ROLE-BASED ACCESS CONTROL FOR INFORMATION FEDERATIONS IN THE INDUSTRIAL SERVICE SECTOR

Kunz, Steffen, Humboldt-Universität zu Berlin, Spandauer Str 1, 10178 Berlin, Germany, steffen.kunz@wiwi.hu-berlin.de

Evdokimov, Sergei, Humboldt-Universität zu Berlin, Spandauer Str 1, 10178 Berlin, Germany, evdokim@wiwi.hu-berlin.de

Fabian, Benjamin, Humboldt-Universität zu Berlin, Spandauer Str 1, 10178 Berlin, Germany, bfabian@wiwi.hu-berlin.de

Stieger, Bernd, ABB Corporate Research Center Germany, Wallstadter Strasse 59, 68526 Ladenburg, Germany, bernd.stieger@de.abb.com

Strembeck, Mark, Vienna University of Economics and Business (WU Vienna), Augasse 2-6, A-1090 Wien, Austria, mark.strembeck@wu.ac.at

Abstract

Information federations promise an enhanced collaboration between individual stakeholders in the life cycle of commercial products, including software and hardware products from arbitrary business sectors. However, information sharing across corporate borders must be controlled by tailored mechanisms for enforcing individual business confidentiality and integrity requirements. One influential current security paradigm to achieve this goal is the application of Role-Based Access Control (RBAC).

Based on ongoing work in the Aletheia project on service-oriented information federation, we present a case study on applying RBAC for information sharing among multiple stakeholders in the industrial service sector. We place a special emphasis on the methodical, tool-supported elicitation and definition of RBAC policies in this environment. In addition, we use the eXtensible Access Control Markup Language (XACML) to transfer RBAC policies between the different nodes in information federations. Further, we present a corresponding security architecture in which those XACML policies are applied for authorization decision and enforcement. The case study was conducted in cooperation with ABB, a large company providing power and automation technologies, products, and services for utility and industry customers.

Keywords: Information Security, Role-Based Access Control, Information Federation, Industrial Services.

1 INTRODUCTION

The emergence of the Internet of Things and Services as well as the increasing availability of detailed, but distributed and very heterogeneous information pose new challenges for today's corporate information systems (IS). This is especially true for information created throughout the life cycle of commercial products, including software and hardware products from arbitrary business sectors (Ameri and Duta 2005). Today's corporate IS are most often not capable of federating the huge amount of product information that is available from business applications, databases, and data warehouses, distributed across multiple organizations. This problem intensifies with the increased use of Web 2.0 information sources (wikis, blogs, social networks, etc.), inter-corporate web-services, smart-item infrastructures based on sensor networks or Radio Frequency Identification (RFID), and the emerging EPCglobal Network.

The Aletheia project (Aletheia 2009) – funded by the German Federal Ministry of Education and Research and organized as a consortium of industry and academic partners – aims to provide a solution for this problem by creating an IS architecture, combining Service-Oriented Architectures (SOA) and semantic technologies. The architecture should provide means for extracting, federating, and refining the comprehensive information across entire product life cycles. This is achieved by deploying components responsible for data aggregation across stakeholders involved in the life cycle of the product, and federating the data by exchanging it via so-called *Aletheia Service Hubs* (ASH) representing semantic-aware information gateways in the Aletheia SOA (Wauer et al. 2009).

In addition to the task of federating distributed information, another important challenge is the design and enforcement of security mechanisms for the Aletheia system. In this paper, we focus on the protection of information against unauthorized access, a necessary precondition to fulfill confidentiality and integrity requirements of the various stakeholders involved. The complexity and inter-organizational character of Aletheia, as well as the heterogeneity and distributed nature of its information sources, make the implementation and enforcement of access control (AC) a challenging task (see, e.g., Bacon et al. 2003)

In our research, we followed a case study approach (see Eisenhardt 1989). In particular, by analyzing the service portfolio of ABB, we identified major issues for semantic data federations that can arise in the industrial service sector (Section 2). Further, we describe how we applied the scenario-driven role-engineering approach proposed by Neumann and Strembeck (2002) for engineering intra-organizational AC policies tailored for our case study (Section 3). In order to exchange RBAC policies in the Aletheia system, we use an export filter that converts the engineered RBAC policies into the eXtensible Access Control Markup Language (XACML), an OASIS standard for the definition of AC policies. In addition, we present the initial design of a service-oriented security architecture capable of enforcing these XACML security policies in semantic data federations. Finally, we discuss outcomes of the case study, identify issues and give recommendations on how they could be solved in future research, and present an overview on related work (Section 4).

2 PROBLEM IDENTIFICATION

2.1 Case Study: Information Federations in the Industrial Service Sector

In the initial stage of the Aletheia project, the potential application domains for semantic information federations in the life cycle of a product (as established by Ameri and Duta 2005) have been concretized, facilitating a common and holistic understanding of the related processes and requirements. Several use cases in each of the eight stages of the product life cycle – requirements

analysis, development & design, manufacturing, sales, operation, maintenance, and recycling – have been identified (Evdokimov 2009).

