Usage Control in Cloud Federations

Gaetano F. Anastasi*, Emanuele Carlini*,Massimo Coppola*, Patrizio Dazzi*,
Aliaksandr Lazouski†, Fabio Martinelli†, Gaetano Mancini†, Paolo Mori†
*ISTI-CNR, Pisa, Italy
{gaetano.anastasi,emanuele.carlini,massimo.coppola,patrizio.dazzi}@isti.cnr.it
†IIT-CNR, Pisa, Italy
{aliaksandr.lazouski, fabio.martinelli, gaetano.mancini, paolo.mori}@iit.cnr.it

***Abstract*—Cloud Federation is a promising approach to enhance cross-cloud application execution. Nevertheless, such approach emphasizes open challenges in Cloud Computing, such as revoking long-lasting authorization on resources as soon as conditions granting the access right are no longer valid. To tackle this kind of issues, we built a prototype of Cloud Federation that leverages the concept of Usage Control (UCON), by continuously monitoring and reassessing the users right on resources. We exploited an extension of the XACML standard and measured the overhead caused by different security policies and distributions of requests. Results suggest that the UCON model can be effectively applied in Cloud Federations and its performance is sustainable when applied to the relevant actions of the lifecycle of applications.**

I. Introduction

Cloud Computing is a computational paradigm where infrastructures, applications and platforms are offered according to a pay-per-use cost model. Cloud Computing offers unquestionable benefits to users, and represents a valid asset for large IT companies. Unfortunately most Cloud providers force their users to operate according to specific models, for instance in terms of communication protocols and virtualization technologies. This leads to the *vendor lock-in*, which precludes moving user data to a different provider and modifying user applications to leverage different service interfaces. Moreover, costs related to the lack of standardization hinder transition opportunities and interoperability across providers.

The Cloud Federation approach [3] aims at providing a unified platform for managing services and resources, providing elasticity beyond the scale of the data center and effectively enabling a market-based approach. From a practical point of view, a Cloud Federation can be considered as a bridge linking cloud users and cloud providers, dealing with the heterogeneity of providers and allowing users to exploit multiple providers at the same time. In our vision, shared also in the Contrail research project (see http://contrail-project.eu/), a Cloud Federation must go beyond mere interface adaptation and act as a mediator between users and providers [6].

Such approach brings to light many challenges that are still not solved in Cloud Computing, such as scalability, interoperability and security. For instance, traditional access control models are commonly adopted to define the authorization support in the Cloud, but they are inadequate to cope with long lasting accesses as they grant permissions based on the user rights at the time of the initial request. We believe that this problem can be solved by adopting the Usage Control (UCON) model, defined by Sandhu and Park [19], [25], that permits to define policies containing conditions that must be satisfied all the time during the access (continuous control). Consequently, the access to a resource can be interrupted as soon as those conditions no longer hold. In Cloud Federations, additional aspects must be taken into account for authorization, such as the difficult to detect and manage violations for a multi-tenancy federated platform, and the fact that security policies involve both factors that can only be evaluated at the federation level, and factors that need to be evaluated at the provider level.

In the following we will clearly motivate the need of adopting the UCON model in Cloud Federation. The Contrail approach to Cloud Federation will be used as a case study but, in our opinion, this work can be easily applied to other models sharing the same vision.

A. Motivation and Contribution

The adoption of the UCON model in the design of an authorization system for Cloud Federation is meant to regulate the usage of cloud resources at the federation level, for enforcing security policies that take into account global goals. For example, a security policy could state that guest users (i.e., users that are trying the system, and they did not complete the registration process yet) can execute their applications on the Cloud Federation as long as its workload is low. On the one hand the Federation wants guest users to exploit its resources to acquire new customers, but on the other hands it prefers to keep some resources free in order to be able to promptly react to a computational peaks due to regular user activities. Hence, as soon as the federation workload goes beyond safety thresholds, guest users' applications are suspended regardless of the workload of the provider where such applications were executed.

The released resources may be used for migrating some virtual machines of regular users that are running on overloaded resources. This kind of policy can be enforced at federation level only, because single Cloud providers are not aware of the federation workload, and they are not interested in freeing their resources in advance (hence reducing their revenues) to reduce the overall workload of the Federation in order to minimize the risk of violating federation Service Level Agreements (SLAs).

