Collaborative Access Control for Multi-Domain Cloud Computing

Souheil BEN AYED†[a], Student Member and Fumio TERAOKA††[b], Member

SUMMARY The Internet infrastructure is evolving with various approaches such as cloud computing. Interest in cloud computing is growing with the rise of services and applications particularly in business community. For delivering service securely, cloud computing providers are facing several security issues, including controlling access to services and ensuring privacy. Most of access control approaches tend to a centralization of policy administration and decision by introducing a mediator central third party. However, with the growth of the Internet and the increase of cloud computing providers, a centralized administration is no longer supported. In this paper, we present a new collaborative access control infrastructure for distributed cloud computing environment, supporting collaborative delegations across multiple domains in order to authorize users to access services at a visited domain that does not have a direct cooperative relationship with the user’s home domain. For this purpose, we propose an extension of the XACML (eXtensible Access Control Markup Language) model with a new entity called Delegation Validation Point (DVP) to support multi-domain delegation in a distributed environment. We describe the new extended model and functionalities of the new component. In addition, we define new XACML messages for acquiring delegation across domains. For exchanging delegation between domains we use SAML (Security Association Markup Language) and Diameter protocol. Two Diameter applications are defined for transporting securely multi delegation requests and answers and for building a trusted path of cooperation to acquire the chain of delegations. We detail the implemented prototype and evaluate performance within a testbed of up to 20 domains.

key words: access control, delegation, collaborative environment, cloud computing

1. Introduction

The Internet infrastructure is evolving with various approaches, such as cloud computing, grid computing and peer-to-peer, on the way to remove barriers between users and organizations for offering better capacities to deliver new services and ensuring the access to different resources. These approaches are expanding to distributed multi-domain environment, such as multi-domain cloud computing, where users have the ability to access securely services and resources across multiple domains.

Cloud computing is a technology for delivering virtual computing resources, such as computation, applications, data access and storage, as services to users via networks typically over the Internet. In distributed cloud computing environment, services and applications are delivered via Internet by different providers called cloud computing providers (CCPs). For CCPs and customers, security is intrinsic to deliver or access the cloud services. CCPs must ensure delivery of services to customers under a secured infrastructure and to protect clients’ data and applications from unauthorized access. The customers must ensure that their information is securely protected by the CCPs. Security issues associated with cloud computing include the areas of security and privacy that ensure protection of users' data from unauthorized access and maintaining data privacy.

For the rest of this paper, we use the term “multi-domain environment” to refer to a multi-domain cloud computing environment.

Distributed access control and secure interoperability in multi-domain environment are widely studied and different approaches were proposed[1],[2]. Most approaches tend to use an infrastructure relying on a single administrator. However, in a multi-domain environment where users and services belong to different domains, the administration of rules is no longer supported by a centralized approach. There is a need for a collaborative approach for access control in multi-domain environment where policies are defined separately at each of the cooperative administrative domains. The delegation of authority is one of the mechanisms to support decentralized administration of access policies. Delegation is the act of granting a temporary permission to accomplish some tasks or entrusting another party with a part of a job. In the context of Role Based Access Control (RBAC) [3] approach, two delegation approaches may be concerned: 1) the security administration for delegating administrative policies designated as the assignment of role-permission and 2) the delegation of a user’s role to another user, referred to as the assignment of user-role. In the rest of the paper, we use the term “delegator” to refer to the performer and the granter of a delegation, and the term “delegatee” to refer to the receiver and to whom the delegation is granted. Access control is based on some processes essentially for providing the functionalities of Authentication, Authorization and Accounting (AAA). Authentication is the process of identifying the requestor and validating whether his/her identity is what is claimed to be. While the authorization process is granting the right that determines the permissions that the requestor can benefit from the resource. The user’s consumption of resources and services are collected by the accounting process.

In this paper, we study the case where a user tries
Authorization for granting him a direct trust relationship to check user’s authentication and provider and the cloud computing provider with which the iterative relationship between the visited cloud computing service based on the trust relationship between the two cloud and granted the appropriate authorization to access the service. In this case, if the both clouds, the visited cloud computing provider and user’s home cloud computing provider, are establishing cooperations relationship with other domains, we can find a path of cooperative domains between the visited and user’s home clouds computing.

Fig. 1 Access cloud computing at visited domain. (1) access control achieved via cooperative relationship between the visited cloud computing and user’s home cloud computing. (2) Due to a missing of cooperative relationship between the visited cloud computing provider and the home cloud computing provider, the access control is performed via a chain of successive cooperative domains between the visited and user’s home clouds computing.

to access a cloud computing with which he/she does not have a pre-established contract. In the case where the visited cloud computing provider has a cooperative relationship with a cloud computing provider, with which the user has a contract (Fig. 1-(1)), the user can be authenticated using his/her credentials at the home cloud computing provider and granted the appropriate authorization to access the service based on the trust relationship between the two cloud computing providers. However, when there is no cooperative relationship between the visited cloud computing provider and the cloud computing provider with which the user has a contract (Fig. 1-(2)), both clouds can’t establish a direct trust relationship to check user’s authentication and authorization for granting him/her the right permission to access the service. In this case, if the both clouds, the visited cloud computing provider and user’s home cloud computing provider, are establishing cooperations relationship with other domains, we can find a path of cooperative domains that connect the visited cloud computing provider and the user’s home cloud computing provider in order to build a chain of delegations between the intermediate cooperative clouds. Then, based on the chain of cooperative delegations the visited cloud computing domain decides whether to grant or deny the access to the service for the user. Therefore, in order to achieve access control in an environment where none of involved domains have a global view of the collaboration environment, we propose a collaborative multi-domain access control for multi-domain cloud computing based on collaborative access control and delegations across multiple domains.

