G-Cloud on Openstack : Adressing access control and regulation requirements

Ibrahim Yonis Omar, Romain Laborde, Ahmad Samer Wazan, François Barrère, Abdelmalek Benzekri
Institut de Recherche en Informatique de Toulouse
University Paul Sabatier
Toulouse, France
{Yonis, Romain.Laborde, Ahmad-Samer.Wazan, Francois.Barrere, Abdelmalek.Benzekri}@irit.fr

Abstract— It is well known that e-Government applications bring several benefits to citizens in terms of efficiency, accessibility and transparency. Today, most of governments tend to propose cloud computing based e-services to their citizens. A key component in these services is the access control management issue. In this paper, we present our research works for building an access control system for the Djiboutian e-Government project that is built using Openstack framework. Specifically, we demonstrate the limitation of the integrated access control system in Openstack for the Djiboutian e-Government access control requirements and for the compliance to the related regulation. Thus, we propose to extend the existing access control system of Openstack by integrating the features of the XACML V3 to the Openstack framework.

Keywords—e-Government; e-services security; Access control; Cloud Computing; XACML; Openstack

I. INTRODUCTION

e-Government refers to a way to serve the users of the public service through the use of new Information and Communication Technologies (ICT)[1]. Usage of e-Government systems contributes to government effectiveness by supplying users with easy access to public services as well as reducing operating costs of the administration.

e-Government systems can be classified according to different target areas:

- Government-to-Government (G2G), also known as e-administration, refers to electronic collaboration between different government agencies. E-administration facilitates the sharing of information between agencies,
- Government-to-Citizen (G2C), is the process that electronically provides on-demand and personalized public services to citizens, in a centralized way,
- Government-to-Business (G2B), sets up online relationship between government and the business sector in order to interactively provide information on regulations, advices, and procedures.

In addition, services in e-Government have four levels of maturity [2] 1) *catalogue* level, which provides presence on the web through a simple web site for instance; 2) *transaction* level allowing interactivity between governmental agency and their consumers; 3) *vertical integration* level where integration of scattered systems at different levels is performed, 4) *horizontal integration* level where information obtained by one agency will propagate through out all government functions.

Although the e-Government system can generate productivity gains, its implementation costs a lot in terms of both financial and human resources. Deploying data center for each public administration sector is a significant cost to governments [3]. Also, management of maintenance, security, and upgrade requires as many qualified IT staffs as data centers. Often the failures of e-Government systems projects are due to those costs [4].

To deal with those cost issues and setting up e-Government systems readily, recent e-Government projects are deployed on Cloud Computing [5]. Thanks to on-demand computing and infrastructure, Cloud computing offers rapid scalability and deployment capabilities to e-Government system regardless of the ICT state of administrative agencies [3]. Cloud Computing permits any department within e-Government systems to deploy all kind of eGovernment services by provisioning upstream stacking needs - from infrastructure to software –, as a service.

Our work is done in the context of the Djibouti eGovernment project. Currently, the government of Djibouti has adopted cloud computing for its e-Government named eGovernment Cloud Community (eGCC).

Although cloud computing facilities E-Government development, it brings out also new challenges [7]. Among those, our task focuses on security and privacy issues. Security ranges from the transmission to the storage through identity and access management of information/data in eGCC. Privacy, in turn, is essential to guarantee protection of personal information collected by a government against the derivatives, such as discriminations.

Each agency has its own access control and privacy policies. Due to multi-tenancy of eGCC services and heterogeneousness of agencies' access control and privacy policies, it is impractical to implement a tailored access control model for every policy. As a consequence, a generic and flexible system for enforcing both policy types must be conceived. We propose in this article to follow an Attribute-Based Access Control (ABAC) approach using the OASIS XACML [8] specification.

The article is organized as follows. Section 2 presents the importance of cloud computing for e-Government services and also the security challenges. Section 3 analyses Openstack [12] for implementing eGCC. Section 4 describes the integration of XACMLv3 and Openstack. Finally, we draw our conclusion and perspectives in section 5.

II. CLOUD COMPUTING AND E-GOVERNMENT

Governments are facing two main challenges when it comes to proposing e-Government services to their citizens. First, the amount of information held by governments is too big and increases very quickly. Second, large parts of this data contain sensitive information that must be protected efficiently.

