

Building Blocks for Identity Federations

Johannes Meinecke, Martin Nussbaumer, Martin Gaedke

University of Karlsruhe, Institute of Telematics,
IT-Management and Web Engineering Research Group,
Engesserstr. 4, 76128 Karlsruhe, Germany
{meinecke,nussbaumer,gaedke}@tm.uni-karlsruhe.de

Abstract. Technologies like XML and Web Services have posed new requirements to authentication, authorization and identity management for the Web as an application platform. Beyond merely providing access control for a single isolated system, modern, flexible architectures support a business-spanning federation of applications and services by sharing digital identities. The diversity of today's specifications and the many aspects to be considered, like e.g. privacy, system integrity and distribution in the Web, makes the construction of these architectures a very time-consuming task. Thus, a uniform view on the overall system is needed that abstracts from technological issues. This can be achieved by extracting the core concepts from the emerging Federation technologies and specifications and formalize them to an extent that they can be used as a foundation for configurable applications and services. In this paper, we introduce a solution catalogue of reusable building blocks for Identity and Access Management (IAM). We also present a configurable system that supports IAM solutions in Web-service-based applications.

1 Introduction

Among the many aspects to be considered during the engineering of Web-based applications and systems is the establishment of an adequate Identity and Access Management concept. Today we are faced with a large number of heterogeneous, partly Web-based business applications from different companies that are all in need of access control as well as the related management of identities. Moreover, many products implement their own access control mechanisms, leading to the challenge of managing a company's overall security policy and raising unnecessary costs. Even more difficulties have to be overcome when applications under the control of different organizations are involved. These problems have been recognized [1] and resulted in standardization efforts aimed at the construction of systems that are interoperable in terms of security. The principal idea behind such solutions is to make use of Web service technology to separate authentication and authorization mechanisms from the applications themselves. Thus, federated architectures are realized by sharing identity and access information. In this context, the Security Assertion Markup Language (SAML) has been specified by OASIS as an XML-based notation for exchanging security-relevant information [2]. Furthermore, the Liberty Alliance project is concerned with standardized mechanisms for discovering and offering identity-related

services and applications [3]. Similarly, WS-Federation defines access profiles that describe, how and in which order the messages necessary for a federated authorization process are exchanged between the involved browsers, clients and servers [4]. Because of the high number of different aspects to be concerned, like security technologies, cryptographic algorithms, and communication protocols, solutions based on those standards entail a high degree of complexity, demanding for abstraction.

In this paper, we propose reusable building blocks based on a simple, extendable model, the Federated Identity Model. The building blocks provide solutions to common problems using concepts from security and federation specifications. They have also laid the foundation for the idFS system, an implementation of distributed Web access control that supports the building-block-based approach.

2 A UML-Based Federated Identity Model

The construction and operation of distributed, federated applications that apply the mentioned specifications and technologies demand for a dedicated and comprehensive support. Especially with large identity infrastructures, notations are required to model the structure of the architecture on an abstract level. As one step in this direction, we propose FIM, the Federated Identity Model. FIM is based on our experience and is not intentioned to cover all federation aspects in full. It should rather be seen as a framework that can be extended with additional elements, like for example pseudonym services, and will evolve over time. The model abstracts from the actual technologies in favor of a view centered on potential real-world scenarios. We chose UML to define a set of classes for the modeling elements. As FIM merely serves as the foundation for the focused building blocks, we omit the extensive UML diagram and give a brief description of the most important classes.

Resources are the parts of the system that provide the actual business functionality seen on an application/service level. In the context of federation, they represent the subjects to be shared between the federation partners. Unlike resources, **Security Token Services (STS)** are only auxiliary architecture components providing a safe and federation-enabled identity management infrastructure. Currently, the model comprises two types of STS, which issue the tokens necessary for accessing the resources. In the case of **Identity Providers (IP)**, the tokens concern identities (e.g. “The owner of this token, who signs messages with the cryptographic key A, has the identity B!”). In the case of **Resource Security Token Services**, they contain authorization statements (e.g. “The user with the identity B is allowed to access the Web site C to perform the set of operations D!”).

Within the model, all resources and security token services (collectively denoted as **Federation Nodes**) own a certificate providing a private key for message security¹. Every federation node can make statements in messages that nodes may choose to believe or not, depending on their trust relationships. This association is directed and simply means that one node knows another node by possessing its public key. Hence, it cryptographically trusts certain types of (signed) statements from the other node.

¹ For simplicity, we assume a PKI is used; other cryptographic options are also possible.

The trust relationship plays a central role in FIM as it allows the realization of various trust models like e.g. described in [5].