Existing access control standards, such as XACML (eXtensible Access Control Markup Language) [4], support delegation, however the process of granting delegation is based on loading and parsing policies and rule set at a single PDP (Policy Decision Point) where the final decision is made. Therefore, we propose an extended XACML model for supporting multi-domain delegation by introducing a new entity called the Delegation Validation Point (DVP). Incorporating DVP into the XACML model will extend its functionalities without interfering the normal model operations. DVP takes charge of exchanging delegations across multiple domains, generating delegation requests and validating the chain of acquired delegations. The XACML delegation request and answer are exchanged between the authorization entities in different administrative domains encapsulated into SAML (Security Association Markup Language) [5], an OASIS standard for exchanging authentication and authorization messages. Moreover, the SAML messages are transported securely across domains by the Diameter protocol [6], an Authentication, Authorization and Accounting (AAA) protocol defined by IETF (Internet Engineering Task Force) as the successor of RADIUS [7]. We extended the SAML-XACML profile [8] to support the new exchanged XACML delegation requests/answers. We also define two Diameter applications: 1) for exchanging the SAML/XACML multi-delegation messages between domains and 2) for building the routing tables of cooperation paths to select a trusted path for acquiring delegations. A prototype implementation was evaluated within a virtual network testbed of up to 20 domains showing a reasonable response time results for performing access control. The negotiation and establishment of the trust relationship between domains are out of scope of this paper. We assume that all domains established the trust relationship with other domains prior to the negotiation of the access control across multiple domains

The rest of this paper is structured as follows. Section 2 presents the SAML and XACML standards for access control in addition to the Diameter protocol. Section 3 gives an overview of related work. Section 4 describes the new collaborative access control infrastructure and the cooperative delegation concept. Section 5 introduces the DVP entity and how the XACML model is extended to support multiple delegations. Section 6 describes the exchange of data for building the routing table of cooperation path and the secure exchange of delegation across multiple-domains. The two new Diameter applications and their messages are presented in Sect. 7. In Sect. 8 we describe the implementation and provide results of the evaluation. Finally, we conclude the paper in Sect. 9.

2. Existing Technologies

2.1 XACML/SAML

SAML [5] and XACML [4] are both XML-based standards proposed by the Organization for Advancement of Structured Information Standards (OASIS). They share the concepts of authentication, authorization and access control; however, they address different problems. While XACML provides the language for describing access control and authorization and enforcing decision, SAML provides the

---

†This work is part of the “Universal AAA infrastructure project” supported by the AAA working group of the WIDE Project for the design of a AAA architecture for next-generation services in the Future Internet.
mechanisms for exchanging authentication and authorization data between security cooperating domains. Since both SAML and XACML are based on XML, they are extensible and flexible standards.

2.1.1 XACML Standard

XACML expresses access control policies in the XML language. It was developed in the way to create a portable and standard language for describing access control entities and their attributes. It also aims to provide a mechanism that offers the ability to carefully define a finer granular access control and enforce before and after actions with both permit or deny permissions. The XACML architecture includes four main components: the Policy Enforcement Point (PEP), the Policy Information Point (PIP), the Policy Administration Point (PAP) and the Policy Decision Point (PDP). As shown in Fig. 2, access requests are sent to the PEP where XACML request is created and sent to its PDP. The PDP evaluates the request and makes decision based on policies then sends back a response to the PEP. Policies and policy sets are configured by the PAP and are available for the PDP. Additional attributes related to the subject, the resources and the environment can be provided to the PDP by the PIP. XACML also defines the mechanisms for creating rules by which authorization is evaluated. The policy language model is structured into three levels of elements: the policy set, the policy and the rule.

In XACML, the ability to delegate administrative rights is one of new features newly implemented in XACMLv3. With delegation capabilities, administrators are able to delegate temporary administrative rights to other administrators. In XACMLv3, the delegation rights, referred to as administrative policies, are separated from the access rights, referred to as access policies. In an authorization scenario where administrative policies are not centralized, making access control based on delegation of authority requires a PDP at each administrative domain for parsing the graph of policies. However, the XACMLv3 model delegation process considers a single PDP that has a global view of the graph of policies which can not be sufficient when the administration of access control is not centralized.

2.1.2 SAML Standard

SAML addresses the need to provide a mechanism for exchanging authentication and authorization information between security domains. It is also a mechanism for ensuring portable trust; hence it solves the problem of Single Sign-On for web applications by allowing to keep user’s information with a few parties trusted by the user and to allow them to share information with other interested cooperat-
ing organizations after an explicit approval of the concerned user. Consequently, the user can access a range of services. In the context of a collaborative multi-domain environment, extending the SAML-XACML profile to support encapsulating new XACML request and answer will enable a secure communication over SAML 2.0 between remote XACML model components. Thus, we use the SAML 2.0 protocol for exchanging XACML new added messages to support access control across multiple domains.

2.2 The AAA Protocol: Diameter

The access control is the way how users are controlled to be allowed or denied to access services, network or data. Authentication, Authorization and Accounting (AAA) is a term for a framework for controlling access to resource, enforcing policies, auditing usage and providing the necessary information to bill for services.

The Diameter protocol [6] is a AAA protocol, defined by IETF to overcome limitations and complications revealed with the previous AAA protocols such as RADIUS. It offers improvements in security, flexibility, scalability and reliability. It has also been adopted by many standard bodies including 3GPP [10], 3GPP2 [11], WimaxForum [12] and MSF (the MultiService Forum) [13] as the main AAA protocol. The Diameter protocol is also intended to be a framework for future AAA services.

The Diameter protocol is composed of Diameter Base Protocol defined in [6] and additional Diameter applications (Fig. 4). Diameter Base Protocol basically focuses on the general way of exchanging Diameter messages between Diameter peers. It provides the common functionalities for capability negotiation and accounting. Nevertheless, the authentication and authorization functionalities are not defined by Diameter Base Protocol as these mechanisms vary among services. The Diameter protocol can be enhanced with additional functionalities by adding new Diameter applications. All Diameter applications can benefit from the Diameter Base Protocol capabilities. Various Diameter applications are defined such as for IP mobility [14], [15], network access [16] and Session Initiation Protocol (SIP) [17].