Handling those challenges in a traditional way consists in buying a dedicated IT infrastructure that will be set up for storing the government's information and hosting their e-applications.

This way presents several disadvantages for governments. Governments have to setup different kinds of resources for managing the upgrade of their IT infrastructure in order to meet their increasing needs in terms of performance and storing capacity. Indeed, using the same technology for several years include different risks in terms of storing capacity and availability of e-services. In addition, the costs needed for maintaining the IT infrastructure can be enormous and unpredictable; thus the quality of maintenance can be degraded because of budget issues.

A. Cloud computing

Recently, a new vision of how using and managing the IT resources have been developed. This vision delivers all discipline, technology, and business models, which setup a powerful IT infrastructure and present it as a service for end users. This is called "Cloud computing". The NIST defines cloud computing as:

"*Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction*." [6]

Thus, cloud computing is a set of computing resources that are accessible by end users according to their needs (*on-demand network access to a shared pool*).

From organizations perspective, cloud computing offers many advantages [9]; in particular:

1. Cost savings: Cloud computing enable organizations to reduce their costs in terms of purchase of new equipment, personnel employment and maintenance. Additionally, cloud providers offer interesting options of payment that help organizations save their cost (e.g., "pay as you use").
2. Scalability: a great feature of the cloud computing is its capacity to meet dynamically the increasing needs of organizations.
3. Low maintenance: maintaining an IT infrastructure is always challenging for organizations. By using cloud computing, organizations delegate this issue to cloud providers.

B. Benefits of cloud computing for e-Government

Cloud computing solutions bring many benefits for e-Governments [3]. Cloud computing reduces public agencies investment in term of technologies deployment in order to implement e-Government-based services. Indeed, instead of having a data center for each public agency (that would leads to situations where computer materials are often under or over-used), a central cloud computing constitutes an ultimate solution for e-Governments. Cloud computing aggregates the computer resources of the public administration in one data center. In addition, thanks to virtualization offer, it permits to agencies to have their own data center as a service without worrying about cooling and electrical issues. Thus, governments can concentrate their efforts on the business issues of their e-services.

Cloud computing have three core levels of service [6].

- *Infrastructure as a Service (IaaS)*: At this level, the cloud provider makes available to consumers a set of computing resources capabilities to deploy and run an information system (IS) such as server, operating system, storage and networks.
- *Platform as a Service (PaaS):* is a level aimed at programming and application development environments.
- *Software as a Service – SaaS*: Software is accessible via a web interface. Software is not purchased it is rented.

In addition to service model, Cloud Computing can be deployed according to four models [6].

- *Private cloud*. Cloud infrastructure is provisioned for exclusive use by a single organization comprising multiple consumers (e.g., business units). It may exist on or off premises.
- *Community cloud*. A specific community of consumers from organizations, that have shared concerns, provisions cloud infrastructure for its exclusive use. It may exist on or off premises.
- *Public cloud*. The general public provisions the cloud infrastructure for open use. It exists on the premises of the cloud provider.
- *Hybrid cloud*. The cloud infrastructure is a composition of two or more distinct cloud infrastructures (private, community, or public) that remain unique entities, but are bound together by standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load balancing between clouds).

Obviously, governments prefer the private or/and community deployment models because they give more control to

governments in terms of government data protection. Generally, Governments don't accept the storage of their sensitive data in servers that are located outside of their borders. That's why different countries (e.g., France and UK) have favored the development of a nation-wide cloud, called G-cloud, for their governments [10].

In the Djiboutian e–government project, we have adopted the community deployment model, where the term "community" refers to the Djiboutian government agencies. Also, we have adopted Openstack for implementing the Djiboutian eGCC infrastructure.

C. Security challenges in eGCC

Traditionally public agencies manage internally all aspects of their IS security and privacy, according to a security policy. This security policy is based on risk analysis (RA) of their IT assets. With the eGCC advent, the context changes. Agencies no longer have physical control on their IT assets. This raises new risks such as:

- Transmission: How should we trust the guarantees of secure transmission from eGCC provider?
- Storage: What is the storage architecture of the eGCC? Does it meet our needs?
- Regulations compliance: What about privacy of our data?
- Identity and Access Management: How is managed identity and access management? Does it meet our needs?

To address those issues, agencies update RA by heeding this new eGCC context. Based with the new RA, agencies define security policy to be applied to the eGCC.