In this paper, we examine one particular stage of the product life cycle in detail, which is product maintenance at ABB, a large company providing power and automation technologies as well as products and services for utility and industry customers. The case study design is based on the qualitative approach proposed by Eisenhardt (1989). Utilizing this approach allowed us to develop a holistic description, interpretation, and understanding of relationships and requirements of the industrial service process phases. We interviewed several service dispatchers, technicians, experts, and some contact center agents to collect details about the current service processes and their associated security requirements. The interviews have been performed according to the semi-structured interview guide of Yin (2003) and complemented by personal face-to-face interviews as suggested by Frey and Oishi (1995). This approach helped us to reduce possible misunderstandings during the process of data collection. Moreover, workshops with participants from several service business units, mainly service technicians and experts, have been organized to verify and validate the collected service process information.

In our case study four different parties are involved: The S*ervice Provider* (SP), here ABB, who is selling, integrating, and maintaining the devices used for power plants (e.g., dynamos, voltage converters, and turbines). The *Logistics Provider* (LP) manages the warehousing and shipping of spare parts and new devices for the service provider. The *customer company* of the service provider uses the devices in its electric power plants. Due to the remote location of this customer company, the service job is not conducted by the service provider itself, but by a *Partner Company (PA)* which is sub-contracted by the service provider.

In general, the service provider and the partner company should be able to provide an equivalent service. However, due to confidentiality policies, access rights of a partner company to product-related information are restricted. Subsequently, we investigate the case where a partner company conducts the service job (see Figure 1). In this example, the plant manager of the customer (1) *reports a problem to the service provider* by connecting to Aletheia, where detailed information about the problem is recorded. The service dispatcher uses this information to (2) *dispatch a technician to the service job* – in this case a technician of the partner company. He also *prepares a service package* with important information about the service job, which is forwarded to the partner technician. The partner technician drives on-site to the customer's power plant, where he first has to (3) *identify the defective device* supported by information provided by Aletheia. Then, he begins the (4) *repair of the defective device*, using additional information provided by Aletheia. After the problem has been solved, (5) device-specific test and calibration measures are conducted to *update the service history* of the device.

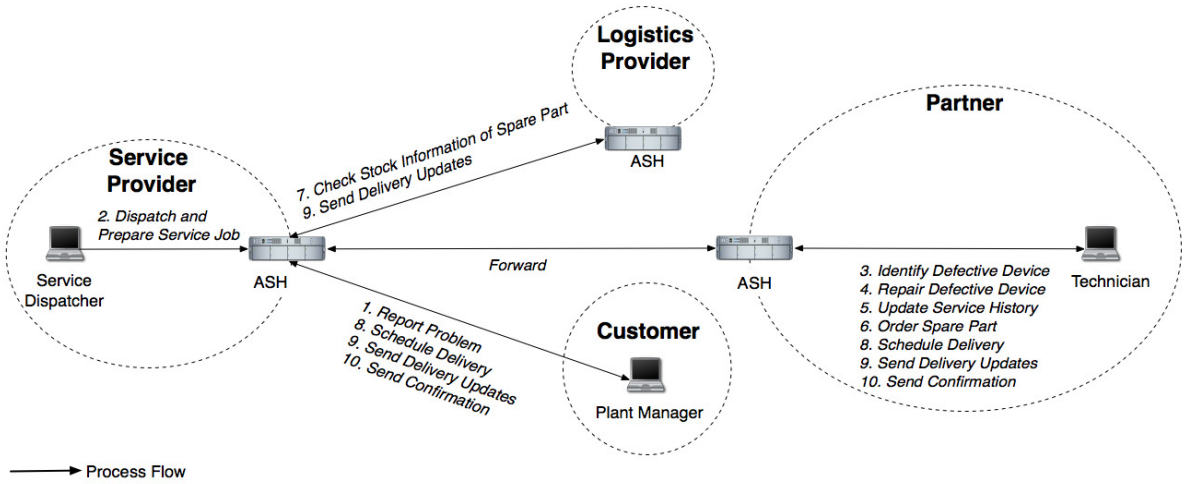

Figure 1. Process Flow between Actors and Aletheia Services in the Different Domains of an Industrial Service Job.

In case a part of the defective device needs to be replaced, a (6) *spare part has to be ordered* from the logistics provider and delivered to the customer's plant. Aletheia (7) *checks the stock information* of the logistics provider warehouses and (8) *schedules delivery and meeting location as well as the date* with the partner technician and customer company. During the shipping of the device, (9) *the status of the delivery is updated*. Based on these updates, Aletheia knows exactly when the spare part is going to arrive at the customer's power plant, and thus can update the scheduled meeting of the partner technician and the plant manager. When the spare part was delivered to the power plant, the partner service technician and the plant manager (10) *confirm the delivery*. Steps (6) to (10) are modeled in finer granularity than step (1) to (5) because these steps are used as example for the scenario-driven role engineering approach in Section 3 and will therefore also be referred to as '*Shipping of Spare Part'* scenario in the rest of this work.

2.2 The Need for Access Control in Information Federations

In the example depicted in the Figure 1, four different parties or domains are involved (service provider, logistics provider, partner company, and customer company), three human actors (service dispatcher, plant manager, and partner technician), and three computer services (ASHs). The underlying idea of information federation is to provide information sharing across these different actors, services, and domains, e.g., if the partner technician wants to know the location of a certain device inside the customer's plant. This information is stored in the domain of the service provider, but not in the domain of the partner company. In order to get access to this information, the partner technician needs the corresponding access rights. But this demand is not trivially satisfiable by granting him full access to all information, since the service provider's semantic data store may also contain sensitive business information that should not be accessible to the partner technician (e.g., the technician could use sensitive contract information from the service provider for acquiring the customer for his own company, or for renegotiating the price of his service sub-contract).

