecture based on attribute-based access control (ABAC) and the semantic service-oriented architecture paradigm (SSOA). We start this section with a short outline of the underlying system.

4.1 System Architecture

As mentioned in Sect. 1, the service-oriented architecture (SOA) is a popular technology to outsource processes to remote locations providing a standardized representation of services and common mechanisms to register and discover them. In Sect. 3 we pointed out that SOAs still need to face the issue of different local vocabularies and index terms. Tim Berners-Lee's Semantic Web targets this issue and provides means to define common, machine-readable vocabularies like taxonomies and ontologies. These concepts are employed by SSOAs to combine the advantages of SOA and the Semantic Web.

Within the scope of the project Access-eGov (see Sect. 1) we developed a SSOA for a European-wide e-Government scenario. The overall system architecture of Access-eGov is depicted in Fig. 1. The Access-eGov architecture represents a Peer-to-Peer network with physically separated service providers and a set of supporting nodes. The employment of a Peer-to-Peer network satisfies the demand for a scalable and well maintainable platform of European e-Government systems.

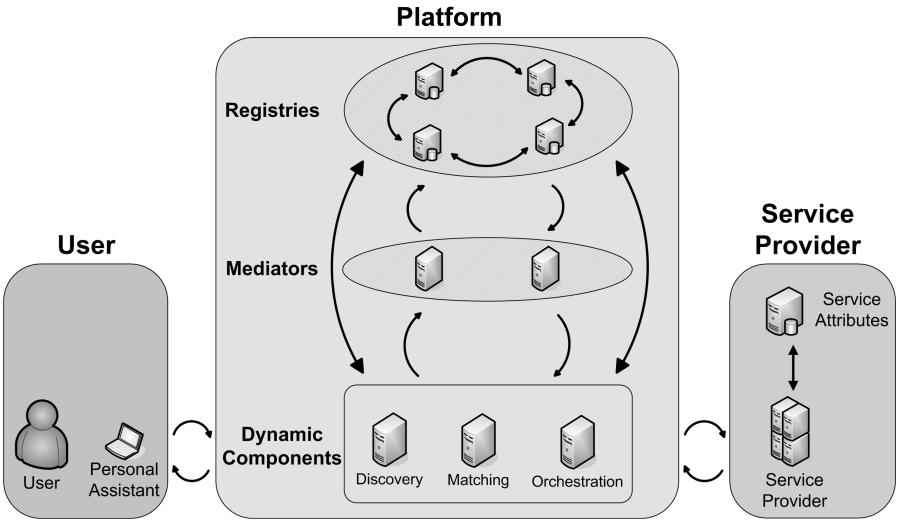

Fig. 1. Structural view of the Access-eGov architecture

Core components of the architecture are the dynamic modules for the discovery, matching and orchestration of services [16]. Supporting components of the platform are distributed storage facilities. These service and ontology repositories facilitate the semantic discovery of services. Annotated services of the service provider are registered in the service repository. The corresponding vocabulary used for the annotation is stored in the ontology repositories. For the semantic discovery the user query is matched with the service attributes using the corresponding "local" ontologies. The platform provides so-called ontology mediators that enable the mapping of ontologies. It also facilitates the composition of atomic services into complex scenarios, offering full coverage of citizens' life events.

The user communicates with the platform via a digital personal assistant. This user interface accesses the infrastructure functionality via standardized interfaces and communicates with the above mentioned components.

4.2 Security Architecture

As the Access-eGov platform deals to a high degree with personal information and brokers this information between a large base of services, special care has to be taken in the development of a security architecture. Examining a set of pilot scenarios we defined the following requirements [17]:

The most important requirement for a security architecture in the described scenario is that personal information stored and processed in the system is secured from unauthorized access. Furthermore, the architecture has to be scalable, as it has to cater for an unspecified but possibly large number of services in the system. The possibility of adding new dedicated security nodes without reconfiguration of the whole network is very important in this context. A fundamental requirement of the end users is privacy, especially since the system deals with sensitive personal data. The administrators on the other hand need a system that can be handled with low administrative overhead. If more than one entity is allowed to issue user credentials, not only the administrative overhead will be minimized but also competence conflicts among competing administration authorities can be eliminated.