From an eGCC provider point of view, complying with the policies of agencies means creating as many personalized resources as agencies. This contradicts the logic of the eGCC, which focuses on the rationalization of resources.

Among the challenges, we focus on those impacting access control requirements.

III. IMPLEMENTING E-GOVERNMENT SECURITY REQUIREMENTS IN OPENSTACK

Openstack is an open source cloud computing system that can be used for public, private, community, and hybrid cloud. In this section, we demonstrate that Openstack lacks some security features for implementing eGCC.

A. Introduction to Openstack

Openstack consists in several services dedicated to specific functionalities. *Nova* is designed to create and manage virtual machines. It is compatible with the majority of virtualization technologies (Xen, KVM, Hyper-V, LXC). Interaction with Nova is performed through an API or a Dashboard (component HORIZON). *Swift* can be defined as a file system. This service stores large amounts of unstructured data via a RESTful HTTP API. *Cinder* provides persistent storage for virtual machines. When a virtual machine is destroyed, this storage is maintained. *Glance* is the registration and delivery service of virtual images for launching virtual machines. Finally, *Keystone* is the service that manages identity and authorization management (IAM).

Keystone centralizes authentication and credentials of users and services. It guarantees the validity of this information by providing tokens that are recognized by the other Openstack services. Keystone does not manage authorization policies of other services. Each service, included Keystone, decides whether to grant access or not to its resources according to its own policy and the received tokens validated by Keystone.

The access control model used by Openstack to specify authorization rules is an extension of RBAC. A user is assigned to a set of roles for given projects. For example, user U owns role R1 for project P1 and role R2 and R3 for project P2. Projects are grouped in domains. Permissions are granted to a role for a project in the context of a domain. For example, permission PERM1 is granted to R1 for project P1 in domain D1.

Figure 1 illustrates the access control messages exchanged when a user tries to access an Openstack service.

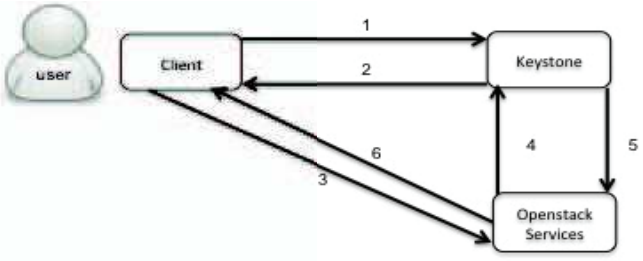

Figure 1. The Openstack authorization process

1. First, a user authenticates to Keystone before using the Openstack service.
2. Keystone sends an *unscoped token* proving the authentication and containing the list of projects associated to the user. Then, the user chooses one of the proposed projects and informs Keystone. Finally, Keystone returns a *scoped token* that includes the list of available services.
3. The client uses *scoped token* every time he makes a request within Openstack services.
4. Openstack services verifies the authenticity of the token with keystone
5. Keystone confirms the validity and gives additional information related to this token (name, roles, membership project).
6. At the end, each service applies its authorization policies.

B. An illustrative scenario

Let consider a public agency AG that consists in three administrative departments D1, D2 and D3. In order to enhance the public service treatments, AG wants to setup a G2G environment. As a consequence, AG creates virtual data center

VDC from an eGCC provider. Each department must have a specific computing infrastructure for its business requirements. Different resources are provided to each department such as servers, networking and storage facilities.

Each department has a dedicated IT manager who administrates these resources or delegates tasks to sub-administrators. Agency AG has defined the following security policies and regulations compliance rules that departments must enforce:

- P1: Employees of AG are permitted to use the services provided by their departments. According to their functions, employees have specific permissions on services.
- P2: Employees of AG cannot execute any administrative task outside working hours. Exception is granted to IT technicians of departments according to their periods of standby duty.

In addition to these policies, AG formulates legal constraints (LCx) to be respected such as:

- LC1: regulation *advices* to encrypt resources of department D1.
- LC2: Due to classification category, regulation *requires* to encrypt resources of department D3.

On the basis of the above use case, requirements below must be filled.

Req1 - There must have flexible access controls models allowing specifying authorization rules according to roles or complex time periods.

Req2 - The authorization system must also include obligations and advices.