The same applies to the exchange of shipping information between logistics and service providers. Most likely, the service provider is not interested in sharing all information stored in his semantic repository with the logistics provider because the logistics provider might also have customers that are competitors to the service provider. Without sound AC policies, these competitors might, e.g., be able to get access to maintenance histories of the service provider's devices and use this information in an unauthorized manner. Accordingly, the service provider is not willing to provide more information than absolutely necessary to the logistics provider in order to conduct the shipping of the spare part. This principle, i.e., providing federated information by Aletheia only on a *need to know* basis, is referred to as the *least privilege* paradigm of information security in the literature (see, e.g., Ferraiolo and Kuhn 1992). Similar reasoning applies to access rights for updating and deleting information in the federated data stores. It serves to help preserving data confidentiality and integrity without constraining necessary functionality.

2.3 Shortcomings of Current Access Control Systems

Today, access control mechanisms deployed in real-world IS are often not prepared to cope with the issues arising in information federations. In the context of globally operating enterprises, it is quite common that not only every company, but also every single underlying data source has its own AC policies. For instance, in a historically grown IT landscape, ERP systems, document management systems, and installed-base management systems usually have their own proprietary AC schemes, whereas file-servers often rely on the user-right management of the underlying operating system.

Global companies are also characterized by a certain organizational structure that adds additional complexity to the AC design. In our case study, the service provider uses the matrix structure as an organizational structure. The structure is characterized by many authorization domains as well as redundancies in these domains, e.g., apart from country specific IS and security managers, there exist

additional IS and security managers in the individual business units. Furthermore, there are AC managers for every single data source and system. It is already a challenge to coordinate all of these participants within a single company regarding a comprehensive AC design and policy. Often, on a higher organizational level, only more general security aspects are dealt with, whereas the real AC responsibility lies with the AC managers of the specific data sources. As a consequence, the maintenance effort increases with a rising number of data sources. Also a combination of AC policies is not easy because of potential policy conflicts, which make it difficult to model them in a central and holistic matrix-based AC system.

The complexity further increases when different parties with different organizational structures, heterogeneous data sources, and AC designs want to share information. From the perspective of semantic data federations, the current AC approaches are most often not appropriate, since attention would have to be paid to the individual AC policies of each data source and IS, which would become tedious in practice and not scalable if federation partners change, e.g., if new service partners are subcontracted.

3 DESIGN AND DEVELOPMENT OF ACCESS CONTROL POLICIES AND ARCHITECTURE

3.1 Engineering of Tailored Role-Based Access Control Policies

In recent years, RBAC (cf. Ferraiolo and Kuhn 1992, Sandhu et al. 1996) developed into a de facto standard for access control in both research and industry. It uses roles to model the work profiles of an organization (see Neumann and Strembeck 2002, Strembeck 2005). In particular, roles are equipped with the permissions needed to fulfill their tasks. Subsequently, these roles are assigned to human users or other active system entities (also called subjects). Therefore, RBAC directly supports the principle of least privilege because each user is assigned to the exact roles, and thus owns the exact number of permissions, that are needed to fulfill his duties. Moreover, thoroughly engineered roles tend to change significantly slower than the assignment of individuals to these roles. Thus, establishing roles as an abstraction mechanism for subjects significantly facilitates the administration of permissions.

Role engineering is the process of defining roles, permissions, constraints, and role-hierarchies. In our case study, we applied the scenario-driven role-engineering process (see Neumann and Strembeck 2002, Strembeck and Neumann 2004) for the elicitation and specification of RBAC policies. Scenario-driven role engineering is a systematic approach to engineer tailored RBAC models. It uses scenario and process models as a primary communication and engineering vehicle. Since its first publication in 2002, scenario-driven role engineering was successfully and repeatedly applied in practice, and is nowadays used by a number of consulting firms and international projects (see, e.g., Coyne and Davis 2008).

The control flow of the scenario-driven role-engineering process is tailored to fit the context parameters of a particular role-engineering project. Figure 2 shows the tailoring we used in our case study. First, (a) *scenarios and processes are identified and modeled* with (structured) text descriptions and different types of diagrams, e.g., message sequence charts, activity diagrams, or Petri-nets. A scenario consists of one or more steps, and each step may be part of several scenarios. Further, each step (potentially) invokes an action, and each action consists of an operation and a target object. Then, we use the scenarios and processes of a particular organization or a particular IS to (b) *derive permissions* and to define tasks. In addition to the permissions, (c) *constraints need to be identified*. Then, the (d) *scenario model has to be refined*; if the scenario model is incomplete it is revised and or extended. If the scenario model is complete, (e) *tasks are then used to define work profiles* that (f) *serve as preliminary role*s and can be used to (g) *define the tailored RBAC model* (for details see Neumann and Strembeck 2002, Strembeck 2005).

