Federation of Services from Autonomous Domains with Heterogeneous Access Control Models

Abdramane Bah[1,2], Pascal André[1], Christian Attiogbé[1], and Jacqueline Konate[2]

[1] LS2N CNRS UMR 6004
University of Nantes, France
[2] FST-USTTB
University of Science and
Technology of Bamako, Mali
{firstname.lastname}@univ-nantes.fr, jacqueline.konate@gmail.com

Abstract. Service-oriented architectures implemented by web services technologies provide standardized protocols for communicating and sharing information across organizational boundaries. The access control of shared services becomes an essential requirement for a secure federation of services. The identity federation provides part of the response by allowing users to authenticate once in an organization and to access the services of others with its authorization information or attributes. However, in a federation, the organizations may have different access control models and authorization attributes with different or even incompatible semantics. Interoperability between the access control models becomes crucial to the federation of services. Existing federated access control solutions are based on the single sign-on with common authorization attributes or the identity mapping that is not scalable in a service-oriented environment. In this paper, we propose a cross-organizational access control method for the federation of services protected by heterogeneous access control models. Our method is based on a new federation architecture that responds to the heterogeneity of authorization attributes via independent attributes introduced at the federation level.

Keywords: SOA, Service Composition, Federation, Access Control, Attribute Mapping, Federated Single Sign-On

1 Introduction

Service Oriented Architecture (SOA) implemented through web service technologies provides standardized protocols to utilize distributed services under the control of independent security domains [1]. A *security domain* or domain is an autonomous security administration unit that includes services, users, and security policies to manage user access to services [2][3]. In SOA, the resources of a domain are service-oriented. The federation of domains compliant with SOA makes it possible to leverage independent domain business services as part of a composite application called *federation of services*, in order to quickly achieve common goals (for example, improve productivity, create new business value) [4]. A *federated service* is a service shared by a federated domain and accessible to authorized users of the federation.

The access control are based on authorization models such as Attribute-Based Access Control (ABAC) model, Role-Based Access Control (RBAC) model [5]. ABAC defines access permissions with a set of boolean rules specified in terms of the attributes assigned to the subjects (e.g. user, application, service) and objects (e.g. service) and environment conditions [6]. All the access control models can be transformed into ABAC [5]. In the case of RBAC, the role is considered as an attribute. The subject attributes that can be considered in the access decisions such as user's role are here called *authorization attributes*. A consistent definition of the subject attributes allows a domain *B* to grant access to the subjects of a domain *C* without requiring prior registration of their identities in *B* [7]. The authentication and authorization of users can be performed and administered in separate domains, while maintaining the appropriate levels of security. The identity federation enables users to authenticate once in their domain (home) and to access the services from other domains (target) based on their authorization attributes obtained in the (home) domain [8]. However, the federated domains can have different authorization attributes with different or even incompatible semantics. This can lead to unauthorized access to the shared services. It becomes crucial to overcome the heterogeneity of domain authorization attributes for a secure federation of services.

Current federation solutions such as Shibboleth, WS-Federation [9] utilize two main approaches to address the heterogeneity of domain authorization attributes: (1) the standardization of authorization attributes; (2) the mapping of domain authorization attributes. In the first approach, the federation imposes to the domains its authorization attributes called here the *federated attributes* (e.g. the shibboleth eduPerson LDAP schema, Renater's SupAnn schema) based on which the access control policies of the domains are specified. Although the domains retain control over the security of their services, their security policies become tightly coupled with the standards of the federation. In the second approach, domains negotiate mappings between their authorization attributes. In spite of its point-to-point nature and the inconsistency of domain authorization attributes (e.g. different role concepts), this approach requires the disclosure of information about security policies, such as business roles, information on the security infrastructure considered in [10] as information leakage of security policies.

In this paper, we propose a federated access control method based on a new federation architecture that allows loose coupling between the domains and the federation in terms of access control. Our method is based on the mapping technique using the federated attributes to address information leakage of security policies. The benefits of service-oriented architectures such as agility are achieved through the composition of services. The heterogeneity of the authorization attributes of service providers is a major obstacle to the secure composition of federated services. Our method supports access control of the service composition.

The rest of the paper is organized as follows: the Section 2 introduces the basic concepts of the federation of services, as well as the challenges and limitations of existing federated access control solutions. The Section 3 describes in detail our access control method. The implementation of the proposed method is described in the Section 4. The Section 5 presents the evaluation of our method applied to a case study. Related works are presented in Section 6. We end with the conclusion in Section 7.