Req3 - Administrators must have full control on employees' permissions. Several levels of administrators must exist.

C. Analysis of e-Government security requirements

Analysis of Req1: The current Openstack access control model allows specifying several concepts. This model being an extension of RBAC, *roles* for managing employees already exist. In addition, it is possible to represent a department using the concept of Openstack *project*. Collaboration between departments can be expressed by grouping projects in *domains*. An example that uses domains can be founded in [11]. However, the Openstack access control model does not allow contextual permissions such as "no access outside working hours".

Analysis of Req2: Openstack does not support regulations compliance expression including obligations and advices. As eGovernment service requires compliance with legislative directives, Openstack have to put up capability to express regulation constraints and enforce them dynamically.

Analysis of Req3: Openstack defines the role of project administrator. However, a project administrator cannot create roles attached to its project. Project administrator should contact Openstack provider to perform this task. In addition, there is only one project administrator role and no role hierarchy. As consequence, it is not possible to define several levels of administrators with different administration permissions.

IV. Integrating XACMLv3 into Openstack for covering e-Government requirements

We propose to integrate XACMLv3 to cover these shortcomings of current Openstack.

A. Introduction to XACMLv3

XACML (eXtensible Access Control Markup Language) version 3 is an XML-based specification for access control that has been standardized by OASIS [8]. XACML describes an architecture, an attribute-based access control policy language and a request/response language.

XACML provides a management architecture that describes the different entities and their roles related to the decision making process. A data-flow model describes this architecture that is similar to previous XACML version.

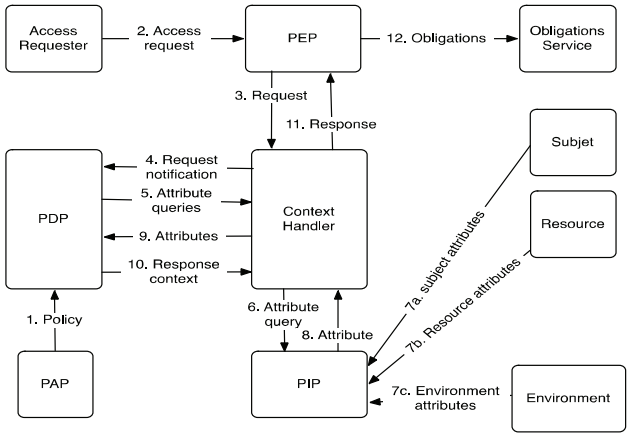

Figure 2. The XACML Architecture [8]

The model operates by the following steps.

1. Policy Administration Points (PAP) write policies and policy sets and make them available to the Policy Decision Point (PDP). These policies or policy sets represent the complete policy for a specified target.
2. The access requester sends a request for access to the Policy Enforcement Point (PEP).
3. The PEP sends the request for access to the context handler in its native request format, optionally including attributes of the subjects, resource, action and environment.
4. The context handler constructs a standard XACML request context and sends it to the PDP.
5. The PDP can request any additional subject, resource, action and environment attributes from the context handler if needed.
6. The context handler requests the attributes from a Policy Information Point (PIP).
7. The PIP obtains the requested attributes.

8. The PIP returns the requested attributes to the context handler.
9. The context handler sends the requested attributes. The PDP evaluates the policy.
10. The PDP returns the standard XACML response context (including the authorization decision) to the context handler.
11. The context handler translates the response context to the native response format of the PEP. The context handler returns the response to the PEP that enforces the authorization decision.
12. If the decision includes obligations, the PEP fulfills them.
13. Finally, the PEP enforces the authorization decision.

The XACML policy language is used to describe general access control requirements in terms of constraints on attributes. Specifically, attributes could be any characteristics of any category such as the subject, the resource, the action, or the environment in which the access request is made. Attributes have an identifier, which is a Uniform Resource Name (URN), and a data type also identified by a URN. Considering attributes makes the language very flexible. Moreover, XACML language is natively extensible.