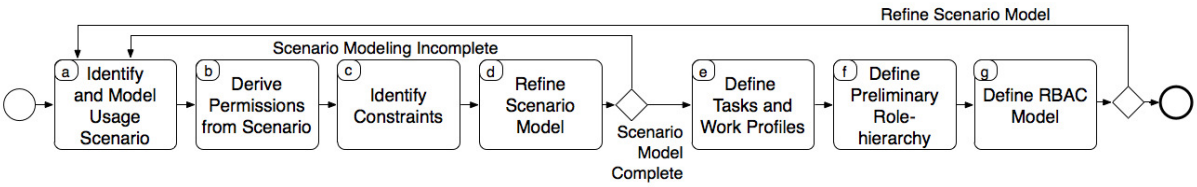

Figure 2. The Scenario-driven Role-engineering Process (cf. Neumann and Strembeck 2002).

In our case study, we modeled 14 scenarios and derived 150 permissions. We used the scenarios to define 11 tasks, to derive 8 work profiles, and 8 roles. However, due to article-size restrictions, we only provide detailed description for step (a) and the final RBAC model after step (g).

Figure 3 shows a message sequence chart that models the 'Shipping of Spare Part' scenario, i.e., steps (6) to (10) of the process flow described in Figure 1. We selected this scenario because it demonstrates a number of inter-organizational information flows. In particular, four domains (service provider, partner company, logistics provider, and customer company), two human actors (partner technician and plant manager) and three computer services (SP ASH, PA ASH, and LP ASH) are involved. The scenario describes the initial order by the partner technician, which is forwarded to and processed by the SP ASH and LP ASH. Sequence step (7.1), which is marked with an asterisk, shows the interaction between SP ASH and LS ASH and will be used as an example for the creation of XACML-based RBAC policies described later in Section 3.2., Figure 5.

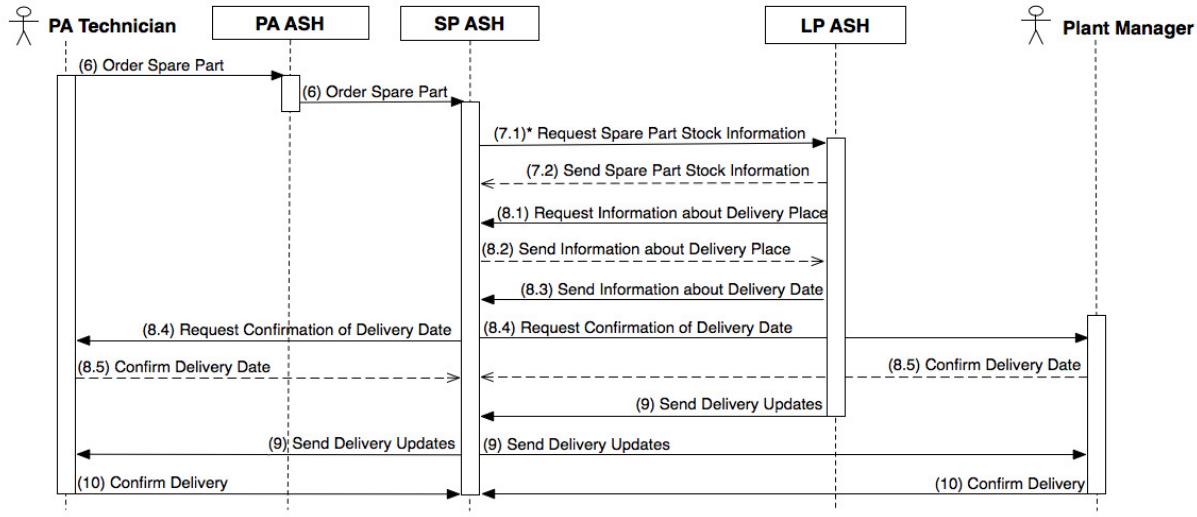

Figure 3. Shipping of Spare Part Scenario as UML Sequence Chart (cf. Process Steps (6) to (10) in Figure 1).

For steps (a) to (f) of the scenario-driven role-engineering process (Figure 2), we used the graphical software tool *xoRET* (Strembeck 2005). As an output xoRET produces preliminary descriptions of the roles that were further refined using another tool, *xoRBAC* (see Strembeck and Neumann 2004), thus completing step (g). Figure 4 displays a screenshot of the xoRBAC tool that illustrates a fragment of the results of the scenario-driven role engineering process for the 'Shipping of Spare Part' scenario. From the 16 steps depicted in Figure 3, 14 permissions were derived, which were assigned to six roles (for simplification we named the roles according to the corresponding actors and services in this case study; the role Guest is always created by default). In Figure 4, the role `SP_ASH` is selected showing six directly assigned permissions on the right-hand side of the screen shot.