2 Federation of Services

SOA is an approach to organize distributed resources as autonomous and remotely accessible units of functionalities called services [11]. Services are discoverable and accessible to end-user applications or other distributed services via standard message interfaces and protocols. The main SOA principle is: the *service provider* hosts and executes the service on the behalf of the *service consumer* which discovered the service description in the *service registry*. *Web services* provide a standard-based implementation of SOA accessible through internet protocols such as HTTP. A web service is a self-describing, self-contained software component that can perform actions on behalf of a user or application [12]. Web services rely on standard protocols such as SOAP and WSDL for the description of the service interface and communication messages. Web Services enable the creation of distributed applications that can be dynamically assembled by composing existing services as needed.

Each service is located in a (security) *domain*, including security authorities and governed by a security policy. A common way to achieve interoperability between domains is to federate them. A *federation* is a set of autonomous domains that adhere to common rules and governance policies to control interactions between them [13]. The federation creates a trusted environment for the secure sharing of services between domains. Access control is a security mechanism to ensure that only authorized users have access to resources (considered as services here). Access control starts with the authentication of users and then checks their authorizations. The federation allows users of one domain to access the services from other domains where each domain is assumed to be independent meaning it has its own access control model. To facilitate the management of identity and authentication of users, which can be numerous, identity federation allows users to authenticate only once in the domain they belong to and access the services of others using a single identity. The domain that provides the identity is called the *identity provider* (IdP) and the domain that use this identity to provide the services is called service provider or *relaying party* (RP). The users authenticate with IdPs who create and transmit the proof of authentication to the RPs as a security token. A *security token* represents a set of *claims* that are declarations made by a third party about the user's identity attributes, such as his name, and his authorization attributes such as his role. Access control in domains (service provider) is based on these authorization attributes. The exchange of security tokens between IdP and RP allows the *federated single sign-on* (FSSO) between the domains. The security tokens are described using the *Security Assertion Markup Language* (SAML) to ensure interoperability between domains. A domain can ensure both the role of IdP (service consumer) and the RP. The federation of services allows to create distributed applications using the services provided by the domains of a federation. Given the decentralized access control at the domains levels, the federation of services remains a major challenge [14] [15].

The access control consists of two essential steps: (i) identification and authentication of users; (ii) authorization of users. The authentication of users is delegated to IdPs through identity federation. The authorization of users remain under the control of RPs.

However, the latter depends on federation architectures, the main ones being Shibboleth, Liberty Alliance, and WS-Federation [16][9].

With Shibboleth, IdP and RP agree to use common authorization attributes whose semantics are defined through LDAP schema such as the *eduPerson* schema. The access control policies of the RPs are defined on these attributes. Shibboleth also allows to IdPs and RPs to map their own authorization attributes on those of the standard schema. But, these attribute mappings are managed by each IdP and are therefore unreliable. For example, when the attribute teacher does not have the same meaning for two different universities, one of which (IdP) considers a PhD student as a teacher and the other (RP) as a student. This may result in unauthorized access. With Liberty Alliance, the user has distinct identities with different IdPs and RPs that are connected for authorization.

WS-Federation supports Shibboleth and Liberty Alliance access control techniques through specialized services such as *authorization service*, *attribute service* and *pseudonym service*. WS-Federation also provides identity mapping solutions that consists of converting an identity of one domain into an identity in another domain by a trusted third party [2]. However, these identity mappings solutions are not flexible enough because they require point-to-point negotiations between each pair of domain. The access control of service composition requires authorization negotiations going beyond two domains. The federated services access control requires a federation architecture that supports authorization negotiations for service composition.

The federation of services faces major challenges: (i) Heterogeneity of domain authorization models. Each domain specifies its access control policies on its own authorization attributes such as role. When domains use authorization attributes with different or incompatible semantics, access to services is either hindered or granted to unauthorized users. (ii) Autonomy of domains. One domain may belong to different federations or collaborate in pairwise way. In any case, the context must not interfere with the local security policy. (iii) Composition of federated services. The composition of federated services must take into account the access control of each service and therefore the heterogeneity of domain authorization attributes. The secure federation of services from independent domains must meet the following requirements:

– *Federated single sign-on*. A user must be able to authenticate once to the federation and then use the services for which he has a valid authorization.
– *Decentralized authorization*. Users must acquire authorizations from their domains and access the services on the basis of these authorizations.
– *Autonomy of domains*. Each domain controls the access to its services.
– *Dynamic adaptation to the federation growth*. The domain's access control mechanisms should not require significant maintenance efforts during authorization changes in the federation.
– *Confidentiality of internal security informations*. The authorization attributes are sensitive informations and should not be disclosed beyond the domain boundaries.

In the next section, we propose an access control method that addresses these needs.

3 A Method for Federated Access Control

Our method is based on a specific federation architecture adapted from current practices to support the attribute mappings.