The main contributions of the paper are: (a) the definition of a Cloud Federation model based on continuous control for authorization. By considering Contrail as case study, we design the integration between components managing the application life-cycle and security ones; (b) the description of the implementation of the UCON authorization system targeting federated Clouds. To the best of our knowledge, this is the first application of the UCON model to Cloud Federation; (c) an analysis of performance results aimed at showing the scalability of the system.

The paper is structured as follows. In Section II related work is discussed, whilst Section III describes the reference architecture for supporting Cloud Federation. Section IV describes the proposed authorization support based on UCON and Section V shows an evaluation of our implementation. Section VI concludes the paper.

II. Related Work

To the best of our knowledge, none of the existing solutions for Cloud Federation exploits an authorization system explicitly realized according to the UCON model, as instead we do in our approach. As a consequence, the related work is split in two parts, one for Cloud Federation and one for Usage Control in Clouds.

The lack of standardized meaning for the Cloud Federation term has led to multiple conflicting definitions [2]. We refer to the most notable approaches that can be compared with ours. *InterCloud* [3] is one of the first works that advocates the need of federated Clouds and validates its approach by means of CloudSim [4], a framework for Cloud modeling and simulation. *SmartFed* [1] has been also implemented on top of CloudSim for simulating Cloud Federations. However these simulation frameworks do not deal with any security related aspects. Other work exists in this field, such as *Reservoir* [20], *Sky* [12] and the work by *Celesti et al.* [7]. Simplifying, such architectures can be considered as "Horizontal Federations", where each Cloud provider is an an autonomous entity that can cooperate with peers for federating together. Instead, Contrail adopts a "Vertical Federation" approach, where the focus is on the provisioning of a vertical integrated Cloud stack that cover both the PaaS and the IaaS levels. Thus, the Contrail federation is a kind of super-entity that exploits provider resources for executing user submitted applications. Among the cited architectures, only the work by Celesti et al. specifically deals with security management but that approach is limited to access control and does not consider continuous control.

Regarding authorization on Clouds, *Gouglidis et al.* [13] survey access control requirements for Cloud and Grid computing, claiming that the UCON model is the best candidate to address those requirements. *Danwei et al.* [11] and *Tavizi et al.* [21] refine this idea and propose two architectures for the enforcement of UCON policies in the Cloud. However, they do not provide any implementation of the presented models. Recently, the UCON model has been successfully adopted in other distributed systems, e.g. Computational Grids [24], [16], [8]. *Sandhu et al.* [24] propose the adoption of their model in collaborative computing systems and study which UCON features can be modeled using XACML. Concerning our past contribution to the field [16], [8], we adopted the UCON model in the Grid to protect Virtual Organization resources from users. In such a context, we proposed U-XACML [9], an extension of the XACML language for expressing UCON features. We also provided a reference architecture [15] for the enforcement of U-XACML policies and a preliminary attempt [14] of integrating an U-XACML authorization system within OpenNebula [17].

III. Contrail Cloud Federation

In the Contrail research project we have conceived a Cloud Federation as a coherent set of Cloud providers sharing a common set of rules, policies, and mechanisms for homogenizing the management and the exploitation of their hardware and software resources. The software module that realizes this homogenization goes under the name of *federation-support* or simply *federation*. For the reader's convenience, this section briefly describes the architecture of the Contrail federation [5], which we refer for presenting our approach.

Contrail Architecture. The high-level components of the Contrail federation are briefly described in the following. The *Web and Programming Interfaces* module represents the multi-tenancy front-end of the federation and exposes the interfaces for accessing to all the resources owned by federated providers. A federation-level account allows users to fully exploit the federation functionalities, like submit applications that can be executed in one or more federated providers. The *Identity Management* module is in charge of mapping a federation-level identity with the corresponding provider-level identities. The information on user's accounts and other metadata are stored in the *Data Store* component. Usually, applications can be submitted to Clouds following very

different approaches and each cloud provider can in principle support a different degree of expressiveness for the application description. In Contrail, applications are described by means of the Open Virtualization Format (OVF) specification [18]. The OVF format describes applications as a hierarchical set of nodes, each one composed by Virtual Machines (VMs) interconnected by virtual networks. Both VMs and networks can be characterized by requirements that relate to the functional aspects of the application. The *SLA* module deals with non-functional requirement, which are usually specified by means of SLAs through QoS properties.