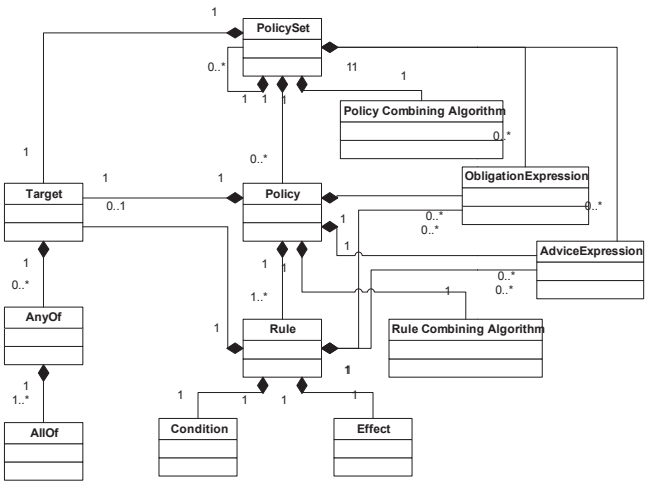

Figure 3. The XACMLv3 policy language mode [8]

An XACML policy is composed of:

- a *target* element which is a first filter for searching the applicable policy
- a set of *obligation expressions* that are instantiated when a matching request is processed. PEPs must enforce obligations.
- A set of *advice expressions* that are instantiated when a matching request is processed. An advice is similar in its form to an obligation. However, PEPs may or may not enforce an advice.
- a set of *rules* which are expressions to determine if a request is denied or permitted. A *rule* contains a *target* and may include *obligations* and *advices* specific to this rule.
- Policies can be grouped in *policy sets*.

B. Expression of e-Government policies in XACMLv3

XACMLv3 being based on attributes of categories, the language is very flexible. It is possible to consider any security characteristic by identifying using a new URN. For example, the employees' department can be represented by attribute *dept* of category *access-subject*. Example in Figure 4 Is the XACMLv3 expression of policy P1. It states that if attribute *role* of category *access-subject* is equal to Employee (lines 21-28) and attribute *dept* of category *access-subject* is equal to attribute *dept* of category *resource* (lines 33-43) then permission is granted. Policy P2 can be represented by adding another rule with contextual constraints using standard XACML attribute "urn:oasis:names:tc:xacml:1.0:environment:current-time". More examples of contextual policies in XACML can be found in [13] and [14].

```
<Rule Effect="Permit" RuleId="Employee-access-their-dept-resource">
    <Target>
        <AnyOf>
            <AllOf>
                <Match MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
                        Employee
                    </AttributeValue>
                    <AttributeDesignator AttributeId="urn:siera:role"
                        Category="urn:oasis:names:tc:xacml:1.0:subject-category:access-subject"
                        DataType="http://www.w3.org/2001/XMLSchema#string" MustBePresent="true"/>
                </Match>
            </AllOf>
        </AnyOf>
    </Target>
    <Condition>
        <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
            <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-one-and-only">
                <AttributeDesignator AttributeId="urn:siera:dept"
                    Category="urn:oasis:names:tc:xacml:3.0:attribute-category:resource"
                    DataType="http://www.w3.org/2001/XMLSchema#string" MustBePresent="true"/>
            </Apply>
            <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-one-and-only">
                <AttributeDesignator AttributeId="urn:siera:dept"
                    Category="urn:oasis:names:tc:xacml:1.0:subject-category:access-subject"
                    DataType="http://www.w3.org/2001/XMLSchema#string" MustBePresent="true"/>
            </Apply>
        </Apply>
```

Figure 4. Specification of P1 in XACMLv3

In addition, XACML natively expresses obligations and advices [15]. This offers the way to specify legal constraints. For instance, Figure 5 is the XACML translation of legal constraint LC2. It states that when access is granted (FulfillOn="Permit" at line 67), obligation "encrypt-data" must be enforced. This obligation has two parameters: 1) data-id that represents the name of the resource to encrypt and 2) "algo" that indicates the name of the security algorithm to apply. Here the obligation can be translated as "encrypt the requested data using AES-128".

```
<ObligationExpression ObligationId="urn:siera:encrypt-data" FulfillOn="Permit">
    <AttributeAssignmentExpression AttributeId="urn:siera:encrypt-data:data-id">
        <AttributeDesignator
            AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
            Category="urn:oasis:names:tc:xacml:3.0:attribute-category:resource"
            DataType="http://www.w3.org/2001/XMLSchema#string" MustBePresent="true"/>
    </AttributeAssignmentExpression>
    <AttributeAssignmentExpression AttributeId="urn:siera:encrypt-data:algo">
        <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
            AES-128
        </AttributeValue>
    </AttributeAssignmentExpression>
</ObligationExpression>
```

Figure 5. Specification of LC2 in XACMLv3

C. *Integration of XACML and Openstack*

Keystone stores all credentials of users and resources information. Then, it can be considered like a PIP in the XACML architecture. In consequence, we propose to integrate XACML to the authorization process as follows. It is compliant with both XACML and Openstack architectures.