The Diameter protocol is a peer-to-peer protocol where any peer can initiate a Diameter request. A Diameter peer can act as a Diameter client, a Diameter server or a Diameter agent. The Diameter agent is an intermediate node in the network that performs a particular task in the AAA network, particularly to proxy Diameter messages across multiple domains. Typically, there are four kinds of Diameter agents: Relay agents, Proxy agents, Redirect agents and Translation agents. Diameter agents are defined to support multi-domain environment by performing particular functionalities to reduce Diameter servers’ tasks and simplify routing messages across multiple domains. All Diameter peers communicate together by exchanging Diameter messages identified by a command code and a session id. The message is composed of a header and a set of Attribute-Value Pairs (AVPs). Security data, application data and routing information are included in successive and different AVPs identified by their AVP codes. Message information such as message length, version, command code and application Id are included in the message header located at the beginning of the message.

3. Related Work

Distributed access control and secure interoperability have been considerably studied in previous work, particularly for centralized and mediated multi-domain environments. Issues and challenges associated with secure interoperability in a multi-domain environment have been addressed in several research efforts [1], [18], [19]. The approaches proposed in the literature can be classified into two categories: centralized approach and decentralized approach.

Mainly, all centralized approaches require a third party as a trusted central mediator that enables domains to collaborate and inter-operate securely. In [1] the centralized third party should have a global view of the collaboration environment to perform composition of multi-domain policy. Dawson et al. [19] proposed secure interoperability for heterogeneous database where all requests go through a central mediator. The central third party should be trusted by all domains and gather all the access control policies of all domains and be notified of any updated policy. Hence, any update or modification of any policy affects the third party decision.

In the second approach, access control decision is made through collaboration between multiple domains. Mainly, two key problems were highlighted: inter-domain role mapping and multi-domain delegation. In [20] Shehab et al. proposed a distributed secure interoperability protocol providing a secure role mapping technique between domains. In [2], a web service discovery protocol based on SOAP (Simple Object Access Protocol) is proposed to discover an access path for achieving mediator-free collaboration by mapping roles between neighbor domains. The discovered path is authenticated to establish a secure access path between the home domain and the target domain. In another research Shehab et al. [21] propose a role routing protocol for searching the neighboring domains for the shortest secure path. The protocol minimizes the path authorizations by choosing the shortest path of role routing between neighbor domains for a secure collaboration. However, targeting the shortest path that minimizes the path authorizations provides no guarantees to receive the optimal or the required authorizations for access control. Several other researches related to role mapping have been proposed in Hu et al. [22]
and Du et al. [23]. However, role mapping researches are faced several challenges and issues such as for discovering roles and establishing the mapping of roles that lead to complex solutions. Chadwick et al. [24] proposed an extension of XACML to support multi-domain user-to-user dynamic delegation of authority for delegation of role from one user to another. The solution solves the problem of delegation in multi-domain by delegating authority of user’s role to another based on a graph of delegation and a hierarchical relationships of the various sets of attributes. However, the final decision based on policies and delegation graph is centralized. While role mapping mechanism requires a high collaboration between domains which rise several challenges, the delegation of authority in a multi-domain environment reduces the complexity of collaboration between domains for making authorization decision by delegating temporary administrative permissions. Therefore, we adopted the delegation of authority approach in order to build collaborative access control across multiple domains.

4. Collaborative Access Control Infrastructure

4.1 Proposed Infrastructure

To address the problem of access control in a multi-domain environment, we propose a new collaborative access control infrastructure (Fig. 5) based on Diameter protocol, an extended XACML model and the SAML standard for enabling delegation between domains. With this Diameter-based infrastructure we aim to allow users visiting a CCP domain to access cloud services based on their contracts established with their home CCP. Therefore, we consider a collaborative environment in which all domains are establishing cooperative relationships with other domains forming a graph of cooperations. However, none of the domains have a global view of the collaborative environment. In addition, all domains adopt the Role Based Access Control (RBAC) [3] to model their access control policies and authorizations. They also share the same classification of services/resources and their attributes and actions. Routing the authentication and the authorization messages between different Diameter servers in the collaborative environment is achieved by the routing protocol described by Diameter protocol. The routing protocol defines how to forward a Diameter message via Diameter agents, including intermediate Diameter proxy agents of non-cooperative domains, across multiple realms to route it to the destination domain then to the appropriate destination host.

The access control can be made easily when the two providers share a cooperative relationship allowing each other provider’s users to access services available in the both domains. However, when such direct cooperative relationship is missing, we provide a solution based on delegation across multiple domains in a collaborative environment. We define a cooperative relationship between two domains as a contractual agreement that engenders trust on other party’s access control decisions and delegations; in addition, it defines the terms of the established cooperations, such as the period of cooperation, the services concerned in a cooperation and fees. Each domain establishing a cooperation relationship with another domain defines a set of administrative policies and rules to be satisfied. It also shares the list of user roles able to benefit from the cooperation and the delegation across multiple domains. In addition, it shares the list of services and resources able to be used or delegated. Delegation is the act of transferring some administrative “power”, allowing other actors or roles to have limited administrative power during a period of time, to create or grant permissions for persons to access some resources. By acquiring delegations to cooperative domains and based on the trust relationship established between cooperative domains, a path of trusted delegations is created from the visited domain to the user’s home domain to enable access control in a collaborative environment.

As depicted in Fig. 5, when a user wishes to access a service at a visited domain CCP-A, the user is firstly identified and authenticated at the home domain CCP-B then authorization is checked. At the visited domain A, the user is requested to identify itself to determine to which domain the user belongs. We assume that user’s identity format includes the user name and the domain to which he/she belongs. Upon identifying the user, the visited domain generates an authentication request and sends it to the home domain. Then, the home domain starts negotiating authentication data with the user. The authentication of the user is achieved based on Diameter protocol and EAP (Extensible Authentication Protocol) [25]. All authentication negotiations are exchanged between the Diameter NASREQ server [16] at the visited domain and the Diameter EAP server [26] at the user home domain via the Diameter network where authentication messages are forwarded through Diameter proxy agents of intermediate diameter domains (Fig. 5-1: Authentication). Since the negotiation of authentication data is carried out across multiple domains which may be untrusted by neither the visited domain nor the home user domain, it is highly recommended to use a strong EAP authentication mechanism supporting mutual authentication such as EAP-TLS, EAP-TTLS or PEAP.