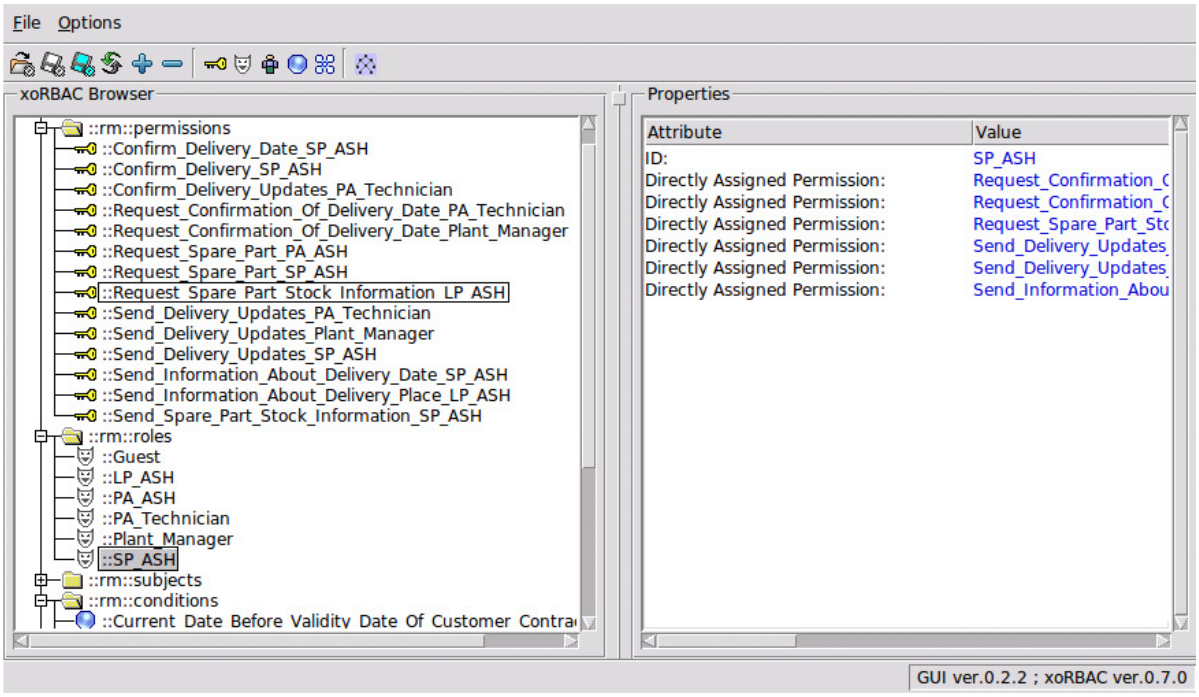

Figure 4. Screenshot of the Shipping of Spare Part RBAC Model in xoRBAC (cf. Sequence Step 7.1 in Figure 3).

3.2 Using XACML as a Common Access-Control Policy Language

The eXtensible Access Control Markup Language (XACML) (Moses 2005) is a standard adopted by the *Organization for the Advancement of Structured Information Standards* (OASIS). XACML provides an XML-based language for the definition of access-control policies. The language syntax is formalized via an XML schema. The abstract nature of XACML and its recognition as a generally accepted standard makes it a perfect candidate for describing and implementing access-control policies in the distributed and heterogeneous environment of Aletheia, offering interoperability between different IS and domains. The existence of free open-source libraries implementing the XACML standard also reduces development effort associated with introducing XACML support for Aletheia. In addition, XACML provides an RBAC profile (Anderson 2005) for expressing policies that use role-based access control. An XACML profile is a specific interpretation of the XACML XML schema, which does not define new elements – i.e., a profile defines how XACML can be used to support certain models or services, like RBAC, for example.

In order to create XACML policies for a given RBAC model, we extended the functionality of the xoRBAC tool by an export filter that allows producing XACML policies compliant with the RBAC profile. Figure 5 shows a fragment of the XACML policy for sequence step 7.1 in Figure 3 and the respective permission `request_spare_part_stock_information_LP_ASH` highlighted in Figure 4. On the left-hand side, the `<Target>` and `<PolicySet IdReference>` sections of the role policy set `RPS:SP_ASH:role` define the mapping between the `SP_ASH` role and permission policy set `PPS:SP_ASH:role`. The actual description of the permission policy set `PPS:SP_ASH:role` is provided on the right side and defines the permission to perform the `request_spare_part_stock_information` action on the resource `LP_ASH_repository`.

```
<PolicySet PolicySetId="RPS:SP_ASH:role"
 PolicyCombiningAlgId="&policy-combine;permit-overrides">
  <Target>
   <Subjects>
    <SubjectMatch MatchId="&function;anyURI-equal">
     <AttributeValue DataType="&xml;anyURI">
      &roles;SP_ASH
     </AttributeValue>
     <SubjectAttributeDesignator AttributeId="&role;"
      DataType="&xml;anyURI"/>
    </SubjectMatch>
   </Subjects>
  </Target>
  <PolicySetIdReference>
   PPS:SP_ASH:role
  </PolicySetIdReference>
 </PolicySet>
```

```
<PolicySet PolicySetId="PPS:SP_ASH:role"
   PolicyCombiningAlgId="&policy-combine;permit-overrides">
   <Policy PolicyId="Permissions:specifically:for:the:SP_ASH:role"
    RuleCombiningAlgId="&rule-combine;permit-overrides">
   <Rule RuleId="Permission:to:perform:request_spare_part_stock_
   information:on:LP_ASH" Effect="Permit">
    <Target>
     <Resources>
      <Resource>
       <ResourceMatch MatchId="&function;string-equal">
        <AttributeValue DataType="&xml;string">
         LP_ASH_repository
        </AttributeValue>
        <ResourceAttributeDesignator AttributeId=
         "&resource;resource-id" DataType="&xml;string"/>
       </ResourceMatch>
      </Resource>
     </Resources>
     <Actions>
      <Action>
       <ActionMatch MatchId="&function;string-equal">
        <AttributeValue DataType="&xml;string">
         request_spare_part_stock_information
        </AttributeValue>
        <ActionAttributeDesignator AttributeId="&action;action-id"
         DataType="&xml;string"/>
       </ActionMatch>
      </Action>
     </Actions>
    </Target>
   </Rule>
  </PolicySet>
```

Figure 5. Example XACML Role and Policy Definition for Sequence Step 7.1 (cf. Figure 3 and Figure 4).