The *Application Execution and Runtime Management (AEM)* is the module that considers the suitability of multiple providers to support a single application by considering different criteria, such as the minimization of economical cost and the maximization of performance levels. It also controls the application life-cycle and management, which may include VM migrations and increasing/decreasing the degree of parallelism. To performs its tasks, the AEM relies upon an abstraction level provided by the *Cloud Adapters*. In particular, the adapters offer a common provider-agnostic interface to the federation, permitting to support different providers by simply implementing drivers for each provider, in order to perform provider-specific operations. Additionally, the adapters provide federation-wide functionalities for the management of inter-cloud operations, such as setting up virtual networks and inter-cloud storage services.

Security in Contrail. The security support in Contrail consists of two main components: *authentication* and *authorization*. The Contrail authentication component [10] supports federated identities. It allows users to authenticate on the federation exploiting the credentials they already own, provided that those credentials have been released by trusted organizations. Authentication includes a Certification Authority and supports delegation, for allowing the federation to act on behalf of users when interacting with providers. The authorization component, instead, is based on the UCON model, and it checks that the access right to a cloud resource is valid at request time and also during its usage, according to the security policy defined at the federation level. This support is the core contribution of this work and it is described in the following.

IV. Authorization Support

The authorization support we propose for the Cloud federation is based on the UCON model [19], [25]. It introduces new features in the decision process w.r.t. traditional access control, such as (i) *mutable attributes* of subjects and objects and, as a consequence, (ii) the *continuity of policy enforcement*. Mutable attributes describe features of subjects and objects that change due to the decision process, e.g., users' reputation and resources' workload. Mutable attributes lead to the need of continuously monitor their values, and re-evaluate the security policy to guarantee that the right of using the resource holds while the access is in progress.

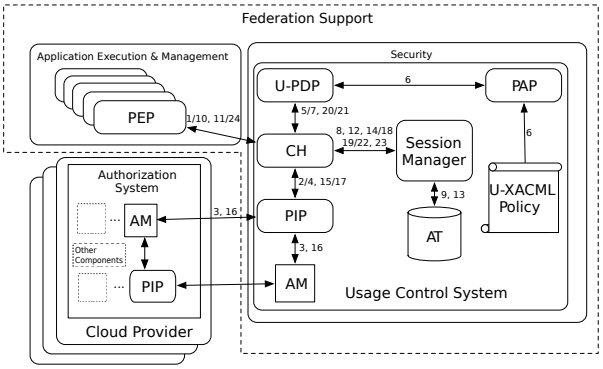

Figure 1. Integration of UCON System within Cloud Federation

This model can be successfully adopted in case of long-standing accesses because the decision process is performed continuously during the access time. The *pre decision* phase corresponds to traditional access control, where the decision process is performed at the request time. The *ongoing decision* phase, instead, is executed after the access is started, ends when the access terminates, and implements the continuity of control that is a specific feature of UCON.

If the decision process detects a policy violation while an access is in progress, this access is revoked and resources are released.

A. Usage Control System Architecture

The UCON system architecture for the Cloud federation is shown in Figure 1. It extends the common authorization systems architecture [22], [23] to deal with a continuous policy enforcement. The main components are described in the following. The *Policy Enforcement Point (PEP)* is integrated within the *AEM* component (see Section III), which is the federation component implementing security relevant actions regulated by security policies. The PEP intercepts these actions, asks the UCON system to evaluate security policies, and enforces resulting decision. The *Context Handler (CH)* is the front-end of the UCON system, that triggers the access decision process. The *Policy Information Point (PIP)* retrieves mutable attributes needed to perform the access decisions process. The PIP contacts the Attribute Manager(s) (AMs), to obtain fresh values of the required attributes. The *Policy Administration Point (PAP)* stores and manages U-XACML policies. Conversely, the *Policy Decision Point (U-PDP)* evaluates such policies to produce the decision for each access request. The

Access Table (AT) keeps metadata regarding accesses in progress, such as status of current sessions, IDs of related attributes and cached values. Finally, the *Session Manager (SM)* manages usage sessions. It manages the ongoing decision phase by monitoring the value of mutable attributes. When the values of some attributes change, the SM triggers the access re-evaluation of all usage sessions that exploit these attributes. When a decision turns to "deny", the corresponding sessions must be revoked.

