Identity as a Service - Towards a Service-Oriented Identity Management Architecture

Christian Emig, Frank Brandt, Sebastian Kreuzer, Sebastian Abeck

Cooperation & Management, Universität Karlsruhe (TH), 76128 Karlsruhe

{ emig | brandt | kreuzer | abeck } @ cm-tm.uka.de

Abstract. Service-oriented architecture (SOA) will form the basis of future information systems. Web services are a promising way to implement SOA enabling the loose coupling of functionality at service interfaces. The focus in SOA changes from traditional software systems to reusable, business-relevant services. Considering the cross-cutting concern of identity management (IdM), it is still an open issue how to construct an SOA-aware IdM architecture enabling "identity as a service" and how to loosely couple the IdM services with SOA's core concern part. In this paper we present a blueprint for a service-oriented identity management architecture featuring interoperability by applying existing standards. Our solution has been tested and evaluated in an implementation case study.

Introduction

Background on Web Service-Oriented Architecture

Currently most enterprises try to align their business processes with the supporting IT by migrating to service-oriented architecture (SOA). Web service technologies are commonly recognized as a promising way for the implementation of SOA; in the following, we focus on web service-oriented architectures (WSOA). With the mutual consent to use WSDL (Web Services Description Language, [1]) for the definition of service interfaces and SOAP (Simple Object Access Protocol, [2]) as the communication protocol, the cornerstone for interoperability is set. Bottom-up approaches start with existing applications and wrap their business functionality to web services. Integration can then be done by composing web services of heterogeneous software systems using process execution languages like BPEL (Business Process Execution Language). Top-down approaches focus business processes and their mapping to composite and basic web services. This allows business analysts to perform "programming-in-the-large", the system-independent orchestration of business-related (web) services along business processes [3].

Motivation for Identity Management in Web Service-Oriented Architecture

Besides the development of WSOA's core concern part there are several cross-cutting concerns that have to be addressed: a central one is to enable security, especially access control. Access control consists of authentication and authorization verification. Looking at the mass and complexity of the existing and upcoming standards in the web service security area like WS-Security, SAML, XACML or the Liberty Alliance's stack proposal it is comprehensible to see software developers often neglect the web service security part. Additionally, state-of-the-art IdM suites are just being prepared for WSOA [4]. As well, current application servers often do not yet support a necessary combination of relevant IdM standards to enable sophisticated access control. This is why as of today existing web services in most cases have little or no security features. Complications even increase when composing several web services which provide functionality from different underlying applications – workarounds like using the applications' built-in IdM are not applicable any more; an overall IdM architecture for WSOA is needed – enabling "identity as a service".

Contributions and Structuring of this Paper

The contributions of this paper are:

1. **The design of a service-oriented identity management architecture**, specified at service interfaces, the implementing components as well as the employed data repositories. The prerequisite is to respect WSOA-specifics like the loose coupling and the existence of basic and composite web services.

2. The alignment of the proposed architecture to existing and promising standards with the goal to **enable interoperability**.

The paper is organized as follows: section 2 introduces the architecture of WSOA and derives the requirements for appropriate IdM services building the bridging point between WSOA's core concerns and the IdM architecture. In section 3 we propose the design of a service-oriented identity management architecture and motivate how to gain interoperability. In section 4 we present our implementation experience. Section 5 treats the related work. A conclusion and an outlook on future work in this area close the body of the paper.

Web Service-Oriented Architecture and Requirements for Identity Management

The basic WSOA "layering" consists of existing applications at the bottom layer that are wrapped to web services, typically using application servers. Web services can be composed at an integration layer using BPEL. Web portals are used to integrate the

(human) users using existing web technology like web browsers. The aforementioned further layers are put on top of the existing applications. Among others, this allows flexible service reuse in different business processes. This common core of WSOA can be found in many publications [5, 6, 7, 8]. It is important to notice that the web service architecture does not imply strict layering. Web services can be accessed either directly or via one or many intermediaries like BPEL engines. From WSOA's viewpoint the service interface of a BPEL-composed web service is not distinguishable from a basic one as they are both described using WSDL.

Before putting (web) service-oriented architectures to fly, there are fundamental questions to be answered: how is access control to be handled in this highly distributed and service-oriented environment? Slicing down existing applications to business related services, the internal IdM structures of the legacy systems are cut off. The alignment of the different system-specific IdM access control models and techniques with the goal to a local handling inside the applications complicates the integrated view on identity management. This is why the development of a WSOA-wide IdM architecture is favored. Being WSOA-aware itself, this infrastructure is meant to expose its functionality at service interfaces decoupling core concerns from IdM, especially access control. Following the paradigm of loose coupling and separation of concerns, the IdM part of WSOA's core concern services should be reduced to the bare minimum [9].