While the authentication requests are forwarded to the user’s home domain based on the Diameter routing mechanism; for the authorization, the requests should be forwarded between successive cooperative domains to acquire delegations and establish a chain of trusted delegations between the visited domain and the home domain. Therefore, prior to the authorization process, a sequence of successive cooperations between the visited domain and the home domain is selected forming a path of cooperative domains, called cooperation path. The user successfully authenticated is requested to choose criteria for the selection of the appropriate path among available paths of cooperations linking the both domains. The protocol for exchanging criteria and the mechanism for negotiating user’s selection of criteria are out of scope of this paper. Since, none of the domains have a global view of the collaborative environment, and the es-
established cooperative relationship are set for a long-term period and rarely updated, we propose a protocol for building the routing tables of cooperation paths at each domain for a partial view of the global environment. Each domain sends the list of its cooperations to all its cooperative domains. Consecutively, domains receiving a list of cooperations forward it to their cooperative domains. Nevertheless, each list of cooperations is forwarded only for a limited number of hops which restrict the range of cooperations knowledge at each domain and fix the maximum length of selected path of cooperative domains. Combining the received cooperations lists, each domain builds its routing table of cooperation path for a limited view of the collaborative environment. The definition of this protocol and details will be described in Sect. 6.

In the XACMLv3 model, both the authorization policies and the administration control rules work together to achieve the delegation of authority [27]. While the authorization decision is made based on the access policies and rules at the visited domain, the delegation is acquired based on administration control policies. To enable delegation across multiple domains, an enhancement of the XACMLv3 model is proposed. In the new extended model, users are authorized at the visited domain, after being delegated by cooperative domains via its home domain (Fig. 5-2: Authorization). The exchange of delegations between decisional points in two cooperative domains is performed by a new component of the access control infrastructure which we call the Delegation Validation Point (DVP). When the user belongs to a domain non-cooperating with the visited domain, the DVP at the visited domain creates a delegation request to the next hop-domain and sends it to the PDP in the visited domain. Then, the PDP makes decision based on administrative policies and returns delegation decision with attributes to the DVP. The DVP forwards the acquired delegations to the next cooperative domain in the selected path until reaching the user's home domain where the user is delegated based on his/her contract with the home domain. The DVP is also in charge of validating the acquired delegations to verify the continuity of the chain of delegation from the visited domain to the user's home domain. All the acquired XACML delegation decisions are encapsulated in SAML messages and securely exchanged between cooperative domains via Diameter protocol.

4.2 Cooperative Delegation of Authority

With cooperative relationship between domains we intend to enable a domain to delegate authority to another domain with which it has a cooperative relationship. Delegation acquired from one domain to another domain grants the delegatee some privileges and permissions to act on behalf of the delegator during a limited period of time. We assume a collaborative multi-domains environment where each domain is establishing one or more cooperative relationship with other domains.

We denote by $D_i(x, y, z)$ the root delegation identified by $i$ granted by the delegator $x$ to the delegatee $y$ for a number of delegation depth equal to $z$. The delegation depth is the number of recursive sub-delegations, also called transitive delegations, allowed to be acquired based on the root delegation. For delegations acquired from a domain $x$ to a user $u$ denoted by $D_i(x, u)$, the delegation depth parameter is not used as we do not consider in this study the ability of users to delegate permissions to other users or domains. A delegation $D_i$ is granted from $x$ to $y$ only if $x$ can-delegate the requested authority to $y$ for a delegation depth $z$ based on the administrative rules defined at the domain of $x$. Where can-delegate is a relation that specifies whether a user is authorized to delegate permissions.

We also define a sub-delegation denoted by $D'_i(n, m, p)$ as a delegation identified by $j$ granted from the delegator $n$ to the delegatee $m$ for a delegation depth $p$ based on another acquired delegation $D_i$. A sub-delegation $D'_i(n, m, p)$ is
granted based on a previously acquired delegation \( D_i(x, y, z) \) only if the delegatee of \( D_i \) is identically the same delegator of \( D_j \), which means \( n = y, n \ can-delegate \) the requested administrative rights to \( m \) based on the administrative rules defined at the domain of \( n \) and the assigned delegation depth is \( p \leq z \). In addition, a sub-delegation \( D' \) granted based on another delegation \( D_i \) must be validated by the domain that issued the delegation \( D_i \).

The chain of delegation between two domains \( U \) and \( V \) is created from an acquired delegation from a domain \( U \) and a successive sub-delegations granted by cooperative domains to the domain \( V \) where:

- The delegatee domain of the delegation \( D_0 \) granted by \( U \) is the delegator of a sub-delegation denoted by \( D_0 \)
- There exists a number \( n \) successive delegations \( D' = \{D_0', D_1', \ldots, D_{n-1}'\} \) such that for all the delegations \( D_{i-1}' \), \( 0 < i < n \) the delegatee is the delegator of \( D_{i+1}' \) and \( D_{n-1}' \) is the delegation from the domain \( V \) to the user \( u \) who requested the access.
- Each of the involved domains is granting one and only one delegation or sub-delegation in the chain of delegations.

As shown in Fig. 6-(a), an example of delegation from a domain \( A \) to another domain \( B \) for a delegation depth equal to \( p_1 \) is denoted by \( D_1(A, B, p_1) \) and referred to as \( D_1 \). Based on the delegation \( D_1 \), the cooperative relationship between domains \( \{A, B, (B, C), (C, D), (D, E)\} \) and the selected path of cooperations \( \{A, B, C, D, E\} \), we can acquire successive sub-delegations from the domain \( B \) to the user \( u \) at the home domain \( E \). As a result, at the home domain a chain of delegations is created as follows:

\[
D_1(A, B, p_1); \ D_2(B, C, p_2); \ D_3(C, D, p_3); \ D_4(D, E, p_4); \ D_5(E, u).
\]

In this chain of delegations, \( D_5 \) is acquired based on the delegation \( D_1 \) and the sub-delegations \( D_2, D_3 \) and \( D_4 \). However, except the root delegation \( D_1 \), all other sub-delegations are not yet validated. The domain \( A \) can not trust on the chain of delegations until the validation of all the acquired sub-delegations.