B. Authorization Workflow

The authorization workflow starts when federation users trigger security relevant actions, such as the execute application. The PEP intercepts invocations of such actions and it sends a *tryaccess* message to the CH (step 1 in Figure 1). The CH retrieves the values of relevant attributes for the decision process by sending the *attr query* message to the PIP (step 2) that, in turn, contacts the relevant AMs (step 3) and sends back these values to the CH, through the message *attr value* (step 4). Then, the CH sends the access *request* to the U-PDP, by including previously collected attributes (step 5). The U-PDP loads the U-XACML policy from the PAP (step 6), evaluates the policy and replies with the *response* to the CH (step 7).

Now, let us suppose the policy permits the execution of the requested action. In this case, the CH sends the *create entry* message to the SM for creating an entry that represents the new usage session in the AT (steps 8 and 9). Finally, the CH replies to the PEP with the *permitaccess* message (step 10).

When the access has begun (e.g., a user started an application), the PEP sends the *startaccess* message to the CH (step 11), that sends the message *update entry* to the SM (step 12). The SM contacts the AT to change the usage session status from *pending* to *active*, and triggers the evaluation of the ongoing access for the first time (step 13). Hence, the SM starts the continuous policy re-evaluation loop and sends the *attr query* message through the CH to the PIP to get fresh values of relevant attributes for this access (steps 14-18). If collected values by AMs differ from cached ones, the SM contacts the CH sending the *policy reevaluation* message (step 19); the CH translates the message for the U-PDP that performs the re-evaluation of the access right (steps 20 and 21). The CH forwards this answer to the SM (step 22). If the decision included in the *response* message is *permit*, the SM performs ongoing attribute updates contained in the U-PDP reply and continues the policy enforcement loop (steps 13-22). Instead, if the content of the *response* message sent by the U-PDP is *deny*, the SM sends the *revokeaccess* message to the CH (step 23) which forwards it to the PEP responsible for forcing the access revocation (step 24).

C. Usage Control for Federation Applications

To perform usage control in a federation requires the integration of the authorization workflow with the operations executed by the federation. In turn, these actions depend on the federation management of the application life-cycle. Figure 2 depicts the transition graph of the application life-cycle, whose detailed description is omitted for brevity.

In order to explain how security relevant actions of the federation can be performed by leveraging the UCON System, we concentrate on the application execution operation, denoted as `execute-app`. The problem requires to identify which UCON messages must be sent in the corresponding actions that generates state transitions in the application life-cycle. The `execute-app` security relevant actions begin when the user performs the *Instantiate* (see Figure 2) action to prepare the application for the execution. In this case, the PEP sends the *tryaccess* message to the UCON System, to get the authorization to perform such action. If the permission is granted, the user performs the *Start* action to actually start the application and, consequently, the PEP sends the *startaccess* message to the UCON System to begin the continuous enforcement of the corresponding policy, defining the rights of users to perform (long lasting) actions on the basis of users' and resources' attributes. When the application terminates, or it is terminated by the user, the PEP sends the *endaccess* message to the UCON System to notify that the `execute-app` is terminated. While the application is RUNNING, the UCON System continuously checks that the policy is satisfied and, as soon as the policy is violated, it sends the *revokeaccess* message to the PEP. In this case the UCON System has the responsibility to send a message to the PEP that, in turn, executes an action that changes the state of the application in the life-cycle. To evaluate policies, the UCON System uses attributes that can be provided either by the federation or by Cloud providers. Examples of attributes provided by the federation are the overall load of the federation and the reputation of the users. Rather, the single Cloud provider can provide only local attributes, like the residual computational capacity.