3.3 Security Architecture for RBAC-Policy Enforcement

Once access control policies have been defined, the system architecture must provide means to enforce them. Figure 6 depicts the integration of RBAC components into the Aletheia SOA (Wauer et al. 2009) from a high-level perspective. In Aletheia, every company possesses a *Protected Information Store*, which stores the relevant and in part confidential information for the application at hand, and which is accessed by external business partners through the ASH. In addition, every company deploys its own set of security components and has full control of role and policy definition within its domain, providing flexibility and scalability. In every company, a *Policy Administration Graphical User Interface (GUI),* in our case the combination of xoRET and xoRBAC, serves as the user interface to the RBAC system; here a security administrator can create or change RBAC policies in advance and during runtime. Those policies are stored in a *Policy Repository*, a dedicated and only internally accessible database for XACML policies.

Requests to retrieve or update information in the Protected Information Store are intercepted by the ASH. The ASH is responsible for authentication procedures; when the origin of the request has been authenticated, a decision request is sent from the ASH to the *Policy Decision Point (PDP)*, i.e., the component of the AC system where the request is checked for its authorization and compliance to the policy set, and a corresponding decision for granting or denying access to the information is made. In particular, the PDP makes a decision based on the policies stored in the corresponding Policy Repository. The decision is transmitted back to the ASH, which now serves as the *Policy Enforcement Point*: According to the AC decision made, it answers or denies the information request, and withholds or passes on the information from the *Protected Store*. Using XACML as a common policy language enables us to combine currently developed Aletheia components like the ASH enforcement component with existing and proven open-source frameworks for XACML-policy decision points, and with xoRET and xoRBAC for policy administration.

Our design assumes that further SOA security mechanisms for Aletheia are deployed. For example, authentication, message encryption, and digital signatures are additional building blocks to construct a comprehensive security architecture. Those mechanisms are currently implemented using existing Web Service (WS) components, which are compliant to common WS-Security standards (Oasis 2006).

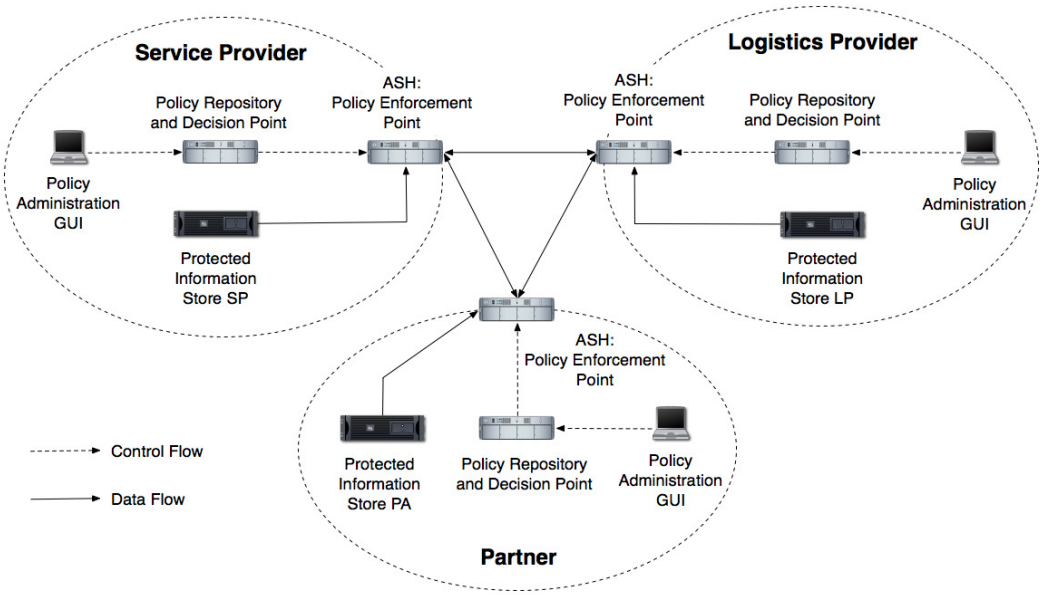

Figure 6. Security Architecture for Inter-organizational RBAC.

4 EVALUATION OF THE CURRENT WORK

4.1 Results of the Current Work

In this paper, we presented a case study for engineering and enforcement of RBAC policies in trans-organizational information federations. In particular, we focused on scenario-driven role engineering, AC policy transportation with XACML, and the policy enforcement in a security architecture tailored to our case study. However, this approach is not only restricted to this particular case study, but can be considered as a generic approach applicable to distributed and federated IS in general. Scenario-driven role engineering allows for decomposing the complexity of federated IS into small subsystems (here scenarios), thus providing a good scalability on the technical and organizational layer. Further, the distributed security architecture fits the distributed nature of the federated information sources and the individual control demands of all stakeholders. From a technical perspective, the chosen XACML standard enables the transport and exchange of RBAC policies between the individual components of our security architecture.