Federation Architecture. The domains are federated by considering that the services in one domain are accessible to other domains based on the trust relationships and access control policies. Cross-domain access control requires that the authorization attributes of a domain be understandable in other domains. The access control policies of domains are specified on their authorization attributes. Domains map their authorization attributes to prevent their access control policies from being dependent on the attributes of other domains. The attribute mappings serve as a means of granting permissions to users outside the domain using only the domain authorization attributes. In order to establish flexible mappings between domain authorization attributes while avoiding information leakage of security policies, we introduce an independent entity called *Global Access Control Mediator (GACM)* at the level of the federation. The GACM serves as a trusted third party between domains. The domains no longer need to negotiate access authorizations with each other (plain arrows in Fig. 1), they only need to negotiate access authorizations once with GACM (dashed arrow in Fig.1) and then access domain services directly with these authorizations as shown in the Figure 1. The main interests of GACM are on the one hand to ensure the secure granting of authorizations to users outside the domains and on the other hand to facilitate the management of trust between domains. The readers can access a detailed introduction in our reserach report [17].

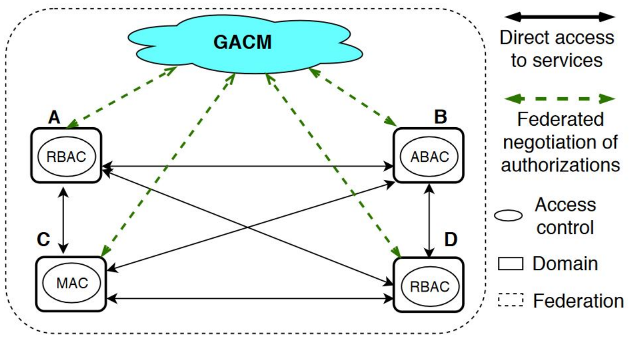
GACM
A
RBAC
B
ABAC
C
MAC
D
RBAC
Direct access to services
Federated negotiation of authorizations
Access control
Domain
Federation

Fig. 1: The proposed federation architecture

Attribute Mapping through the GACM. GACM does not define domain access control policies or grant access permissions to users. It represents the federation authority serving as a bridge between the domains. The primary purpose of attribute mapping for domains is to understand the authorization attributes of each others in order to determine the local access permissions for the external authorization attributes. However, the federation evolves; new domains join it with new authorization attributes and others leave.

Attribute mapping must dynamically adapt to the evolution of federation and authorization attributes changes in domains. To achieve these objectives and avoid leakage of security informations, the GACM defines the authorization attributes of the federation, the *federated attributes*, independently of those of the domains. Federated attributes are public and understandable by all domains.
We define mappings between federated attributes and domain authorization attributes at two levels as shown in Figure 2:

- At the GACM level: the domains negotiate once with the GACM, the mappings between their authorization attributes and the federated attributes. This first mapping, called the *federated mapping* is registered with the GACM and a copy is registered in the domains;
- At the domain level: each domain locally defines mappings between the federated attributes and its authorization attributes. This second mapping is called the *domain mapping*.

Interactions between domains are then performed using federated attributes. Using federated attributes, domains can grant access authorizations to all other domains of the federation without knowing their local authorization attributes. As a result, domains can access one another's services despite the heterogeneity of their authorization attributes. The advantage provided by our approch to the users is to obtain the access authorizations in other domains based on their original authorization attributes.

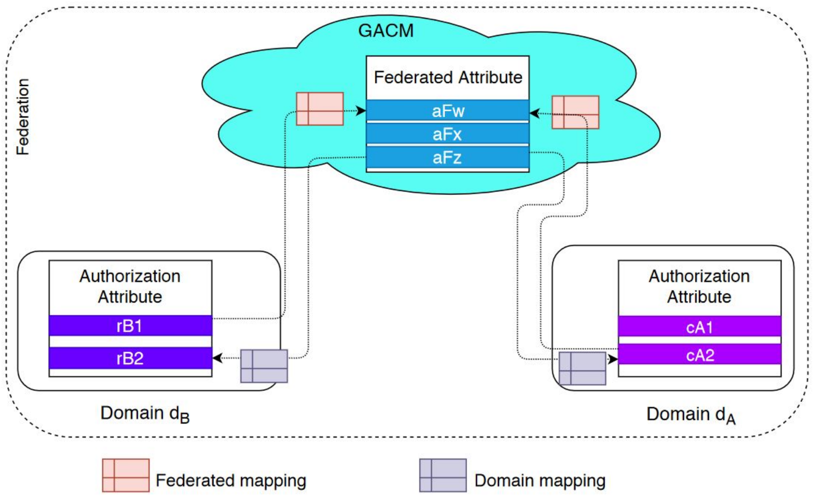

Fig. 2: Federated attribute for the mapping of domain authorization attributes

Now, the services of domains accessible in the federation are seen as federated services.