On a higher level, federation nodes are grouped into **Security Realms**. All systems in one realm are under control of a certain owner, as for example the applications and services at a single company site. This implies that there exists a common role system or other form of access control strategy. If resource security token services are used, only one per realm is required. The establishment of trust between inner nodes is straightforward, as they are operated by the same party. However, two nodes do not necessarily need to trust each other just because they belong to the same realm.

Additional modeling classes represent abstract modules and user interfaces that can be seen as separate system parts, with the configuration specifying the concrete algorithms and mechanisms to be used (like e.g. **Authentication Mechanisms**, **Authorization Mechanisms**, **Delegation Mechanisms** or **Management Interfaces**).

3 Building Block Catalogue

The model presented in the previous section took a rather general approach, allowing for the description of a wide range of solutions. This also included very conventional systems, as for example a monolithic Web application with built-in authentication and authorization mechanisms, and an integrated account management interface for administrators. In the context of the mentioned identity management problems however, guidelines are required for building solutions in a high-quality and cost-effective way. Therefore, we present a catalogue of building blocks based on FIM. Similar to design patterns in Software Engineering [6], it contains a statement of a certain problem together with a description of how to solve this problem as well as a discussion of the solution. In the following, we describe a selection of four building blocks out of our catalogue.

Single Sign On (SSO): An organization runs several Web applications that require access control. Integrating IAM components into the applications would result in high management costs and inconveniences for the users who would have to remember all their credentials.

Corresponding to the approaches described in section 2, an IP and a resource STS is set up. The Web applications are all configured to send users requesting protected resources to the STS by performing an HTTP redirect. The STS redirects the user again, now to its configured IP that displays a sign-in form. In case valid credentials have been supplied, a security token is generated and passed on to the STS. The STS determines the permissions for the identity stated in the incoming token, generates a new token and redirects to the original application. When the user accesses one of the other federation-enabled applications within the same session, the IP will not have to show the sign-in form again, as it has already authenticated the user and the issued token is still available as session data. The separation of the authentication and authorization processes from the applications allows for reusing accounts and access policies. Any redundancy of identity information is avoided, which lowers the cost of management. A central STS enables the organization to define a uniform role system for all connected applications. Once the infrastructure exists, new applications can easily be integrated by just configuring them for the use of the STS.

Self-Service Identity Management: The operation of large, Web-based systems with many different users causes high administration costs. A vast amount of effort is spent on identity-related tasks like creating user accounts for new employees or resetting passwords.

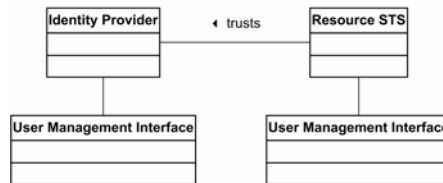


Fig. 1: Self-Service Identity Management

To reduce management expenses, the tasks related to accounts and access policies are delegated to the users themselves as far as possible. Both IP and STS are equipped with management interfaces that are used by the account holders in a self-service manner (cf. Fig. 1). For the IP, this means, enabling users to make changes to their own accounts, like resetting passwords and changing contact details. Even the creation of accounts can occur without any dedicated staff involved, e.g. with online registration forms for anonymous users. In case of the STS, a self-service authorization mechanism can allow account holders to enter *activation codes* in order to put themselves in certain roles and, in a way, authorize themselves as foreseen by the business process. An obvious restriction to the self-service concept concerns the security aspect. Anonymously created accounts or role administration by users may not be suitable in high-security zones, but in other areas, close to 100% self-operating solutions can be achieved.

Distributed Web Service Access Control: An organization operates a service-oriented architecture (SOA) with a large number of Web services. These need to be partly available publicly, whereas some of their methods are only intended for a limited group of callers. The policy stating who is allowed to access which functionality changes frequently. If this policy was wired into the services statically, the resulting maintenance would raise unnecessary costs.

The organization may already have set up an IP and an STS for their Web applications. Similarly to the Single Sign On building block, the Web services are configured to trust a central STS, which itself trusts a central IP. Any accessing program has to first call the SOAP interfaces of the IP and the STS to receive the necessary security tokens. For the duration of the token validity, applications can then attach these to all of their SOAP requests. Hence, a standardized way of accessing the organization's Web services is provided. When implementing this relatively complex approach to secure Web services, the impact on performance has to be considered carefully. In the worst case, the number of Web service calls is tripled, with each call requiring time-intensive cryptographic operations. However, the solution brings with itself a lot of advantages regarding flexibility. The same infrastructure that is already applied to handle application access control is reused. Thus, it is possible to treat individual programs and their owners with the same technical concept.

Identity Federation of Enterprises: Separate enterprises want to cooperate by the interconnecting of users, applications and systems. This includes problems for the arrangements of transcendental access to distributed processing and data sharing.