Design of a Service-Oriented Identity Management Architecture

From WSOA's perspective, the complexity of the IdM architecture is encapsulated at a set of service interfaces which should not have business domain-specific characteristics. The central goal of the IdM architecture is to verify authorization for service usage at runtime by enabling access control. Access control is based on two prerequisites: first, an authentication process checking any possible credentials has to be passed. This can be done once with validity for a series of subsequent accesses (relates to a single sign-on approach) or on every access – which is not favored in WSOA as there is usually a significant amount of services to be invoked. User authentication can be initiated at WSOA's portal layer for instance. Second, an authorization verification process is needed which checks if permission has been granted for the authenticated subject to invoke a WSOA service. The functionality of both (i.e. authentication and authorization verification) should be encapsulated at service interfaces featuring "identity as a service" for both basic and composite services. This implies that they simply hand over relevant data to the IdM services for calculation of access control.

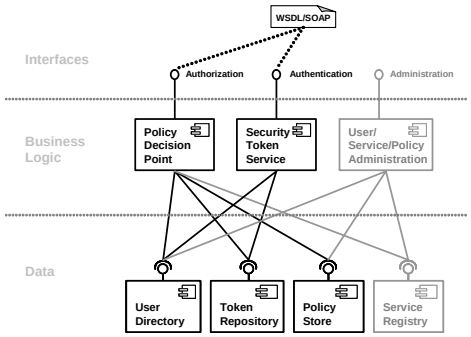

Figure 1. Blueprint of a WSOA-aware IdM Architecture

In figure 1, we present the design of our web service-aware IdM architecture. There are three types of elements that are of interest: first, the IdM services towards WSOA's core concern part and towards administration. Second, the service implementing components and last but not least the data which the components operate on. In the following, we use this structuring to describe the design of our IdM architecture.

Interface Layer

Access control is based on authentication and implies authorization verification which typically are separate processes. Nowadays, access control is typically handled inside the system boundary of an application. In WSOA, the traditional application boundaries are put aside. Instead, web services are addressed exposing applications' core concerns. Following the concepts of "identity as a service", they import all functionality needed for access control using external service invocations.

Authentication is handled at the respective web service interface providing different operations to verify different types of credentials like username / password-based authentication, certificate-based authentication and so on. To enable single sign-on and to enhance privacy, a security token (establishing a session context) with WSOA-wide validity and the possibility for time-limitation is issued on successful authentication. User authentication can be initiated at WSOA's portal layer before accessing protected web services.

Authorization verification is based on an access control model. We have introduced an access control metamodel for web service-oriented architecture in [10] enhancing [11] and [12]. In short, for access control it is relevant to know which user is trying to access which web service operation and what the submitted invocations parameters are, as web services are defined at a high granularity. For the identification of the web service operation, they are all assigned a unique identifier. If a web service operation needs access control, it invokes the authorization verification service sending its identifier, the user's security token and the parameter the user had handed over. Using its internal policy data, the authorization verification service calculates a Boolean value which is returned and enables the web service to either proceed or stop operations.

The third interface of our IdM architecture is an administrative one. It is used to maintain the data as described later. It does not necessarily imply WSDL/SOAP, as administration is often done by humans.

Business Logic and Data Layer

The component implementing the authentication service is the *Security Token Service*. It takes user's identifier and corresponding credentials and does the verification. To protect users' privacy, we suggest issuing temporary security tokens on successful authentication. They are used as opaque handles towards user's identity which is thereby hidden to the core concern web services. Authentication is based on *User Directories*. Here the users, defined by their identifiers, credentials (e.g. passwords or certificates) along with their attributes are stored. The tuples of security token, user identifier and time limitation of the token are stored at the *Token Repository*.

Authorization verification is implemented at the *Policy Decision Point*. It takes the object identifier of the calling core concern operation, the user's security token and the operation's parameters and evaluates them using the corresponding access control policy which is deployed in the *Policy Store*. Here the information according to our WSOA access control metamodel is stored.

Besides identity management, there are further cross-cutting services like the WSOA's service registry. It is important to notice that it is implicitly linked to IdM: the *Service Registry* is used to store web services' descriptions whereas in the *Policy Store* the related policies are put. These two data repositories are linked using the web service operations' object identifiers that are assigned at deployment time. Thinking of relational databases, this identifier is analogous to foreign keys in WSOA's *Service Registry* and as a primary key in the *Policy Store*.