3.1 Access to Federated Services

We now present how to access the federated services.

Authentication and Trust Brokering. The access control of services relies on the authorization attributes of users asserted by a trusted third party. Each domain has its own authentication mechanism called *local token service* (LTS). The LTS authenticates users and issues a security token signed by the domain security certificate. The services of a domain are accessible only with a security token issued by the domain's LTS.
In order to establish trust between domains, we introduce in the GACM a specialized authentication mechanism called *federated token service* (FTS) for domain authentication. We identify the domains and the GACM with the public-key certificates. The security certificates of domains are forwarded to the GACM which in turn transmits its certificate to the domains. The domains authenticate to the GACM with the security tokens signed with their security certificates. In response, the FTS delivers the security tokens signed by the GACM's security certificate. Consequently, the domains of the federation trust each other through the *federated security tokens*.
As shown in Figure 3, to access to a service (S_B) of domain B (d_B) from a domain A (d_A), the authentication of the user (U_A) is performed with the following steps :

1. the LTS of d_A authenticates U_A and delivers an security token (ST_A) signed with the d_A security certificate ((1.a) dashed arrow in Fig.3);
2. d_A authenticates to the FTS using ST_A and obtains on behalf of U_A, a federated security token (ST_F) signed with the GACM certificate (1.b);
3. the service consumer use ST_F to obtain a security token (ST_B) from d_B signed with d_B security certificate. The ST_F signature proves that d_A and U_A belong to the federation and are trustworthy (1.c).
4. finally, S_B is called on behalf of U_A with ST_B (1.d).

The authorization attributes contained in the ST_B being specific to d_B, are used for the access control of S_B.

Authorization. The security token used to invoke a service must contain the authorization attributes of the domain providing this service. The user initially has the authorization attributes of his domain that must be successively mapped to the federated attributes and the target domain's authorization attributes during the authentication process. We assume that the federated mapping and domain mapping discussed in Section 3 are already established.
In the Figure 3, we illustrate the attribute mapping by considering the steps presented in 3.1. To achieve the authorization of U_A, the authorization attribute of U_A (*cA1*) is used by the FTS to compute the federated attribute *aFx* corresponding to cA1. The aFx sent to d_B, allows the d_B's LTS to compute the authorization attribute *rB2* corresponding to aFx. This latter allows finally to access the service targetted by U_A.

3.2 Composition of Federated Services

Each composed service has its own authorization attributes requirements. The access control of the service composition is done at two levels: the composite service's access control and the composed services's access control. This creates two additional issues: (1) the specification of the composite service's access control requirements; (2) and the

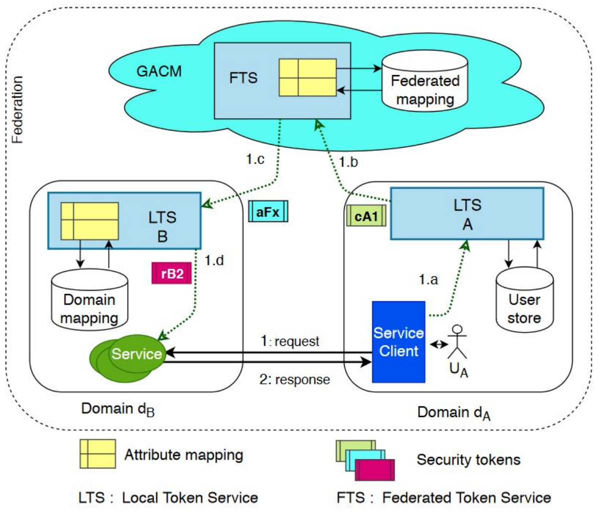

Fig. 3: Sequence of federated Single Sign-On and cross-domain authorization

federated single sign-on between the composite service consumer (intial requester), the composite service and the composed services. We solve these issues by considering two scenarios: (i) we invoke the composed services on the behalf of composite service; (ii) we invoke composed services on the behalf of the intial requester.

In the first scenario, the access control of the composite service is performed like in any federated service. The access control requirements of the composite service are independent of those of the composed services. To invoke a composed service, the composite service follows the authentication steps described in the Section 3.1.

In the second scenario, the composite service's access control requirements depend on those of the composed services that may be different from one service to another. The composed services require a security token containing the authorization attributes of their domains. The composite service consumer must provide a security token that satisfies these requirements.

For this purpose, first, we introduce the *token store* at the composite service level to store the security token of the initial requester (ST_{init}). The composite service must convey the ST_{init} to invoke the composed services. But, the ST_{init} contains the authorization attributes of the domain that provides the composite service. Second we perform a new authentication process using the ST_{init} in order to have the authorization attributes of the composed service's domain corresponding to those contained in ST_{init}. Figure 4 illustrates the service composition with this scenario.