The chain of delegations is created from delegations granted successively from the visited domain \( A \) to the home domain \( E \). However, the validation of the chain of delegation is performed from the domain issuing the last acquired sub-delegation \( D_5 \) until reaching the domain issuing the root delegation \( D_1 \). Each delegator domain must validate all the sub-delegations recursively acquired based on the delegation or the sub-delegation it grants. A delegation or a sub-delegation is considered valid after 1) authenticating and validating credentials of the issuer domain, 2) verifying the integrity of its sub-delegations and the continuity of the chain of sub-delegations and 3) validating all of its sub-delegations. The invalidity of one of the sub-delegations implies the invalidity of the delegation and the failure to acquire the chain of delegations.

Figure 6-(b) illustrates an example of validating acquired delegations between domains. In this figure a trusted delegation \( i \) is denoted by \( \{D_1', A, i\} \), while \( D_{i-1}' \) denotes a delegation not validated but not invalid. After acquiring all the delegations, the domain \( E \) returns to the domain \( D \) the user delegation \( D_5 \), by default validated by the issuer domain \( E \), and all the other acquired delegations. At the other involved domains \( A, B, C \) and \( D \), delegations and their sub-delegations must be validated. However, since there is a trust relationship between the validator domain and its previous domain, the former can trust on the validation performed by the latter on the sub-delegations reducing the number of sub-delegations to be validated at each domain. For example, at the domain \( C \), the received delegations validated by its previous domain \( D \) are \( D_2' \) and \( D_3' \). The domain \( C \) must validate the sub-delegations of \( D_3' \) which means the delegations \( D_4' \) and \( D_5' \). Then, after authenticating and validating the issuer of \( D_3' \) (i.e., the domain \( D \)) and based on the trust relationship between \( C \) and \( D \), the domain \( C \) can trust on the validity of the sub-delegation \( D_4' \) and its sub-delegation \( D_5' \). Consecutively, at the visited domain \( A \), after successfully authenticating and verifying the issuer of the delegation \( D_5' \) and based on the trust relationship between domains \( A \) and \( B \), the sub-delegations of \( D_2', D_3', D_4' \) and \( D_5' \), successfully validated by the domain \( B \), are also considered as valid sub-delegations by the domain \( A \).

5. The DVP and Extensions of the XACML Model

This section describes the DVP component and how it is incorporated into the XACML model. It details the DVP functionalities and which parts of the existing model are modified and what kind of data are exchanged with the XACML components.
5.1 Extended Model

As we indicated, we extend the XACML model by incorporating a new component called Delegation Validation Point (DVP) to enable it to support delegation in a multi-domain environment. The extension of the XACML model does not interfere its normal functionalities. The DVP component is called by the PDP when a decision of an authorization request requires delegation from another domain.

In Fig. 7, we show the extended XACML model. When an authorization access request is received, the PEP creates an XACML decision request (Fig. 7-3) then sends it to its local PDP. Based on local policies, the PDP checks the ability to verify the visitor user’s authorization. If the user belongs to another domain with which the visited domain does not have a cooperation relationship, the PDP generates an XACML multi-delegation request (Fig. 7-11.a) and sends it to its DVP. An example of an XACML multi-delegation request is given in Fig. 8 (Line 1 to Line 34). It includes the identity of who requests the access (Fig. 8-L.7), the resource (Fig. 8-L.14), the requested action (Fig. 8-L.20), the delegator identity (Fig. 8-L.26) and optionally the acquired delegations if they exist. After choosing the cooperation path, the DVP creates an XACML delegation request (Fig. 7-11.b) for acquiring delegation to the next hop domain and sends it to the PDP. After verifying the delegation request and generating the appropriate delegation answer, the PDP sends back an XACML delegation response (Fig. 7-11.c) to the DVP. For verifying a delegation request, the PDP uses only administrative policies and parses them to find a path of policies that allow granting the requested action for the resource from the delegator to the delegatee. XACML multi-delegation requests are forwarded from the cooperative domain to the next cooperative domain in the selected path until reaching the user's home domain. Then an XACML multi-delegation answer (Fig. 7-11.f) is returned to the visited domain DVP including all the acquired delegation to be validated.

Figure 9 shows an example of XACML delegation request and response exchanged between a DVP and its local PDP. The XACML delegation request (Fig. 9-L.1 to L.30) includes a <DelegationRequest> that contains the delegator identity (Fig. 9-L.25), to whom the delegation is requested (Fig. 9-L.6) and for which actions (Fig. 9-L.19) and resources (Fig. 9-L.13), in addition to other optional attributes. The delegator and delegatee attributes are given from the cooperation relationship. The format of the delegator and the delegatee attributes includes the subject identity and the domain to which it belongs separated by ‘@’ as defined by the data type rfc822Name. The XACML delegation response (Fig. 9-L.32 to L.56) includes a Decision Result (Fig. 9-L.34), including either permit or deny, and the decision attributes, e.g., the delegatee (Fig. 9-L.39), the delegator (Fig. 9-L.45), the delegation MaxDepth (Fig. 9-L.51) and optionally the delegation validity time. After verifying the received delegation answer, the DVP creates an XACML multi-delegation request which includes all the acquired delegations, then submits the new request to the next hop domain in the cooperation path. At intermediate domains, when receiving an XACML multi-delegation request, the DVP creates a delegation request for the next hop and sends...
the PDP. The process continues until reaching the user’s home domain, where the user is delegated. The DVP at the user’s home domain encapsulates all the acquired delegations in a multi-delegation response (Fig. 8-L.36 to L.67) and sends it back to the previous hop domain. The XACML multi-delegation responses are forwarded through DVPs to the PDP at the visited domain. By receiving all the acquired delegations and delegation attributes, the PDP at the visited domain generates the associated administrative requests and builds the reduction graph of policies. Then, it searches for a path of policies to verify access authorization as described by the XACML administrative delegation process.