Implementing these requirements, the following paragraphs will present our security architecture.

Distributed, Semantically-enriched ABAC To overcome the issue of different vocabularies we target to develop an ABAC approach that processes semantically-enriched attributes and performs a semantic mapping of attributes from different vocabularies. Figure 2 presents a picture of our security architecture. The picture resembles the already presented system architecture with a focus on the involved security components.

We build our architecture on the ABAC approach standardized by [12]. After the discovery of a queried service the user's personal assistant tries to access a service on the service provider's premises. The request is diverted to a Policy Enforcement Point (PEP) which - after consulting a Policy Decision Point

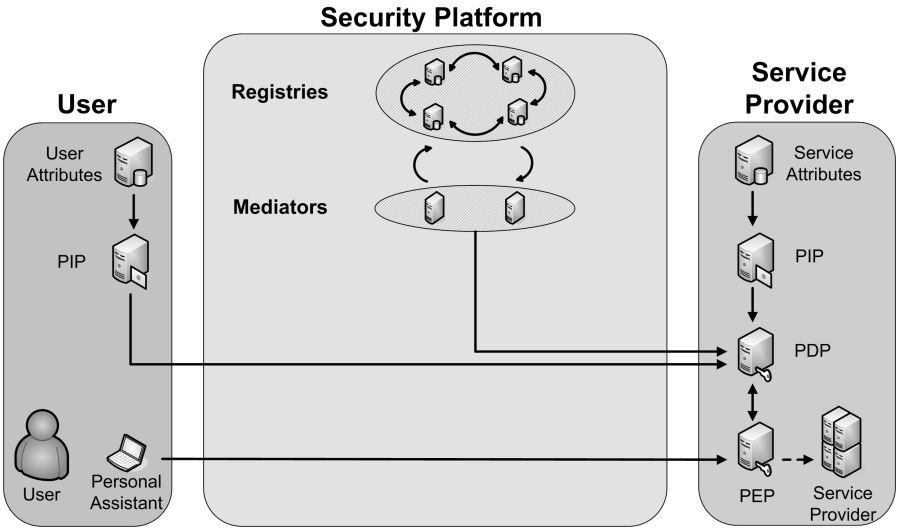

Fig. 2. Distributed security architecture with semantic components

(PDP) - grants or rejects access to the resource. To make an access decision, the PDP retrieves the service provider's access policy and collects attributes from both the user and the service, accessing the corresponding Policy Information Points (PIP). If attributes from the service requester and the provider come from domains using the same ontology, the PDP looks up the local ontology and performs a semantic matching. If attributes come from sources with different vocabularies, the PDP accesses the semantic components of the system architecture collecting the unknown vocabularies and employing ontology mediators for the semantic mapping.

An inference engine in the ABAC model, as introduced by Priebe et al. [18], is a step towards utilizing the potential of Semantic Web technologies in access control. A first notion of semantic mapping in [13] extends this approach by introducing the possibility of defining access-control relevant attributes in the Web Ontology Language (OWL). Domain independence is achieved through brokering the semantic descriptions using different ontologies. We extend this approach to encompass many different domains and their respective attribute combinations. Large-scale SSOAs like Access-eGov benefit from such a semantic attribute processing, even though the task of mapping attributes given in many different ontologies is complex.

Privacy and Trust As ABAC approaches usually deal with sensitive user attributes, we enhance our security architecture with mechanisms protecting privacy. In our architecture the Policy Decision Point (PDP) gathers all necessary attributes for an access decision. Protecting privacy, we do not consider user and service attributes to be stored in the platform. The PDP rather accesses the involved Policy Information Points (PIP) to collect only the data needed

for a particular access decision. As the PDP in our generic approach resides in the domain of the service provider, this controlled disclosure of attributes guarantees that other information about the user is not transmitted in any way. This is a first step towards a privacy-sensitive ABAC.