4 Implementation of our Method

The goals are to develop the required software modules; to select and customize existing security services to support our access control method.

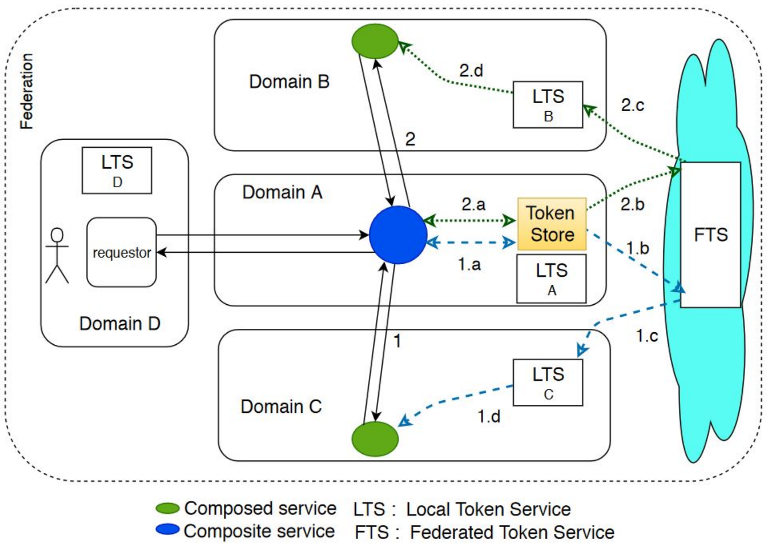

Fig. 4: Invocation of composed services on the behalf of the initial requester

WS-Trust, WS-SecurityPolicy and WS-Security provide the basic model of the federation of web services [18]. WS-Trust is implemented with *Security Token Service* (STS) that provides methods for issuing, validating, transforming, and renewing security tokens. The LTS of domains and the FTS of GACM are implemented with WS-Trust STS. We have three types of STS in our architecture: the STS in the services providers domains (named STS_{SP}), the STS in the services consumers domains (named STS_{SC}) and the STS of the GACM named STS_{GACM}. The implementation of our method involves four steps.

Step 1: *Definition of claim dialect of federated attributes*. The security requirements of federated web services must be specified using the federated attributes defined by the GACM. However, WS-SecurityPolicy does not define a claim dialect for the expression of claim requirements. We define a claim dialect (XML schema) to describe the federated attributes. Each web service specifies its authorization attributes requirements using this claim dialect.

Step 2: *Definition of federation-specific security requirements of STS_{SP} and STS_{GACM}*. First, the target web service requires a SAML[3] token issued by the STS_{SP} of its domain with specific claims. The STS_{SP} requires a SAML token issued by the STS_{GACM} which also requires a SAML token. The STS_{GACM} does not specify the token issuer because it trusts all STS_{SC} in the federation.

Step 3: *Implementation of attribute mapping of STS_{SP} and STS_{GACM}*. The STS_{SP} and the STS_{GACM} are customized in order to implement the attribute mapping. These STS must be able to retrieve authorization attribute (claims) contained in the SAML tokens and exchange them with the corresponding attributes stored

[3] Security Assertion Markup Language (SAML) OASIS Standard

in the *mapping module* that contains the pre-established attribute mapping. We implement the mapping module with a relational database to be queried in order to easily find the desired attributes.

Step 4: *Implementation of service's access control enforcement*. Web services access control is based on XACML[4] which has several logically distinct components, including the *Policy Enformcement Point* (PEP) and the *Policy Decision Point* (PDP). The PEP intercepts the SOAP request, extracts the authorization attributes contained in the SOAP message header and enforces acces decision made by the PDP. We assume that domains already have access control policies defined on their local authorization attributes. As a result, the existing PDP are maintained. But, we implement the PEP with *Apache CXF*[5] *interceptors*.

Web services access control requires the composition of several security standards, namely WS-Security, WS-Trust, SAML, XACML and WS-Federation [19] [20]. SAML and XACML are implemented independently. But WS-Federation, WS-Trust, WS-Security are dependent. The deployment of WS-Federation depends on the those of WS-Trust which depends on WS-Security forming thus the layers of security protocols. This dependency is difficult to deal with because there are no solutions that deploy these layers together. The web service developers are constrained to deploy each of its layers separately. What is likely to generate configuration errors. The main web services development solutions providing the security layers are *Apache CXF*, *Axis2*[6], *Glassfish Metro*[7] for those that are compatible with *JAX-WS*[8] specification and *Microsoft's WCF*[9]. These solutions do not directly integrate the access control (XACML and SAML). Solutions that integrate access control such as *WSO2 Application Server*[10] are not customizable. Finally, the implementation of web services access control becomes quickly a real challenge.