5.2 Delegation Validation Point (DVP)

The DVP is incorporated into XACML in order to support multiple delegations across a distributed multi-domain environment. DVPs act as intermediaries between two XACML models in two different domains. They exchange SAML queries and responses, and with other DVPs located at cooperative domains. Further, the DVP exchanges the XACML delegation queries and responses with the local PDP for acquiring delegations. In addition to creating and forwarding delegation queries and responses, each DVP component performs the following functionalities:

- **Validate credentials**: The DVP receives queries and responses from its cooperative neighbors. The DVP should validate and authenticate the SAML signature if present. It verifies that the key signed by the Authentication Authority has not been compromised and that its public key is still valid.
- **Check delegated nodes**: The DVP receives responses containing multiple-delegations that form the path of delegations. These responses may include delegations from or to domains not having a cooperative agreement with the current DVP domain. Therefore, based on a list of revoked domains a DVP may revoke an issued delegation included in the received response which implies the break of the chain of delegations. In that event, a new delegation path should be selected for achieving the multi-delegations.
- **Validate the acquired delegations**: The DVP must verify that all acquired delegations included in a received multi-delegation response are valid and form a path of trusted delegations. This means that none of the delegations were revoked by another DVP.

5.3 SAML Profile for the Extended Model

We use the SAML protocol for carrying the XACML requests and responses. Figure 10 shows components and the
newly defined SAML messages exchanged between the visited domain and the next delegation hop domain. We add four new SAML messages to exchange the XACML requests and responses, we also modified the SAML/XACML profile to support the new messages. The DVP component exchanges with the local PDP XACML delegation request and answer encapsulated respectively in an XACMLDelegationQuery and XACMLDelegationResponse. Between DVPs in different domains two messages are exchanged: XACMLMultiDelegationQuery and XACMLMultiDelegationResponse. These two SAML messages include the XACML acquired delegations for the selected chain of delegation in addition to the issuer information including its digital signature and certificate.

6. Cooperation Path and Secure Exchange of Delegations

In this section, we describe the procedure for exchanging cooperations data between domains to create the routing table of cooperation path that serves for selecting a path to establish delegations and enable collaborative access control. Then, we discuss the secure exchange of delegations across multiple domains.

6.1 Paths of Trusted Cooperative Domains

One of the challenge of the solution is how to establish a chain of trusted cooperations between domains. As none of the domains have a global view of the collaborative environment, we propose a procedure for collecting the cooperations data of a limited view of the global collaborative graph by specifying a hop-depth. The hop-depth is a counter that limits the maximum number of domains through which an update of cooperation data is forwarded before discarding it. Each domain dispatches to each of its cooperative domains an update message including the list of cooperations it establishes with other domains.

As shown in Fig. 11, domains A, B, C, D, E, F and G are establishing cooperations in a multi-domain environment. We define $C_i(y, a)$ sent from a domain $x$ as a cooperative relationship between the two domains $x$ and $y$, where $a$ is an optional parameter of the cooperation attributes. For example, in this figure, domain A established three cooperations with domains B, C and D as follows: $C_A(B, attr)$, $C_A(C, attr)$, $C_A(D, attr)$. When a cooperation is established, modified or deleted, the concerned domain forwards an update message including the list of updated cooperations and their attributes to its neighbor cooperative domains. Various attributes may be included with a cooperation such as the type of update: created, deleted, or modified cooperation, the lifetime of the update and the limitations of the cooperation. The format of attributes parameter and the syntax are out of scope of this paper.

As shown in Algorithm 1, each domain receiving a cooperation update extracts the included list of cooperations and updates its local routing table of cooperation path. The algorithm is executed by the Diameter Cooperation Path Routing Application as described in Sect. 7.1. A message ID is used to identify the received update and avoiding parsing a redundant update. Then, the update is forwarded to the other cooperative neighbors if not reaching the hop-depth limit of the update message otherwise it is discarded. In order to avoid forwarding update messages in a loop, they are forwarded only to neighbor cooperative domains that are not included in the update message.

6.2 Secure Exchange of the Multiple Delegations

DVPs exchange XACML multiple-delegations messages between cooperative domains via Diameter network. Diameter messages are forwarded through Diameter agents, such as proxy, that may exist in non-cooperative intermediate domains. Therefore, encryption of the exchanged chain of delegations between cooperative domains and ensuring their integrity and authentication are essential for securing the exchange of delegations. SAML 2.0 uses digital signature based on the XML signature standard for the authenti-

![Fig. 11](image-url) An example of forwarding cooperative updates across multiple domains.

**Algorithm 1** Algorithm executed by a cooperative node upon receiving a cooperation update message

```plaintext
Require: msg (Cooperative message)
if isNewMsg(CoopMsg,Id) then
  for i = 0 to CoopMsg.CoopDomainsLength do
    Domain := CoopMsg.CoopDomains[i]
    if !CoopExist(CoopMsg, Domain, Id) then
      NewCoop(CoopMsg, Domain, Id)
    end if
  end for
if CoopMsg.Hop > 1 then
  for n = 0 to Neighbors do
    if (Node(n) != CoopMsg.D)
      and(Node(n) in CoopMsg.CoopDomains)
      and(Node(n) != CoopMsg.Src) then
        ForwardMsg(CoopMsg, CoopMsg.Hop − 1, Node(n))
      end if
    end for
  end if
else
  Discard(CoopMsg)
end if
```

IEICE TRANS. INF. & SYST., VOL.E95–D, NO.10 OCTOBER 2012
cation and message integrity. To ensure authentication and message integrity, all XACML multi-delegation messages exchanged between DVPs are encapsulated in SAML messages including a digital signature and credentials. In addition, between each pair of two cooperative domains we use the Diameter protocol end-to-end AVP encryption to encrypt all AVPs transporting the XACML data encapsulated in SAML messages. As shown in Fig. 12, with Diameter AVP encryption Diameter agents in intermediate domains may forward messages, however, they are not able to read SAML AVP content nor the exchanged delegation between cooperative domains. Moreover, with SAML digital signature we ensure the authenticity and the integrity of encapsulated chain of delegation between DVPs in cooperative domains. Furthermore, Diameter protocol uses TLS (Transport Layer Security) protocol to exchange securely Diameter messages between Diameter nodes.

7. Diameter Applications

For exchanging messages across domains, we define two Diameter applications: the Cooperation Path Routing (CPR) Application and the Multi-Domain Delegation (MDD) Application.