V. Performance Evaluation

This section describes the performances achieved by the prototype of the authorization system for the Contrail Federation, realized according to the architecture described in Figure 1. The prototype has been implemented using Web Services (WS) developed by leveraging Axis2[1]. Access requests, responses, and attribute queries were encoded into messages compliant with the

[1] http://axis.apache.org/axis2

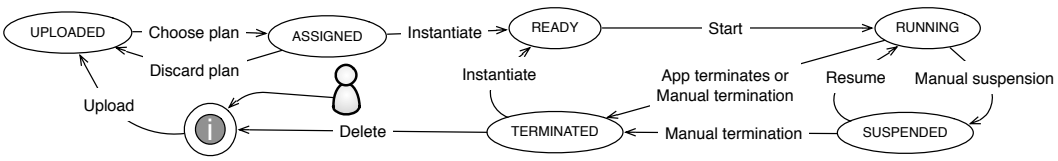

Figure 2. State Diagram of the Application Life-cycle

standard "SAML 2.0 profile of XACML". For the CH implementation we used OpenSAML2.0[2], whilst the Sun XACML Engine[3] and WSO2 Balana XACML Engine[4] have been adopted to evaluate U-XACML policies.

First, we measured the overhead which occurs during the pre-authorization phase as a result of the access request construction, attributes retrieval, and evaluation of the policy against the access request. The overhead is measured as the sum of the following time intervals: t_{pepOut}, t_{attr}, t_{pdp}, t_{pepIn}. The time t_{pepOut} is needed for building a UCON request in SAML/XACML and send it from the PEP to the UCON System, whilst the time t_{pepIn} is needed for building the response and receiving it. Both intervals have been measured in the order of ten ms. The t_{attr} is the time spent by the UCON System for processing the access request, building a SAML Attribute Query, retrieving fresh attributes from PIP and constructing the final XACML request. The time t_{pdp}, instead, is needed to evaluate such request against the U-XACML policy and get the access response. The attribute retrieval time contributes the most to the overhead and it slightly grows with the number of attributes, as can be seen in the top line of Figure 3. Regarding the t_{pdp}, we also noticed a linear growth by increasing the number of attributes in the policy, as can be seen in the middle line of Figure 3. However, such trend is most notable when the Sun XACML Engine was adopted for policy evaluation, whilst the adoption of the Balana XACML Engine allowed for drastically dropping down the t_{pdp} (see bottom line of Figure 3). Thus, we exploited Balana as the primary engine.

Second, the scalability of the UCON System in the Contrail Federation has been measured regarding the continuous control phase. A scenario with many providers has been considered and each provider executes several applications, i.e., resources, to be continuously controlled. The UCON System serves N concurrent sessions, with $N = N_P * N_R$, where N_P is the number of providers and N_R is the number of resources per provider. We assume that resources are distributed uniformly among providers. From the Federation prospective, it is interesting to measure the elapsed time between

[2]http://www.bccs.uib.no/~hakont/SAMLXACMLExtension
[3]http://sunxacml.sourceforge.net
[4]http://xacmlinfo.com/category/balana/