5 Application and Evaluation

We present a case study on which we experiment the proposed method.

5.1 Case study: Federation of Scholarship Services

The case is a federation of three institution systems involved in the payment of students scholarship. Initially, the scholarships were paid by the national treasury but three independant higher education institutions are responsible to grant the scholarship to students: the Center of University Studies (*CUS*), the Directorate of Higher Education (*DHE*) and Universities. The usual scholarship is allocated by CUS. An additional aid

[4] eXtensible Access Control Markup Language (XACML) OASIS Standard
[5] Apache CXF, https://cxf.apache.org
[6] Axis2, http://axis.apache.org/axis2/java/core/
[7] Metro web service stack, https://javaee.github.io/metro/
[8] Java API for XML-Based Web Services, Sun Microsystems, Inc.
[9] Windows Communication Foundation
[10] https://wso2.com/products/application-server

is allocated to disabled students by the DHE. Some universities grant on their budgets an aid to the non-scholarship students. The Treasury pays the scholarships and the various aids for the account of each institution that dispatches them to students. To this end, each institution establishes and submits to the treasury a scholarship payment card consisting of a set of attribution codes (*sc-code*) and their amount. Each sc-code represents a student's scholarship. In order to facilitate the payment of scholarships, the decision is taken that all payments should be now made by the accounting departments of universities. To put into practice, the CUS, DHE and universities decided to federate their systems to share securely the scholarships attribution codes. The CUS provides the sc-codes of the usual scholarship. DHE provides the sc-codes of disabled students aids. The Universities collect the sc-codes of their students at CUS and DHE to establish their payment cards. The scholarship of a disabled student is the sum of his sc-codes. The table 1 describes the different domains that will participate in the federation.

Table 1: Description of the domains to be federate

Domain	A.C. model	Authorization attribute (role)	Web services
CUS	RBAC	financial-officer, accounting-officer, chief-accountant	scholarshipService
DHE	ABAC	cashier, accountant	disabled-grantService
UTS	RBAC	administrator, financial, accounting-secretary	

As described in Table 1, the CUS and DHE are the service providers and the universities such as University of Technical Sciences (UTS) are the consumers of these services. Each domain has its access control model (A.C. model) with its authorization attributes that are the roles in the RBAC model. The role is considered in the ABAC model as an attribute. To create its scholarship payment card, the UTS accounting department must access the web services of the CUS and DHE. This requires establishing trust between their systems and the interoperability between their access control models.

5.2 Federation of Domains and Services

To federate the CUS, the DHE and the universities and their services, we follow the steps described in the Section 4.

Step 1 - Definition of federated attributes and the claims dialect. An autonomous department of DHE, the Department of Administrative Affairs (DAA) is designated to host the GACM. A security certificate is created for the DAA, CUS, DHE and all universities belonging to the federation. DAA registers the security certificates of the domains. The CUS, DHE and universities also register the DAA security certificate. The DAA and the domains of the federation can now trust each other. The DAA defines the federated attributes as shown in the Table 2 and creates the claim dialect to describe them. The DAA negotiates with the domains to establish the federated mapping. The CUS and the DHE establish their domain mapping. The DAA deploys the federated token service, the STSDAA. It is assumed that the domains already have their STS. Otherwise,

Table 2: The DAA federated attributes

	Authorization attribute (userAffiliation)
Finance	finance-director, finance-assistant, finance-secretary, ...
Administration	administration-director, administration-adjt, ...
Information technology	it-administrator, ...

the domains install their STS. The STS of the UTS is configured to support the claims dialect.
Step 2 - Definition of STS security requirements. The CUS and DHE specify the access control requirements of their web services and their STS using the DAA claims dialect.
Step 3 - Implementation of attribute mapping. The DAA, CUS, and DHE create a database to store the federated mapping and the domain mapping respectively. Their STS implement the attribute mapping with a generic software component to query the mapping databases.
Step 4 - Web services access control. The CUS and the DHE deploy SOAP message interceptors to extract the authorization attributes from the SAML assertions and enforce the services access decision.
After these steps, the UTS and other universities can then access the web services of the CUS and the DHE to collect the students sc-codes and create their scholarship payment card.

5.3 Evaluation

We evaluate our service federation architecture based on the following criteria:

– *Applicability*: the ease of implementation and integration into an existing security environment;
– *Scalability*: the adaptation to the evolution of the federation and changes of the authorization attributes in the domains;
– *Reliability and security*: The reliability of attribute mappings for granting access permissions to external users.
– *Extensibility*: the support of others access control models different from ABAC.