7.1 Cooperation Path Routing Application

Diameter Cooperation Path Routing application aims to create a proactive routing table of cooperation path that serves to select the cooperation domains for acquiring the required delegations in order to perform the access control. The Diameter CPR application should be available at all cooperative domains. Each Diameter application starts with a pre-configured list of the cooperative domains and forwards this list to all domains with which it established a cooperative relationship. It also receives cooperation updates from other cooperative domains to enrich and update its routing table of cooperation path. We define a new Diameter message to include the list of updated cooperations (Fig. 13). Newly added AVPs for this Diameter application are described in Table 1. When an update is made for one of local established cooperations, the Diameter CPR application creates a Diameter cooperative update request message including a new created update identifier, a Hop limit, and list of updated cooperation and their attributes. Then it forwards the update to all its cooperative domains except the domain with which the cooperation is updated. If more than one cooperation are updated, different Diameter cooperation update messages are created for each cooperative domain and the destination of the message, including a different update identifier. Each received Diameter cooperation update message including a redundant update identifier is discarded. Received cooperation updates with a hop-limit less than or equal to 1 are discarded after updating the local routing table.

7.2 Multi-Domain Delegation Application

The Diameter Multi-Domain Delegation Application is used for carrying out the SAML/XACML queries and responses between the cooperative domains. We define a new Diameter message (Fig. 14) and new AVPs: the SAML-Content AVP, the Delegation-Path AVP and the Path-Node AVP (Table 1). The SAML-Content and Delegation-Path AVPs are specified as encrypted AVP for end-to-end security so that only the source and the destination hosts are able to read their contents. When a Diameter MDD application receives a Diameter Multi-Delegation request it creates a new multi-delegation session and extracts the SAML-Content to forward it to the local DVP in the same domain.
cess delegation is returned in the delegation answer received from the DVP, the Diameter MDD application creates a new Multi-Delegation request and forwards it to the next hop-domain. Otherwise, if the delegation request is rejected, it returns a Diameter Multi-Delegation answer message to the previous domain including a failure for acquiring the multiple delegations. A Reply-Message AVP may be included in success and failure answer messages including a text message to display to the user. The Diameter MDD application receives also Diameter multi-delegation answer from its next hop-domain. The Diameter multi-delegation answer can include either a chain of delegations to be validated and forwarded to the visited domain or a failure answer. When a failure answer is received the Diameter MDD application forwards the same answer to the previous domain in the delegation path until reaching the visited domain. Otherwise, it retrieves the SAML-Content and forwards it to the local DVP for validation. Then, it creates a new Diameter multi-delegation answer and includes the SAML-Content returned by the local DVP after validation then sends it to the previous domain in the path of delegations. The Diameter MDD application maintain the multi-delegation session until forwarding the associated Diameter multi-delegation answer to its previous domain or until expiration of the multi-delegation session time.

8. Implementation and Evaluation

8.1 Implementation

We implemented a prototype to validate the collaborative access control and evaluate the proposed solution. Three main components were implemented: 1) the DVP and the XACML model components, 2) the Diameter Cooperation Path Routing Application and 3) the Diameter Multi-Domain Delegation Application.

For the implementation of the DVP and the XACML components, we faced a number of problems when looking for a full implementation of XACMLv3. As the final specification was published in 2010, we did not find an available XACML package implementing all the required functionalities. Therefore, we opted to use the SunXACML [28] package offering an implementation on XACMLv2 and the SIC-SACML: XACML 3.0 patch [29] that updates SunXACML with some of the functionalities of XACML 3.0. We updated the missing parts to be able to implement a PDP and PEP for exchanging requests and accepting delegations. We also implemented the DVP according to the extensions described in Sect. 5. We have specified the list of cooperative relationship, the administrative access rights and the policies required for achieving delegation across domains.

The two Diameter applications were implemented using freeDiameter [30], [31], an implementation of Diameter Base protocol. We also make use of the DiamEAP server [32], [33], an implementation of the Diameter EAP Application, for the authentication of users. For the two newly defined applications we created a new dictionary file for including the definition of the new Diameter messages and new added AVPs. All dictionaries are loaded at the startup of the Diameter servers to enable to parse Diameter messages and understand the new exchanged AVPs. Diameter CPR application loads the list of cooperations from a pre-configured file and creates a list of cooperation relationship and their associated attributes. Then, when the application server is launched, a Diameter message including the list of local established cooperations is created and forwarded to all its cooperative domains. Also a thread is started for checking any modification of local cooperations and forwards updates to cooperative domains. When a Diameter message including cooperation update is received, the message identifier is verified based on a cache table of previous received updates. The routing table of cooperation path is parsed row by row for updating the table based on each entry of the received data. For forwarding cooperation path data to other domains, a new Diameter message with new session id is created, while maintaining the same update identifier AVP and enhancing the delegation path with the local delegation path data. The main functionality of the Diameter MDD application is to exchange securely SAML/XACML messages between cooperative domains. Based on the DVP exchanged messages, the MDD application creates a Diameter message including the SAML-Content AVP and Delegation-path AVP. Then the Diameter MDD application forwards the Diameter message to next hop destination following the delegation-path order.

<table>
<thead>
<tr>
<th>AVP name</th>
<th>AVP Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Update-Id</td>
<td>OctetString</td>
<td>contains an unique identifier for each cooperation update message.</td>
</tr>
<tr>
<td>Path-Source-Host</td>
<td>OctetString</td>
<td>contains the source host identifier.</td>
</tr>
<tr>
<td>Path-Source-Realm</td>
<td>OctetString</td>
<td>contains the source realm identifier.</td>
</tr>
<tr>
<td>Coop-Domains</td>
<td>Group</td>
<td>contains the list of cooperations updated at the source host.</td>
</tr>
<tr>
<td>Coop-Node</td>
<td>Group</td>
<td>contains Node-Realm, Node-Host and Coop-Attributes AVPs.</td>
</tr>
<tr>
<td>Node-Realm</td>
<td>OctetString</td>
<td>contains a domain identifier.</td>
</tr>
<tr>
<td>Node-Host</td>
<td>OctetString</td>
<td>contains a host identifier.</td>
</tr>
<tr>
<td>Coop-Attributes</td>
<td>OctetString</td>
<td>contains cooperation attributes.</td>
</tr>
<tr>
<td>Hop-Limit</td>
<td>Unsigned32</td>
<td>contains the max hop to not exceed when forwarding this path update.</td>
</tr>
<tr>
<td>SAML-Content</td>
<td>OctetString</td>
<td>contains the SAML requests and responses. Encrypted AVP.</td>
</tr>
<tr>
<td>Delegation-Path</td>
<td>Group</td>
<td>contains ordered Coop-Node AVPs defining the path for acquiring delegations. Encrypted AVP.</td>
</tr>
<tr>
<td>Path-Node</td>
<td>Group</td>
<td>contains Node-Realms and Node-Host AVPs. It describes one delegation domain.</td>
</tr>
</tbody>
</table>