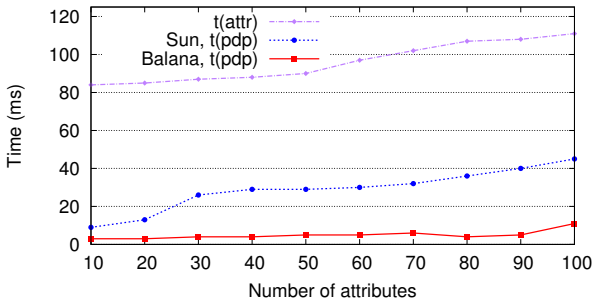

Figure 3. Overhead of Attribute Retrieval and Decision Process

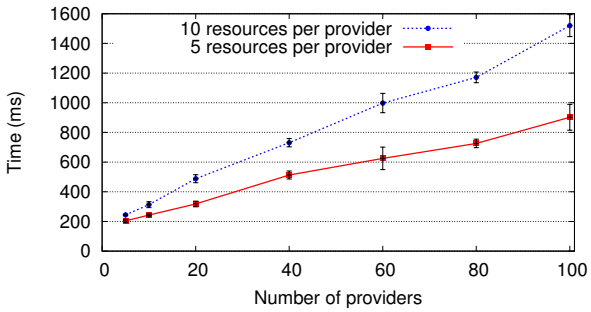

Figure 4. Time for Revoking Ongoing Accesses by Varying N_P

the instant when attributes change their values (and thus usage sessions must be revoked) and the instant until the Federation actually starts the revocation. Such time, indicated as t_{revAll} is due to the policy re-evaluation by the UCON System and the delivery of revoke access messages to corresponding PEPs. Top line of Figure 4 shows how t_{revAll} depends on N_P when $N_R = 10$. Bottom line of 4 shows how t_{revAll} depends on N_P when $N_R = 5$. It can be noticed that t_{revAll} is moderate and increases with the growth of N. The maximum average value, measured for $N_P = 100$ and $N_R = 10$, corresponds to 1520ms (with 74ms as standard deviation) for the UCON System to get the attributes violating the policy, to re-evaluate the policy and to broadcast the revoke access message to all corresponding PEPs in the federation. Also, results suggest that t_{revAll} does not depend on how the load in the Cloud Federation is distributed among providers. For instance, if there are 400 concurrent sessions, t_{revAll} for $N_P = 80$ and $N_R = 5$ (726ms) is almost equal to t_{revAll} for $N_P = 40$ and $N_R = 10$ (731ms). Then,

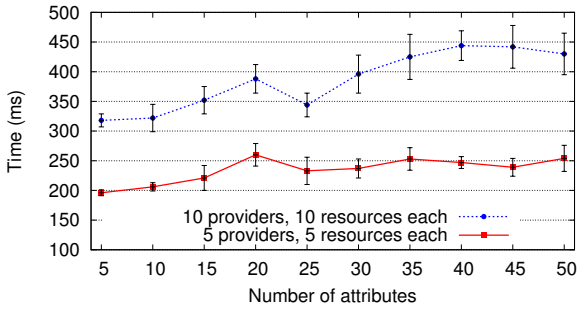

Figure 5. Time for Revoking Ongoing Accesses by Varying N_A

we measured how t_{revAll} changes if all applications are executed by one provider and we noticed that t_{revAll} is of the same range of previous cases.

Finally, we measured how t_{revAll} depends on the number of attributes N_A needed for the access evaluation. We varied N_A from 5 to 50 and considered two configurations of load distribution: (i) $N_P = 5$ and $N_R = 5$, (ii) $N_P = 10$ and $N_R = 10$. Figure 5 shows a moderate growth of t_{revAll} with N_A. In percentage terms, the growth is similar for both configurations.

VI. Conclusion

This paper presented the authorization system adopted in the Contrail Cloud Federation. Such approach is based on the UCON model and, to the best of our knowledge, it is the first work that applies UCON to Cloud Federation. UCON allows for a more effective authorization control, especially on long-lasting accesses to federation resources, permitting to interrupt ongoing accesses as soon as they violate security policies.

To evaluate the feasibility of this approach, some tests have been performed with our implementation, showing an acceptable scalability for realistic setups.

Acknowledgment

The authors acknowledge the support of Projects: FP7-257438 Contrail: Open Computing Infrastructures for Elastic Services and FP7-256980 NESSoS: Network of Excellence on Engineering Secure Future Internet Software Services and Systems

References

[1] G. F. Anastasi, E. Carlini, and P. Dazzi. Smart cloud federation simulations with cloudsim. In *Proc. of the 1st ACM workshop on Optimization techniques for resources management in clouds*, ORMaCloud '13, pages 9–16, New York, NY, USA, 2013. ACM.

[2] D. Bermbach, T. Kurze, and S. Tai. Cloud federation: Effects of federated compute resources on quality of service and cost. In *Cloud Engineering (IC2E), 2013 IEEE International Conference on*, pages 31–37, 2013.

[3] R. Buyya, R. Ranjan, and R. N. Calheiros. Intercloud: Utility-oriented federation of cloud computing environments for scaling of application services. *Algorithms and Architectures for Parallel Processing*, 6081/2010(LNCS 6081):20, 2010.

[4] R. Calheiros, R. Ranjan, A. Beloglazov, C. De Rose, and R. Buyya. Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and Experience*, 41(1):23–50, 2011.

[5] E. Carlini, M. Coppola, P. Dazzi, L. Ricci, and G. Righetti. Cloud federations in contrail. In *Euro-Par 2011: Parallel Processing Workshops*, pages 159–168. Springer, 2012.