Applicability. For example, in Section 5.2, when an university —using an LDAP registry with OpenAM[11] as the authentication mechanism and RBAC as authorization model— participates in a federation built according to our method, the existing security mechanisms (LDAP, OpenAM and RBAC) are maintained. OpenAM is configured to support the dialect of the federation. Internal roles used for the authorizations are never disclosed to CUS and DHE. This reduces the dependencies between domains for the access control. The only change in CUS and DHE is the implementation of an STS in order to support the attribute mapping. Our architecture fits well with existing access control mechanisms of domains and its adoption requires minimal configuration efforts.

[11] ForgeRock OpenAM, https://backstage.forgerock.com/docs/openam

Scalability. The evolution of the federation has no effect on the access control of the CUS and DHE because of the stability induced by the incoming mapping of the domains. The domains (service providers) adapt themselves only to the evolution of the federated attributes.
Reliability and security. The federated attributes of a user is asserted by the GACM through the federated mapping. This ensures the reliability of the authorizations granted by the service providers through their domain mapping.
Extensibility. Our approach assumes different access control models in each domain of the federation. Since all access control models can be transformed into ABAC [5] and the mappings rely on the authorization attributes, our method supports other access control models.
However, our approach focuses more on the access control of external users to the domains. In the case where the federated service is also used within the domain, the access control of the service will always use the GACM. Internal use of the federated service requires a new service contract that does not employ the GACM.

6 Related Work

Jasiul et al. [21] present an analysis of authentication and authorization challenges for users and services in federated SOA environments. They identify SOA-specific requirements for federated access control and argue that only cross-domain, trust-based authentication and authorization can meet these requirements. They recommend the federated single sign-on (FSSO) mechanisms to avoid overloading security services and identity propagation to secure the service composition.
In a federation, users belong to different security domains. Two approaches are generally used to perform federated authentication: a central identity provider, a federation of local identity providers. The authors in [4] compare both approaches to illustrate the benefits of federated identity systems. For interoperability between heterogeneous access control models in the federation, Hafeez et al. [5] propose to transform the domain access control models into ABAC using XACML and apply the resulting policy to remote requests. In our approach, we assume that domain access control models are defined using XACML. Our challenge is rather the heterogeneity of domain authorization attributes.
Several solutions to the heterogeneity of authorization attributes have been proposed in the literature [14], [22], [15], [23], [24], [6]. In [22], the mapping of attributes is proposed. It consists to transform the local attributes using derivation rules to federated attributes, which are attributes defined by the domains but recognized by the federation. The federated attributes in our approach are defined by the federation and are independent of those of the domains. A flexible architecture based on the ABAC model is proposed in [15] to ensure the heterogeneity of inter-domain access control. However, the authorisation decision to access a service is make in the service consumer side based on the collaboration contract as proposed in [10]. While this approach preserves the autonomy of the domains in terms of security, it is point-to-point and hardly supports the authorization changes. Preuveneers et al. [6] propose to align the authorization attributes of domains by declaratively defining equivalence relations between their names

and their values. We use a similar approach, but we utilize intermediate attributes to define the equivalence relationships. Our goal for this is to minimize dependencies between domains and to avoid leakage of security informations.

7 Conclusion

We proposed a cross-domain access control method for service-oriented environments, based on a federation architecture where the domains stay responsible of the access control of their services. At the federation level, a third-party entity, the global access control mediator (GACM) handles trust, interoperability and service composition between the heterogeneous domain access control models. Federated attributes play the role of medium between domains with a double mapping mechanism between domain attributes (such as roles) and federated attributes to keep the domain independence. We proposed an implementation of our method for the access control of web services in which we detail the different steps of implementation, the necessary components and the difficulties encountered. Our method was applied to a case study in order to evaluate it according to feasibility, reliability, scalability and security criteria.
Short term perspectives involve the application of our method. We did not study the performance and scalability of our implementation. In particular, we plan to distribute GACM information into special areas of the domains to improve performance. We also plan to experiment the access control of service composition where composed services are invoked on the behalf on initial requester. Token store integration in the orchestration engines must be also implemented. We plan to extend attribute mapping with individual access permissions to allow fine-grained cross-domain access control. Further work is required to bring assistance in building the mappings when the models (both domain and federation) evolve. Ontologies may be helpful in this case. Since our approach is really modular, we are convinced that it fits to the case where a domain can belong to different federations and possibly to federations of federations.