Enabling Interoperability

A major reason for the adoption of web-service oriented architecture is the interoperability which allows best of breed approaches with an easy integration of business functionality. WSDL [1] and SOAP [2] build the cornerstone for interoperability in WSOA, but the same challenges have to be fulfilled at the cross-cutting identity management architecture. We describe the authentication and authorization verification web service interfaces using WSDL and we apply SOAP communication with WS-Security-based message encryption between core concern web services and the IdM architecture [13]. After successful authentication, a SAML-conforming token (Security Assertion Markup Language, [14]) is issued to the user. The user is enabled to use his existing technology like his favored web browser as this token is mapped to a session cookie which is passed between the user's web browser and the web portal. At the portal this session cookie is mapped to a SAML token which is then piggybacked during all WSOA communication. As of now, there is a high interoperability problem if the security token of the user is sent in the SOAP header; this is why we put it in the SOAP body. The token in SOAP header is only

used for point to point message encryption between participating web services. For the *Policy Enforcement Point* at the core concern part we use the design pattern *Secure Service Agent* as described in [15]. It handles the communication with the *Authorization Service* by sending the web service operation's identification, the user's SAML security token and the operation's parameters. We suggest using a relational database for the *Token Repository* and an LDAP-based *User Directory* where all users along with their attributes are stored. The *Policy Decision Point* applies a XACML component (eXtensible Access Control Markup Language, [16]) to verify authorization against XACML policies which are deployed at the *Policy Store*, in XACML contexts preferably using XML files. For WSOA's *Service Registry*, we suggest applying UDDI [17].

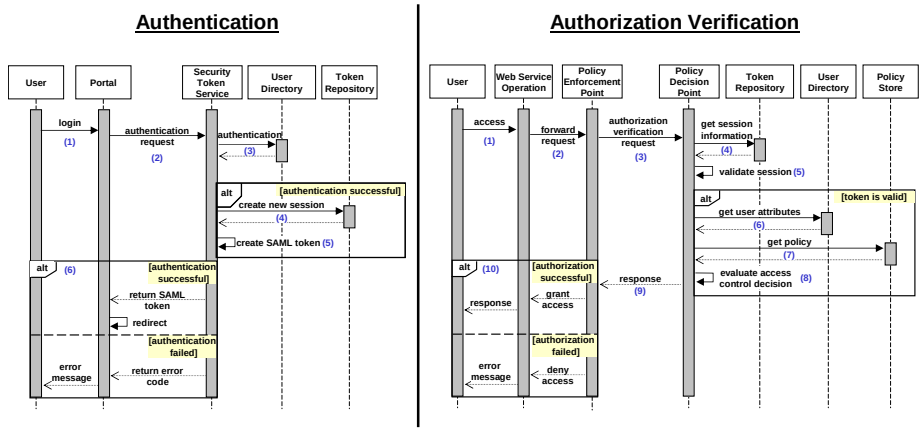

Figure 2. Authentication and Authorization Verification Processes

In figure 2 we depict on the left hand side the process of authentication and its involved parties using UML 2.0 sequence diagrams. Notice the gap between the *Portal* and the *Security Token Service* on the left hand and the *Policy Enforcement Point* and the *Policy Decision Point* on the right hand is the (virtual) border between WSOA's core concern part and the IdM architecture. Issuing security tokens and thereby establishing a session context enables the WSOA for single sign-on capabilities. On the right hand side the authorization verification process is depicted. The *Policy Enforcement Point* is a *Secure Service Agent* deployed once at every application server.

Implementation Experience

We implemented the *Security Token Service* based on OpenSAML 1.1 [18]. The *Policy Decision Point* uses Sun's XACML 1.2 implementation [19]. Both components are realized as Enterprise Java Beans (EJB). To be deployable as web services, we used stateless session beans. The web service communication between core concern web services and the IdM services is encrypted using WS-Security [13]. Recognizing that application servers often do not support outbound encryption using the

requester's key (JBoss, Oracle), we switched to BEA WebLogic 9.2 [20] which supports this necessary feature. The XACML policies are stored as XML files. To increase the performance, in a future release the policies will be stored in a relational database. The *User Directory* is realized using OpenLDAP 2.3. The *Token Repository* is stored in a relational database table using MySQL 5.0. We use SuSE Linux 10.1 as the operating system.

Related Work

There are several papers that address the core concern part of WSOA like the development of web services and their composition neglecting the aspect of access control, like [21, 22, 23]. They admit the necessity of identity management though they do not delve into it. On the other hand there are papers which explicitly address access control, but there they mostly focus on the implementation experience of a specific standard like XACML or SAML [24, 25, 26]. Additionally they lack the integrated view on the IdM architecture by focusing on either authentication or authorization and thereby do not support the concept of "identity as a service".