[6] R. G. Cascella, L. Blasi, Y. Jgou, M. Coppola, and C. Morin. Contrail: Distributed Application Deployment under SLA in Federated Heterogeneous Clouds. In *Future Internet Assembly*, pages 91–103, 2013.

[7] A. Celesti, F. Tusa, M. Villari, and A. Puliafito. How to enhance cloud architectures to enable cross-federation. In *3rd Int. Conf. on Cloud Computing*, pages 337–345. IEEE, 2010.

[8] M. Colombo, A. Lazouski, F. Martinelli, and P. Mori. Controlling the usage of grid services. *International Journal of Computational Science*, 4(3):373–386, 2009.

[9] M. Colombo, A. Lazouski, F. Martinelli, and P. Mori. A proposal on enhancing XACML with continuous usage control features. In *proceedings of CoreGRID ERCIM Working Group Workshop on Grids, P2P and Services Computing*. Springer US, 2010.

[10] M. Coppola, P. Dazzi, A. Lazouski, F. Martinelli, P. Mori, J. Jensen, I. Johnson, and P. Kershaw. The contrail approach to cloud federations. In *Proc. of The International Symposium on Grids and Clouds (ISGC)*. Proceedings of Science, 2012.

[11] C. Danwei, H. Xiuli, and R. Xunyi. Access control of cloud service based on UCON. In *Proc. of the 1st International Conference on Cloud Computing*. Springer-Verlag, 2009.

[12] A. A. Falasi, M. A. Serhani, and S. Elnaffar. The sky: A social approach to clouds federation. *Procedia Computer Science*, 19(0):131 – 138, 2013.

[13] A. Gouglidis and I. Mavridis. On the definition of access control requirements for grid and cloud computing systems. In *Networks for Grid Applications*, volume 25 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 19–26. Springer Berlin Heidelberg, 2010.

[14] A. Lazouski, G. Mancini, F. Martinelli, and P. Mori. Usage control in cloud systems. In *Proceedings of The 7th International Conference for Internet Technology and Secured Transactions (ICITST-2012)*, pages 202–207. Infonomics Society, 2012.

[15] A. Lazouski, F. Martinelli, and P. Mori. A prototype for enforcing usage control policies based on XACML. In *Trust, Privacy and Security in Digital Business*, volume 7449 of *Lecture Notes in Computer Science*, pages 79–92. Springer Berlin, 2012.

[16] F. Martinelli and P. Mori. On usage control for grid systems. *Future Generation Computer Systems*, 26(7):1032 – 1042, 2010.

[17] D. Milojičić, I. M. Llorente, and R. S. Montero. Opennebula: A cloud management tool. *Internet Computing, IEEE*, 15(2), 2011.

[18] Open Virtualization Format Specification, Version 1.1. Specification, DMTF, Jan. 2010.

[19] J. Park and R. Sandhu. The $UCON_{ABC}$ usage control model. *ACM Transactions on Information and System Security*, 7, 2004.

[20] B. Rochwerger, D. Breitgand, E. Levy, A. Galis, K. Nagin, I. M. Llorente, R. Montero, Y. Wolfsthal, E. Elmroth, and J. Caceres. The reservoir model and architecture for open federated cloud computing. *IBM Journal of Research and Development*, 53, 2010.

[21] T. Tavizi, M. Shajari, and P. Dodangeh. A usage control based architecture for cloud environments. In *Parallel and Distributed Processing Symposium Workshops PhD Forum (IPDPSW), 2012 IEEE 26th International*, pages 1534 –1539, may 2012.

[22] J. Vollbrecht, P. Calhoun, S. Farrell, L. Gommans, G. Gross, B. d. Bruijn, C. d. Laat, M. Holdrege, and D. Spence. AAA Authorization Framework. 2000.

[23] XACML. eXtensible Access Control Markup Language (XACML). http://www.oasis-open.org/committees/xacml.

[24] X. Zhang, M. Nakae, M. J. Covington, and R. Sandhu. Toward a usage-based security framework for collaborative computing systems. *ACM Transactions on Information and System Security*, 11(1):1–36, 2008.

[25] X. Zhang, F. Parisi-Presicce, R. Sandhu, and J. Park. Formal model and policy specification of usage control. *ACM Transactions on Information and System Security*, 8(4):351–387, 2005.