Table 1 AVPs for Diameter Cooperation Path Routing and Diameter Multi-Domain Delegation applications.
8.2 Performance Evaluation

The evaluation of the DVP and the Diameter applications were conducted in a virtual machine environment. A testbed of up to 20 domains was prepared for the performance measurement. The number of cooperation relationship per domain is configured aleatory without exceeding 8 cooperations per domain. For synchronizing the clocks of computer system at all domains, we configured Linux servers to use Network Time Protocol (NTP) which supports an accuracy of time down to nanoseconds. The evaluations do not include the transmission time for sending signal between Diameter servers.

Two separated evaluations were conducted. The first evaluation was prepared to measure the average time for building the routing table of cooperation path at all domains. The second evaluation concerns the multi-delegation process, including the evaluation of policies and the validation of delegation during all the authorization process. Firstly, we configured all the DVPs and PDPs at the different collaborative domains to work with administrative and access policies and we prepared the Diameter applications to enable and configure cooperation parameters retrieved from their application configuration files. All evaluations were carried out 100 times and all measurements at the different domains were collected and averaged.

For the first conducted evaluation, we varied the number of hops for dispatching the delegation path information form 2 to 9 hops. During the evaluation, we collected the average time for building the routing table of cooperation path. Results are shown in Table 2. The average time is increasing without exceeding 52 ms for 9 hops. As the evaluation is conducted within a virtual network, we expect more time for building cooperations path table when it is deployed within a real distributed environment. If we estimate that the transmission delay for one hop between two cooperative domains is 5 ms, then the time required to build the routing table for 9 hops is: (9 (hops) × 5 ms) + 51.3 (processing time) = 96.3 ms. Therefore, we can say that building the routing table at the beginning takes less than 100 ms for 9 hops which is not so significant.

In the second evaluation, we varied the delegation path length by [3,4,6,8,10] and we measured the average time for achieving both delegations and authorization. Results are shown in Table 3. The average time is the combination of the different task performed by the PDP such as receiving the request, creating and forwarding delegation request to the DVP, exchanging Diameter messages between the domains, performing authorization, and acquiring and validating delegations. The table shows that as the delegation path length increases between 4 and 8 the time for authorization across multiple domains increases approximately by 35 ms. Then, if the delegation path length becomes more than 8, the average time increases faster which may be due to the increase of the number of delegations acquired and validated at each DVP. In addition, the number of exchanged messages is more important as the delegation path increases.

The case of remote providers with an estimated transmission delay of 5 ms, the build of the chain of delegations for 10 hops is equal to: (5 ms × 10 (hops) × 2 (round trip)) + 307.12 (processing time) = 407.12 ms. We can notice that the multi-delegations process over 10 hops takes less than 0.5 second, which is reasonable and not significant for the user to be authorized via the proposed approach.

9. Conclusion

There is an increasing demand and a big need for collaboration in multi-domain environment to enable secure interoperability and collaboration between domains. This paper presented an access control infrastructure enabling collaborative authorization decision by acquiring delegations across cooperative domains. Collaboration in a multi-domain environment is achieved through acquiring delegations across cooperative domains based on a trust relationship established between cooperative domains. Therefore, we extended the XACML access control model with incorporating a new component, the Delegation Validation Point (DVP) to enable to exchange delegation across multiple domains. In addition two Diameter Applications were defined for searching for cooperation path across cooperative domains and exchanging securely XACML/SAML requests and answers for acquiring the required delegations to build the chain of trusted delegations. A test of validation of our implemented prototype achieved successfully the collaboration between multiple domains to acquire delegations and make access control decision. Our evaluation results indicate that both, building the cooperation path and performing the collaborative access control, are achieved in reasonable time. However, the evaluation was conducted in a virtual network environment which did not take consideration of the large dispersion of network domains, nor the processing time of network equipment for the propagation of exchanged messages. Thus, compared to evaluation results, the deployment of the solution in a real environment increases time for achieving collaborative access control that depends on various network parameters.

References


Souheil Ben Ayed received Eng. degree in Computer Networks and Telecommunications from INSAT (Institut National des Sciences Appliquees et de Technologies), University of Carthage, Tunisia and a MSc degree in Telecommunications from Sup’Com (Ecole Superieure des Communications), University of Carthage, Tunisia in 2005 and 2008, respectively. Since October 2009, he has been working toward the PhD degree in computer science at Keio University, Japan. His work is focused on multi-domain access control in distributed environments. From 2005 to 2009, he had the opportunity to work as an engineer in multinational companies in Tunisia. His research interests lie in the areas of computer network, network security, distributed systems and network mobility.

Fumio Teraoka received a master degree in electrical engineering and a Ph.D. in computer science from Keio University in 1984 and 1993, respectively. He joined Canon Inc. in 1984 and then moved to Sony Computer Science Labs., Inc. (Sony CSL) in 1988. Since April 2001, he is a professor of Faculty of Science and Technology, Keio University. He received the Takahashi Award of JSSST (Japan Society for Software Science and Technology) and the Motooka Award in 1991 and 1993, respectively. He also received the Best Paper Award in 2000 from IPSJ (Information Processing Society Japan). His research interest covers computer network, operating system, and distributed system. He contributed to the activity of the Mobile working group of IETF by developing Virtual IP (VIP). He was a board member of the WIDE Project from 1991 to 2010. He was a board member of IPSJ from 2000 to 2002. He was a board member of JSSST from 2005 to 2009. He is a member of ACM, IEEE, JSSST, and IPSJ.