However, we have also encountered several problems that still have to be solved: Our RBAC policies have been created by a central design team with knowledge about all the intra- and especially most inter-organizational processes of the different companies. In most business scenarios it is not very common to share internal process information across company borders. Accordingly, the question arises: How can inter-organizational RBAC policy creation be supported in a systematic fashion? Similar issues have been discussed in the business process management community focusing on process choreographies. Considering the fact that some approaches for extracting RBAC models from process models (Mendling et al. 2004) have already been developed, an extension of the scenario-driven role-engineering process through the coordination of inter-organizational stakeholders

(choreography) seems to be promising (see Figure 7). In the recently published Business Process Management Notation 2.0 (BPMN 2.0), artifacts for inter-organizational process choreography were integrated (see BPMN 2009), which should also be examined in detail for future work on scenario-driven role-engineering choreography.

In our case study, we identified the need to integrate pre-existing AC policies in the role engineering activities. In information federations, the integration of pre-existing policies is of special importance, as we extract information from organization-internal sources and federate them in inter-organizational repositories. However, the scenario-driven role engineering approach presented in Neumann and Strembeck (2002) does not address the integration of pre-existing policies. Therefore, in Figure 7, we included the integration of existing AC policies as step (x) into the scenario-driven role-engineering process. From a technical standpoint, the question arises, how the source AC policies can be linked or attached to the information chunks extracted from the information sources.

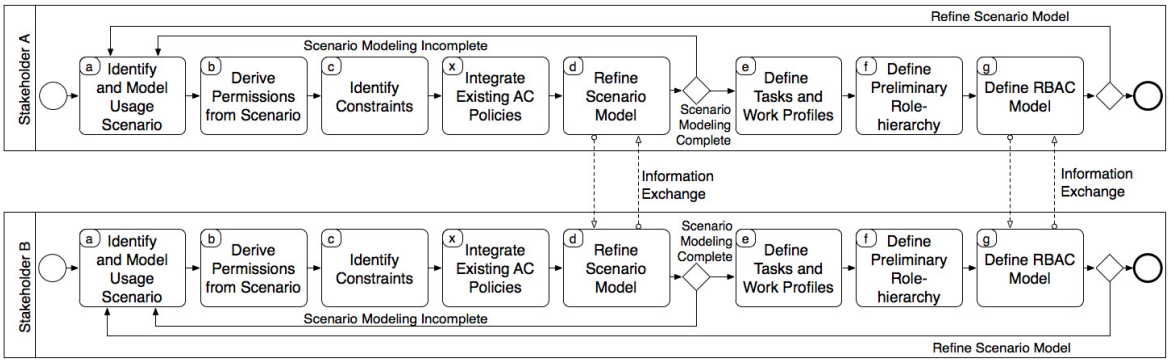

Figure 7. Inter-organizational Scenario-driven Role-engineering Process Choreography.

Since, depending on the application scenario, not only the actual data, but also the integrating ontology of Aletheia itself may be subject to confidentiality requirements, it must be investigated, if AC enforcement for this ontology has to be developed (see, e.g., Reddivari et al. 2007). In addition, we are also investigating practical problems like the long-term administration of distributed RBAC policies (see, e.g., Bacon et al. 2003, Dekker et al. 2008) and technical issues, like semantic-aware AC and implementing policy-enforcement modules for the ASH and semantic data repositories in general.

4.2 Further Related Work

The problem of information sharing and security in dynamic coalitions (see Cohen et al. 2002) is related to our work. A coalition consists of two or more different organizations (resp. their employees) that temporarily work together to achieve a common goal. Coalitions can be formed dynamically, and each coalition member may share a number of information resources with other coalition members. However, each party must be able to individually tailor the access rules for their content according to the status of other coalition members. Belokosztolszki et al. (2003a) present an approach to control information flow in (and out of) the OASIS RBAC system (Bacon et al. 2003). In particular, they use so-called “contexts” to classify elements of their RBAC system. These contexts are applied to control information flows between system entities. In Belokosztolszki et al. (2003b) the integration of OASIS role-based access control into the Hermes middleware platform (Pietzuch and Bacon 2002) is presented.

In distributed environments, where stakeholders use XACML for defining their own independent AC polices, it can be convenient to generate a single general policy consistent with the policies of the stakeholders (Mazzoleni et al. 2008, Li et al. 2009). In Aletheia, however, we emphasize a coordinated, but distributed policy-engineering process, and do not aim for creating a global policy, since every domain controls the access to its own data stores. Nevertheless, for consistency monitoring of changing policies at runtime such methods could become important.

In addition to security components for RBAC, Aletheia will also feature basic security measures from WS-security (OASIS 2006).

5 CONCLUSION

In this paper, we identified major issues for designing an access-control solution for information federations using a case study based on the product portfolio of ABB, a large company providing power and automation technologies, products and services for utility and industry customers. Based on this analysis, we applied the scenario-driven role engineering process proposed by Neumann and Strembeck (2002) for tool-supported engineering of tailored inter-organizational AC policies. We presented a security architecture for the enforcement of RBAC policies expressed in XACML, and closed the gap between the development of RBAC policies and their enforcement by creating an export filter that allows the creation of XACML-coded RBAC policies from our graphical models.

Our main contributions are: (i) description of a case study for the scenario-driven engineering and enforcement of RBAC policies in the Aletheia system, (ii) the identification of open issues concerning the engineering of tailored access-control policies for information federations based on a case study in the industrial service sector, (iii) the provision of a practical tool-based methodology for designing roles and AC policies across organizational borders through a central design team with knowledge about all the corresponding processes, and (iv) the design of flexible AC policy enforcement mechanisms for information federations, which is scalable and can cope with frequent policy changes.