References

1. OASIS: Reference Architecture Foundation for Service Oriented Architecture Version 1.0. (12 04 Dec 2012)
2. OASIS: Web Services Federation Language (WS-Federation) Version 1.2. Standard. (05 22 May 2009)
3. International Telecommunication Union: Baseline identity management terms and definitions. (04 Avril 2010)
4. Fabian, B., Kunz, S., MüLler, S., GüNther, O.: Secure federation of semantic information services. Decis. Support Syst. **55**(1) (April 2013) 385–398
5. Hafeez, K., Rajpoot, Q., Shibli, A.: Interoperability among access control models. In: 2012 15th International Multitopic Conference (INMIC), Islamabad, Punjab, Pakistan, IEEE (December 2012) 111–118
6. Preuveneers, D., Joosen, W., Ilie-Zudor, E.: Policy reconciliation for access control in dynamic cross-enterprise collaborations. Enterprise Information Systems **12**(3) (March 2018) 279–299

7. Hu, V.C., Ferraiolo, D., Kuhn, R., Schnitzer, A., Sandlin, K., Miller, R., Scarfone, K.: Guide to Attribute Based Access Control (ABAC) Definition and Considerations. Technical Report NIST SP 800-162, National Institute of Standards and Technology (January 2014)
8. Beer Mohamed, M.I., Hassan, M.F., Safdar, S., Saleem, M.Q.: Adaptive security architectural model for protecting identity federation in service oriented computing. Journal of King Saud University - Computer and Information Sciences (March 2019)
9. Kallela, J.: Federated identity management solutions (2008) T-110.5190 Seminar on Internetworking.
10. Menzel, M., Wolter, C., Meinel, C.: Access control for cross-organisational web service composition. Journal of Information Assurance and Security **2**(3) (2007) 155–160
11. Dikmans, L., Van Luttikhuizen, R.: SOA made simple discover the true meaning behind the buzzword that is "service oriented architecture". Packt Pub, Birmingham, UK (2013) OCLC: 847034163.
12. Papazoglou, M.P.: Web services: principles and technology. Pearson/Prentice Hall, Harlow (2008) OCLC: 255863191.
13. Duan, N.: Design Principles of a Federated Service-oriented Architecture Model for Net-centric Data Sharing. The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology **6**(4) (October 2009) 165–176
14. Decat, M., Van Landuyt, D., Lagaisse, B., Joosen, W.: On the need for federated authorization in cross-organizational e-health platforms. In: Proceedings of the 8the international conference on Health Informatics. Volume 8. (2015) 540–546
15. Haguouche, S., Jarir, Z.: Managing Heterogeneous Access Control Models Cross-Organization. In Lopez, J., Ray, I., Crispo, B., eds.: Risks and Security of Internet and Systems. Volume 8924. Springer International Publishing, Cham (2015) 222–229
16. Fragoso-Rodriguez, U., Laurent-Maknavicius, M., Incera-Dieguez, J.: Federated Identity Architectures. In: Proc. 1st Mexican Conference on Informatics Security (MCIS '2006). (2006) 8
17. BAH, A., André, P., Attiogbé, C., Konaté, J.: Federated Access Control in Service Oriented Architecture. Research report, LS2N, Université de Nantes (April 2019)
18. Bertino, E., Martino, L., Paci, F., Squicciarini, A.: Security for Web Services and Service-Oriented Architectures. Springer Berlin Heidelberg, Berlin, Heidelberg (2010)
19. Aruna S, VIT Vellore: Security in Web Services- Issues and Challenges. International Journal of Engineering Research and **V5**(09) (September 2016) IJERTV5IS090245
20. Singhal, A., Winograd, T., Scarfone, K.A.: Guide to secure web services. Technical Report NIST SP 800-95, National Institute of Standards and Technology, Gaithersburg, MD (2007)
21. Jasiul, B., Sliwa, J., Piotrowski, R., Goniacz, R., Amanowicz, M.: Authentication and Authorization of Users and Services in Federated SOA Environments - Challenges and Opportunities. (2010) 13
22. Rubio-Medrano, C.E., Zhao, Z., Doupe, A., Ahn, G.J.: Federated Access Management for Collaborative Network Environments: Framework and Case Study. In: Proceedings of the 20th ACM Symposium on Access Control Models and Technologies - SACMAT '15, Vienna, Austria, ACM Press (2015) 125–134
23. Na, L., Yun-Wei, D., Tian-Wei, C., Chao, W., Yang, G., Yu-Chen, Z.: Cross-Domain Authorization Management Model for Multi-Levels Hybrid Cloud Computing. International Journal of Security and Its Applications **9**(12) (December 2015) 357–366
24. Diniz, T., Felippe, A.C.d., Medeiros, T., Silva, C.E.d., Araujo, R.: Managing Access to Service Providers in Federated Identity Environments: A Case Study in a Cloud Storage Service. In: 2015 XXXIII Brazilian Symposium on Computer Networks and Distributed Systems, Vitoria, Brazil, IEEE (May 2015) 199–207