We further this approach by introducing the notion of privacy preferences in our security architecture. A user can fine-tune his digital personal assistant with regards to the attributes he wants to transfer under which circumstances and under which constraints. Circumstances and constraints can for example involve the context of transmission: A user might rather disclose a social security number to a bank than to an e-commerce shop. Also technical aspects can influence user preferences. For certain attributes the user might only approve a transmission to trusted authorities or via a SSL encrypted connection. A candidate for a proper language for defining privacy preferences is APPEL (A P3P Preference Exchange Language) [19], designed within the scope of the Platform for Privacy Preferences project[2] (P3P).

Extending the concept of user privacy preferences, we introduce the idea of flexible locations of the PDP. As the PDP's location is generally considered to be in the service provider's domain, a distributed ABAC always faces the issue of service providers reliant on attributes for the access decision and users not willing to disclose certain attributes, even if they are solely used for the access decision.

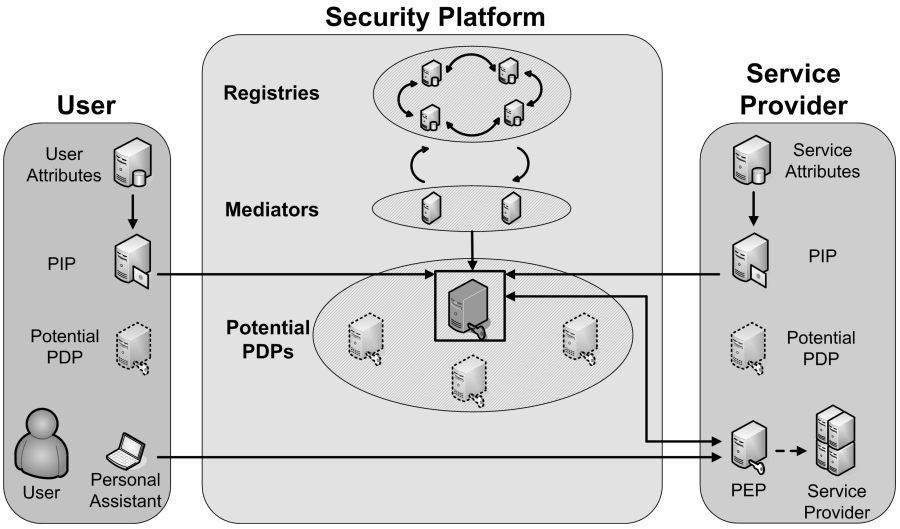

Fig. 3. Security architecture with flexible locations of the Policy Decision Point

[2] http://www.w3.org/P3P/

Our security architecture resolves this issue by giving users and service providers the possibility to choose from a set of PDPs. There are numerous potential positions for a PDP, from which we will describe three to show the extremes that limit the PDP positions. As mentioned above, the most common location of the PDP is a node of the service provider. This is the best alternative for the service provider, as it solely controls the entity deciding on the access control. However, this choice is prone to conflicting with privacy preferences of the user, as personal information needs to be handed out.

If the PDP is placed on a node between the user and the service provider, we have the classic concept of a trusted third party, both entities have to rely on. Based on user's privacy preferences and the service provider's trust settings both parties need to negotiate and agree about an acceptable position of the PDP in the platform. In this case user attributes for the access decision are transferred to a third party that does not conflict with his privacy preferences. The attributes are not disclosed to the service provider itself. As the service provider loses total control of the access decision, it needs to trust the third party to a certain level. Figure 3 depicts the scenario of a PDP between the user and the service provider.

The third and most extreme possibility is to place the PDP on the user side. This is only possible, if a correct execution of the PDP can be guaranteed to the service provider. Obviously, this is not viable with current hardware and operating systems. For this approach ideas of the Trusted Computing Group[3] initiative on hardware-secured platforms are of use. Such a platform is able to execute a PDP in a trusted environment enabling maximum privacy for the user, as no privacy-sensitive attributes are given away. On the other hand the necessary level of trust for the provider is guaranteed, as its interests are secured through the trusted platform. [20, 21, 22] present research in the trusted computing field.

It is noteworthy that a security architecture flexible enough for arbitrarily placing PDPs on platform nodes is able to handle the addition of new PDPs without reconfiguration of the whole network, resulting in improved scalability of the architecture.