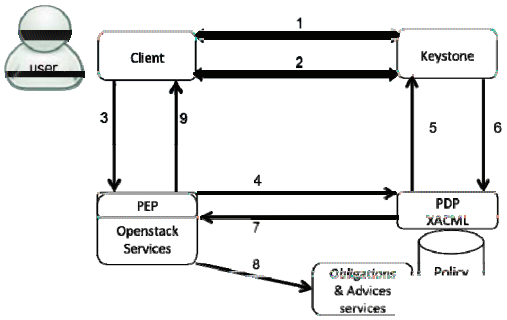

Figure 6. Integration of XACML and OpenStack

First, the user authenticates, chooses its project and sends its scoped token to request a service like in the classical Openstack access control process (steps 1, 2 and 3). The service, playing the role of PEP sends XACML request including the scoped token to the XACML PDP service (step 4). After receiving this request the PDP calls Keystone to validate the scoped token and receive additional information (step 5). The PDP takes its decision according to the policy and sends it to the requesting service (step 7). When the requesting service gets the decision it applies it. If the decision contains an obligation or an advice, it sends the XACML obligations/advices to a dedicated service.

V. CONCLUSION

In this paper, we have presented our research works for building an access control system for the Djiboutian e-Government project that is built using an Openstack framework. First, we have demonstrated the advantages of cloud computing solutions for E-Government applications. We have studied the Openstack framework and identified the limitations for enforcing access control and regulations requirements. To cope with this issue, we have proposed to integrate XACMLv3 to Openstack since it provides features like a generic and extensible language, obligations/advices expressions.

We are currently implementing this architecture. Afterwards, we will measure the performance of our prototype in the Djiboutian environment in order to prove the feasibility of the solution. Another issue is to manage the inconsistency between different rules from agencies and/or regulations. Different stakeholders (agencies, legislative power, etc) are involved in the eGCC system and will produce access control/regulation rules. How t
administration [16], especiall
these rules?

VI. REF

[1] Field, T. (Ed.). (2003). OE
Government Imperative. OECD

[2] Layne, K., & Lee, J. (2001). Dev
A four stage model. Government

[3] Jia, Y., Yuchun S., Tong X., Jir
G2G E-Government Informa
information Sciences and Servic

[4] Cellary, W., & Strykowski, S. (
on cloud computing and service-
the 3rd international conference
governance (pp. 5-10).

[5] Zwattendorfer, B., Stranacher, K
Cloud Computing in E-Govern
Enabled Innovation for Democr
181-195).

[6] Mell, P. and Grance, T. (20
computing.

[7] Wyld, D. C. (2009). Moving t
computing in government. IBM

[8] OASIS Standard. (2013) eXtens
(XACML) Version
open.org/xacml/3.0/xacml-3.0-cc

[9] Hashemi, S., Monfaredi, K.,
Computing for E-Government
Academy of Science, Engineerin
of Computer, Information Scienc

[10] Zwattendorfer, B., Stranacher,
(2013). Cloud Computing in
Technology-Enabled Innovatio
Governance (pp. 181-195).

[11] Tang, B., & Sandhu, R. (2014
with domain trust. In Network an

[12] Openstack URL: https://wiki.c
access March 2015.

[13] Laborde R., Kamel M., Wazan
secure collaborative web-based e
International Journal of Web
Publishers, Special Issue on I
Information Society, Vol. 5 N. 2,

[14] Kabbani B., Laborde R., Barrèr
and Enforcement of Dynamic
Situations. In IFIP Internationa
Mobility and Security (NTMS 2
1-6, avril 2014.

[15] Laborde R., Kabbani B., Barrèr
XACMLv3 policy enforcement
Systems for Communication
workshops, p. 620-625, juillet 20

[16] Nasser B., Laborde R., Benzek
Access Control Model for
Organizations. In On the Move
OTM 2005 workshops. MIOS
LNCS 3762, p. 537-551, 2005.