Conclusion and Further Work

In this paper we presented a blueprint for a service-oriented identity management architecture for web service environments. An authentication service issuing security tokens enables the web services for single sign-on. Our authorization verification service enables separation of concerns – the core concern web services apply access control via this loosely-coupled service. We have done a prototypical implementation securing our existing web services which we have summarized.

Our next steps are to consider a conjoint and model-driven development of web services with their associated access policies. Starting from computation-independent models at the business process level, they can be derived to platform-independent models and transformed to platform-specific models (i.e. IdM architecture-specific) which are effective calculable policies.

References

1. W3C: Web Services Description Language (WSDL) 1.1, March 2001. http://www.w3.org/TR/wsdl
2. W3C: Simple Object Access Protocol (SOAP) 1.1, May 2000. http://www.w3.org/TR/soap
3. Christian Emig, Jochen Weisser, Sebastian Abeck: Development of SOA-Based Software Systems – an Evolutionary Programming Approach, IEEE Conference on Internet and Web Applications and Services ICIW'06, Guadeloupe / French Caribbean, February 2006.

4. Mike Neuenschwander: Enterprise Identity Management Market 2006–2007, Burton Group Identity and Privacy Strategies, November 2006.
5. Ali Arsanjani: Service-Oriented Modeling and Architecture, IBM developer works, 2004.
6. Eric Newcomer, Greg Lomow: Understanding SOA with Web Services, Addison Wesley Professional, December 2004.
7. Object Management Group (OMG): The OMG and Service Oriented Architecture. http://www.omg.org/attachments/pdf/OMG-and-the-SOA.pdf
8. Bernhard Humm, Markus Voss, Andreas Hess: Rules for high-quality service-oriented Architectures (in German), Informatik Spektrum, Volume 29 Number 6, December 2006.
9. Burton Group: Directory Landscape – Directory Products evolve towards Identity Services, Version 1.0, November 2004.
10. Christian Emig, Frank Brandt, Sebastian Abeck, Jürgen Biermann, Heiko Klarl: An Access Control Metamodel for Web Service-Oriented Architecture, submitted for publication, 2007.
11. Eric Yuan, Jin Tong: Attribute Based Access Control (ABAC) for Web Services, IEEE International Conference on Web Services (ICWS 2005), Orlando / Florida, July 2005.
12. D. F. Ferraiolo, Ravi Sandhu, Serban Gavrila, D. R. Kuhn, Ramaswamy Chandramouli: Proposed NIST standard for role-based access control, ACM Transactions on Information and System Security (TISSEC), Volume 4 , Issue 3, p. 224 – 274, August 2001.
13. Anthony Nadalin, Chris Kaler, Ronald Monzillo, Phillip Hallam-Baker (Editors): Web Services Security (WS-Security) Version 1.1, February 2006.
14. OASIS Security Assertion Markup Language (SAML) 2.0, 2005. http://www.oasis-open.org/specs/index.php#samlv2.0
15. Christian Emig, Heiko Schandua, Sebastian Abeck: SOA-aware Authorization Control, International Conference Software Engineering Advances ICSEA'06, Tahiti / French Polynesia, November 2006.
16. OASIS: eXtensible Access Control Markup Language (XACML) 2.0 http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml .
17. OASIS: Universal Description, Discovery and Integration (UDDI) 3.0.2, February 2005. http://www.oasis-open.org/committees/uddi-spec/doc/tcspecs.htm#uddiv3
18. OpenSAML - an Open Source Security Assertion Markup Language implementation, Project Homepage. http://www.opensaml.org
19. Sun's XACML Implementation, Project Homepage. http://sunxacml.sourceforge.net .
20. BEA WebLogic Server® 9.2, Product Homepage http://www.bea.com/framework.jsp?CNT=index.htm&FP=/content/products/weblogic/server/
21. Adam Bosworth: Developing Web Services, 17th International Conference on Data Engineering, 2001.
22. Roy Grønmo, David Skogan, Ida Solheim, Jon Oldevik: Model-driven Web Services Development, IEEE International Conference on e-Technology, e-Commerce and e-Service, 2004.
23. James Pasley: How BPEL and SOA Are Changing Web Services Development, IEEE Internet Computing, Vol.9, Iss.3, p. 60- 67, May-June 2005.
24. Tuncay Namli, Asuman Dogac: Using SAML and XACML for Web Service Security & Privacy, Middle East Technical University, Ankary / Turkey, 2007.
25. Han Tao: A XACML-based Access Control Model for Web Service, IEEE Conference on Wireless Communications, Networking and Mobile Computing, 2005.
26. Yu Peng, Quanyuan Wu: Secure Communication and Access Control for Web Services Container, Fifth International Conference on Grid and Cooperative Computing, 2006.

All URLs last verified: May, 16th 2007.