5 Implementation

In the previous sections we laid out a security architecture built in the project Access-eGov (see Sect. 1). For the evaluation of our architecture we are in the process of building a prototype security system that is intended to serve as security facility of the Access-eGov platform. A lively open source community allows us to reuse existing and proven solutions like the following, which are at the same time our main building blocks.

Attribute certificates according to the ITU-T Recommendation [6] are a promising way of relating identities and attributes. [23] proposes an approach

[3] https://www.trustedcomputinggroup.org/home

to a complete attribute certificate framework. In a previous project the authors have implemented initial X.509 attribute certificate support for the widely used crypto provider bouncycastle[4] which was the main reason for choosing it for this project. The specific task of attribute certificates in the Access-eGov architecture is to link attributes to users. A user with a set of attributes and some additional metadata is what we refer to as a user profile.

The concept of the flexible placement of the Policy Decision Point (PDP) establishes the need to pass on authorization decisions between requesters and enforcers of those decisions. The idea of Single Sign On (SSO) ideally fits to this concept of distribution, as in SSO models the entity doing the access control decision (the PDP) and the entity making use of the result of that decision (the Policy Enforcement Point) are not identical as well. The Security Assertion Markup Language (SAML) [24], used in many SSO projects, is our choice for the task of passing the authorization decisions between the nodes in our platform.

Before even being able to pass SSO tokens around, the system first needs to arrive at an access control decision. As previously mentioned, our architecture partially follows the XACML specification [12]. For this reason, it is a logical choice to use XACML as the language for our authorization requests and access control results.

It is not necessary to reinvent ontologies for semantic service-oriented architectures, because there is a number of promising projects in this area. After a careful evaluation and selection process, we picked the Web Service Modeling Ontology (WSMO) [25], including WSMO's ontology language WSML, for semantically describing our services and the Web Service Execution Environment (WSMX)[5] as the underlying technology platform. While there are very good reasons for choosing WSMO and WMSX, neither one has any special preparations for security concepts. Therefore the Access-eGov project will extend WSMX and to a certain extent WSMO to be able to accommodate our security infrastructure.

6 Conclusions

Due to the distributed character of modern information systems the requirements of a suitable security architecture have changed significantly. Modern distributed architectures favor security concepts focusing on dynamic attributes rather than static information. Furthermore, users' privacy concerns move to the center of attention, as users are not willing to pass personal information to every service provider.

In this paper we presented a distributed security architecture in a semantic service-oriented architecture (SSOA) focusing on dynamic access control. We built the access control component upon the existing attribute-based access

[4] http://www.bouncycastle.org
[5] http://www.wsmx.org

control model. In order to overcome the issue of diverging attribute vocabularies in a distributed environment, we integrated semantic components of the underlying SSOA for semantic attribute matching and the mapping of different ontologies. Furthermore, our approach facilitates the movement of the Policy Decision Point (PDP) from the service provider to a trusted location in the architecture. Based on the user's privacy preferences and the service provider's trust settings a PDP can be chosen dynamically.

Future work will involve ways to express and edit privacy preferences on user side as well as trust settings of the service provider. We further want to pursue the technical possibility of creating a trusted environment on user side in order to move the PDP to the user, the ultimate privacy solution.

Acknowledgment

The work reported in this paper is done within the Access-eGov project which is funded by the European Union under the Sixth Framework Program (Contract No. FP6-2004-27020). We would like to thank our project partners for helpful comments and stimulating discussions.