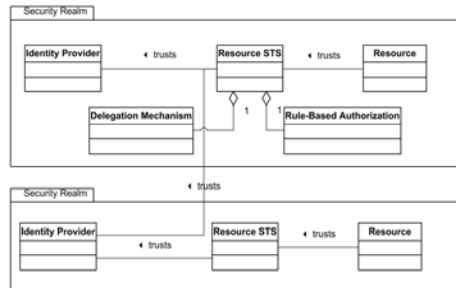


Fig. 2: Identity Federation of Enterprises

As a solution, the security infrastructures of the partners are linked up by connecting their security realms, for example as depicted in Fig. 2. Trust relationships are established between the STS of the realm where resources need to be accessed and the identity providers of the realms where the accessing users have their accounts. When a user accesses a resource, the responsible STS dynamically allocates the proper identity provider based on the rules of a delegation mechanisms. When the security token returns back from the IP to the STS, it is quite possible that the stated foreign identity is unknown in this realm. Therefore, an authorization mechanism should be applied that issues permissions depending on rules. Although the identification process is delegated to an external system, the partners are still in full control of their access policies. External users do not receive permissions unless this has been explicitly stated at the STS. Users of a federation partner do not need an extra account for the external sites and only sign in once at their enterprise. The full potential of this solution relies on the fact that the work of the users transcends organizational borders, as business processes demand.

4 System Support by idFS

As a technological infrastructure for supporting Web-service-based architectures in correspondence to the Federated Identity Model (FIM), we developed the Identity Federation System (idFS). It consists of configurable components that offer the necessary SOAP and Web application interfaces for identity providers, security token services and protected Web resources. As the underlying technological platform, the .NET Framework was used, which already offers some support for WS-Security. The FIM building blocks can be realized by configuring solutions for the use of different security token formats, protocols and security token services. In the current version, these underlying federation concepts are implemented with the standards WS-Federation and SAML. This allows e.g. to establish Single Sign On (SSO) in compliance with the WS-Federation specification, with SAML being used as the security token format. The screenshot in Fig. 3 shows the application of idFS in a real world scenario at the University of Karlsruhe campus, demonstrating the combined use of two federation concepts: SSO and self-service identity management. idFS can be downloaded at <http://mwrq.tm.uni-karlsruhe.de/downloadcenter/>.

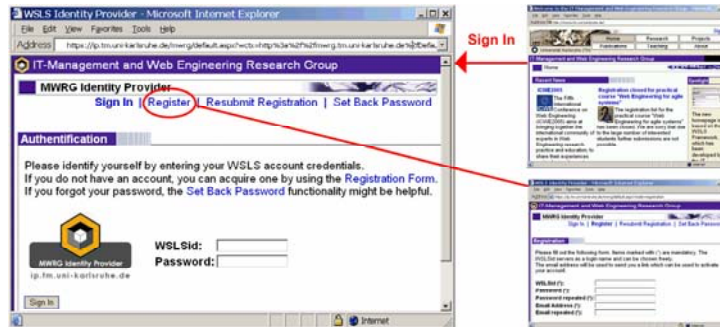


Fig. 3: Sign-in form at the idFS identity provider.

5 Conclusion and Future Work

In this paper, we looked at various concepts of federated identity management. Founded on the model FIM, we presented a catalogue of building blocks that can be applied to support scenarios which take full advantage of the federation idea. Early experiences with the implemented supporting system idFS led already to very promising results: the maintenance effort of student accounts was practically eliminated, existing heterogeneous account systems of the university could be integrated, and the participation of legacy applications and services in the research group's partner network was achieved almost without complications. We are now working on a shift towards a more agile approach to configuration, which focuses the whole life cycle of all interacting federation nodes.

References

1. Witty, R.J. and Wagner, R.: The Growing Need for Identity and Access Management. 2003: Stamford, CT
2. Maler, E., Mishra, P., and Philpott, R.: Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V1.1 - OASIS Standard (2003): <http://www.oasis-open.org/specs/> (18.10.2004)
3. Liberty Alliance Specifications - Web site (2004), Liberty Alliance Group: <http://www.projectliberty.org/resources/specifications.php> (18.10.2004)
4. Bajaj, S., et al.: Web Services Federation Language (WS-Federation) - IBM (2003): <http://www-106.ibm.com/developerworks/webservices/library/ws-fed/> (14.10.2004)
5. Linn, J., et al.: Trust Models Guidelines - OASIS Working Draft (2004): http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security (19.01.2005)
6. Gamma, E., et al.: Design patterns: elements of reusable object-oriented software. Addison-Wesley professional computing series. 1995, Reading.: Addison-Wesley. xv, 395