The methodical part of our case study is subject to ongoing research and may result in a corresponding extension of the scenario-driven role-engineering process. On the technical side, we will implement, test, and integrate our security architecture into the SOA of Aletheia.

References

Aletheia (2009). Aletheia Project Web Site. URL: http://www.aletheia-projekt.de/.

Ameri, F. and Dutta, D. (2005). Product Lifecycle Management: Closing the Knowledge Loops. Computer-Aided Design and Applications, 2 (5), 577-590.

Anderson, A. (2005). Core and Hierarchical Role Based Access Control (RBAC) Profile of XACML v2.0. OASIS Standard. http://www.oasis-open.org.

Bacon, J., Moody, K., and Yao, W. (2003). Access Control and Trust in the Use of Widely Distributed Services. Software: Practice and Experience (SP&E), 33 (4).

Belokosztolszki, A., Eyers, D.M., and Moody, K. (2003a). Policy Contexts: Controlling Information Flow in Parameterized RBAC. In Proc. of the 4th International Workshop on Policies for Distributed Systems and Networks.

Belokosztolszki, A., Eyers, D.M., Pietzuch, P.R., Bacon, J., and Moody, K. (2003b). Role-Based Access Control for Publish/Subscribe Middleware Architectures. In Proc. of the International Workshop on Distributed Event-Based Systems.

BPMN (2009). Business Process Model and Notation (BPMN) 2.0. URL: http://www.omg. org/spec/BPMN/2.0 Business Process Model and Notation (BPMN) Specification 2.0.

Cohen, E., Thomas, R.K., Winsborough, W., and Shands, D. (2002). Models for Coalition-based Access Control (CBAC). In Proc. of the 7th ACM Symposium on Access Control Models and Technologies.

Coyne, E.J. and Davis, J.M. (2008). Role Engineering for Enterprise Security Management. Artech House.

Dekker, M.A.C., Crampton, J., and Etalle, S. (2008). RBAC Administration in Distributed Systems. Proc. of the 13th ACM Symposium on Access Control Models and Technologies.

Eisenhardt, K.M. (1989) Building Theories from Case Study Research. Academy of Management Review 14 (4), 532-550.

Evdokimov, S., Fabian, B., Kunz, S. (2009). Challenges for Access Control in Knowledge Federations, In: Proc. International Conference on Knowledge Management and Information Sharing (KMIS 2009), Madeira, Portugal.

Ferraiolo, D.F. and Kuhn, D.R. (1992). Role-Based Access Controls. In Proc. of the 15th National Computer Security Conference, 554-563.

Frey, J.H. and Oishi, S.M. (1995). How to Conduct Interviews by Telephone and in Person. SAGE Publications, Thousand Oaks.

Li, N., Wang, Q., Qardaji, W., Bertino, E., Rao, P., Lobo, J., and Lin, D. (2009). Access Control Policy Combining: Theory Meets Practice. In Proc. of the 14th ACM Symposium on Access Control Models and Technologies.

Mazzoleni, P., Crispo, B., Sivasubramanian, S., and Bertino, E. (2008). XACML Policy Integration Algorithms. ACM Transactions on Information and System Security 11 (1), 1-29.

Mendling, J., Strembeck, M., Stermsek, G., and Neumann, G. (2004). An Approach to Extract RBAC Models from BPEL4WS Processes. In Proc. of the 13th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises, Modena, Italy.

Moses, T. (2005). eXtensible Access Control Markup Language (XACML) Version 2.0. OASIS Standard. http://www.oasis-open.org.

Neumann, G. and Strembeck, M. (2002). A Scenario-driven Role Engineering Process for Functional RBAC Roles. In Proc. of 7th ACM Symposium on Access Control Models and Technologies.

OASIS (2006). Web Services Security: SOAP Message Security 1.1 (WS-Security), OASIS Standard Specification. http://docs.oasis-open.org/wss/v1.1/.

Pietzuch, P.R. and Bacon, J.M. (2002). Hermes: A Distributed Event-Based Middleware Architecture. In Proc. of the International Workshop on Distributed Event-Based Systems (DEBS).

Reddivari, P., Finin, T., and Joshi, A. (2007). Policy-Based Access Control for an RDF Store. In Proc. of the IJCAI-07 Workshop on Semantic Web for Collaborative Knowledge Acquisition.

Sandhu, R.S., Coyne, E.J., Feinstein, H.L., and Youman, C.E. (1996). Role-Based Access Control Models. IEEE Computer, 29 (2).

Strembeck, M. and Neumann, G. (2004). An Integrated Approach to Engineer and Enforce Context Constraints in RBAC Environments. ACM Transactions on Information and System Security, 7 (3).

Strembeck, M. (2005). A Role Engineering Tool for Role-Based Access Control. In Proc. of the 3rd Symposium on Requirements Engineering for Information Security.

Wauer, M., Schuster, D., Meinecke, J., Janke T., and Schill, A. (2009). Aletheia – Towards a Distributed Architecture for Semantic Federation of Comprehensive Product Information; IADIS International Conference WWW/Internet.

Yin, R.K. (2003). Case Study Research: Design and Methods, 3rd ed., Sage Publications, Beverly Hills, CA.