References

1. MacKenzie, C. M. and Laskey, K. and McCabe, F. and Brown, P. F. and Metz, R. Reference Model for Service Oriented Architecture 1.0. *OASIS Standard*, October 2006.
2. J. Lopez, R. Oppliger, and G. Pernul. Authentication and Authorization Infrastructures (AAIs): A Comparative Survey. *Computers & Security*, 23(7):578–590, 2004.
3. R. Sandhu and P. Samarati. Access Control: Principle and Practice. *Communications Magazine, IEEE*, 32(9):40–48, 1994.
4. E.B. Fernandez and G. Pernul. Patterns for Session-Based Access Control. In *Proc. of the Pattern Languages of Programming conference (PLoP '06)*, October 2006.
5. C.J. McCollum, J.R. Messing, and L. Notargiacomo. Beyond the Pale of MAC and DAC - Defining New Forms of Access Control. In *IEEE Symposium on Security and Privacy*, pages 190–200, 1990.
6. ITU-T Recommendation. X.509: The Directory – Public Key and Attribute Certificate Frameworks, March 2000.
7. S. Farrell and R. Housley. RFC3281: An Internet Attribute Certificate Profile for Authorization. *Internet RFCs*, 2002.
8. W. Johnston, S. Mudumbai, and M. Thompson. Authorization and Attribute Certificates for Widely Distributed Access Control. In *Proc. of the 7th Workshop on Enabling Technologies (WETICE '98)*, pages 340–345, Washington, DC, United States, 1998. IEEE Computer Society.
9. J.S. Park and R. Sandhu. Smart Certificates: Extending X.509 for Secure Attribute Services on the Web. In *Proceedings of the 22nd National Information Systems Security Conference (NISSC)*, October 1999.

10. D. Chadwick, A. Otenko, and E. Ball. Role-based Access Control with X.509 Attribute Certificates. *IEEE Internet Computing*, 7(2):62–69, 2003.
11. N.R. Adam, V. Atluri, E. Bertino, and E. Ferrari. A Content-based Authorization Model for Digital Libraries. *IEEE Transactions on Knowledge and Data Engineering*, 14(2):296–315, 2002.
12. T. Moses. eXtensible Access Control Markup Language (XACML) Version 2.0. *OASIS Standard*, February 2005.
13. T. Priebe, W. Dobmeier, and N. Kamprath. Supporting Attribute-based Access Control with Ontologies. In *Proc. of the 1st International Conference on Availability, Reliability and Security (ARES '06)*, pages 465–472, Los Alamitos, CA, United States, 2006. IEEE Computer Society.
14. R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, United States, 1999.
15. T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, May 2001.
16. P. Bednar, S. Dürbeck, J. Hreno, M. Mach, R. Lukasz, and R. Schillinger. Access-eGov Platform Architecture. Access-eGov deliverable D3.1, October 2006.
17. R. Klischewski, S. Ukena, and D. Wozniak. User Requirements Analysis & Development/Test Recommendation. Access-eGov deliverable D2.2, July 2006.
18. T. Priebe, W. Dobmeier, B. Muschall, and G. Pernul. ABAC - Ein Referenzmodell für attributbasierte Zugriffskontrolle. In *Proc. of the 2nd Jahrestagung Fachbereich Sicherheit der Gesellschaft für Informatik (Sicherheit '05)*, pages 285–296, 2005.
19. L. Cranor, M. Langheinrich, and M. Marchiori. A P3P Preference Exchange Language 1.0 (APPEL 1.0). *World Wide Web Consortium Working Draft*, April 2002.
20. B. Balacheff, L. Chen, S. Pearson, D. Plaquin, and G. Proudler. *Trusted Computing Platforms: TCPA Technology in Context*. Prentice Hall PTR, Upper Saddle River, NJ, United States, 2002.
21. T. Garfinkel, B. Pfaff, J. Chow, M. Rosenblum, and D. Boneh. Terra: A Virtual Machine-based Platform for Trusted Computing. In *Proc. of the nineteenth ACM symposium on Operating systems principles (SOSP '03)*, pages 193–206, New York, NY, United States, 2003. ACM Press.
22. R. Sandhu and X. Zhang. Peer-to-Peer Access Control Architecture Using Trusted Computing Technology. In *Proc. of the tenth ACM symposium on Access control models and technologies*, pages 147–158, New York, NY, United States, 2005. ACM Press.
23. J.A. Montenegro and F. Moya. A Practical Approach of X.509 Attribute Certificate Framework as Support to Obtain Privilege Delegation. In *Proc. of the 1st European PKI Workshop (EuroPKI '04)*, pages 160–172. Lecture Notes in Computer Science (LNCS), 2004.
24. John Hughes, Eve Maler, and Rob Philpott. Technical Overview of the OASIS Security Assertion Markup Language (SAML), Version 1.1, May 2004.
25. D. Roman, H. Lausen, and U. Keller. Web Service Modeling Ontology (WSMO). WSMO deliverable D2v1.3, October 2006.