Zentrum für sichere Informationstechnologie – Austria
Secure Information Technology Center – Austria

A-SIT

Seidlgasse 22 / 9, 1030 Wien
Tel.: (+43 1) 503 19 63–0
Fax: (+43 1) 503 19 63–66

Inffeldgasse 16a, 8010 Graz
Tel.: (+43 316) 873-5514
Fax: (+43 316) 873-5520

http://www.a-sit.at
E-Mail: office@a-sit.at

DVR: 1035461
ZVR: 948166612
UID: ATU60778947

MULTILATERAL INTEGRATION AND MANAGEMENT OF XACML POLICIES

Version 1.1 from 20.11.2017

Bojan Suzic – bojan.suzic@a-sit.at

Abstract: XACML is a framework that establishes an XML-based language, architecture and process models for the externalized authorization management. XACML has been by its design focused on the protection of information assets present in the realm of a single organization. Because of this narrow scope, a range of tools and methodologies emerged that support the application of XACML policies to a single environment only. In this project, motivated by this limitation, we extend the management model of XACML to support the transparent application in different, interacting and integrating environments. We consider this challenge from the perspective of the ongoing global transformation both on business and on technical levels that increasingly requires the entities to be interconnected and exchange the data in a continuous and uninterrupted manner. In such highly interdependent environment, the consideration and execution of security policies with the focus on an isolated realm become not only costly to implement and maintain, but also incurs additional overhead to impose security control.

Contents

1. Introduction

The overall cloudification of services and an increasing amount of data that is stored in, delivered from, or exchanged between different systems deployed in the cloud requires rethinking existing paradigms of security and privacy management. Traditional models and case-scenarios assume the user's data to be stored and processed primarily at a single entity. In contrast to that, emerging business models assume the dynamic processing and reuse of data in complex multi-party transactions. As a result, a tremendous amount of data gets collocated at third-party premises, used and shared by many applications that run beyond the user's premises or control.

In this project, we consider the management of data sharing processes in the cloud from the viewpoint of a single organization that consumes multiple cloud services. The base building block of our scenario is XACML reference model, which defines the architecture, language, and protocol for the authorization management in the enterprise environment. Being conceptualized earlier, and focused on a scenario consisting of a single organization, this model exhibits scalability and manageability deficiencies when it comes to the involvement of multiple organizations and assets. These deficiencies prevent the broader adoption of XACML beyond the larger enterprises due to the added overhead and required skills and infrastructure to support the complexity of the whole deployment.

This report presents our contribution toward mitigating barriers concerning scalability and manageability of heterogeneous XACML-based environments. Our contribution includes policy adaptor that allows the translation between harmonized RESTful service description models to the policies and attributes specific to proprietary environments. Being integrated with policy administration point of XACML, this adaptor allows the management of security policies at different clouds using a single and abstract service description model. This allows reducing the overhead of managing diverse services based on non-standardized approach. It also allows achieving the consolidated overview of security controls implemented at different services.

This work is experimental and focuses on RESTful services. Hence, it is not appropriate to be used with traditional web service stack or other types of XACML deployments.

2. XACML Architecture

XACML (eXtensible Access Control Markup Language) [1] is an OASIS standard that defines an architecture, policy language and interaction protocol for access control. The goal of XACML is to promote common technology and building blocks for interoperability of access control implementations by multiple vendors. In Figure 1 we show general building blocks that constitute XACML.

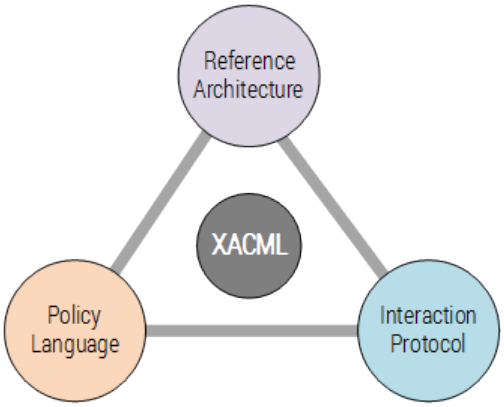

Figure 1: XACML Building Blocks

The core components of the XACML architecture and its accompanying data-flow model include Policy Enforcement Point (PEP) and Policy Decision Point (PDP) as well as Policy Administration Point (PAP), Policy Information Point (PIP) and Policy Retrieval Point (PRP). These are shown in Figure 2, which also shows a typical deployment of these components inside the premises of an arbitrary organization.

It should be noted that the components defined under XACML framework focus on providing the functionality related to the definition, evaluation, enforcement and management of security policies. The protocol itself is organization-centric, focusing only on interactions that are relevant to the enforcement infrastructure. The interactions that fall under the broader scope of authorization management, such as requesting authorizations, are not present. Instead, the only point that accessing clients interact with is PEP, which implements an opaque enforcement mechanism to control their accesses. Therefore, the clients are often not aware of the existence of an underlying infrastructure.

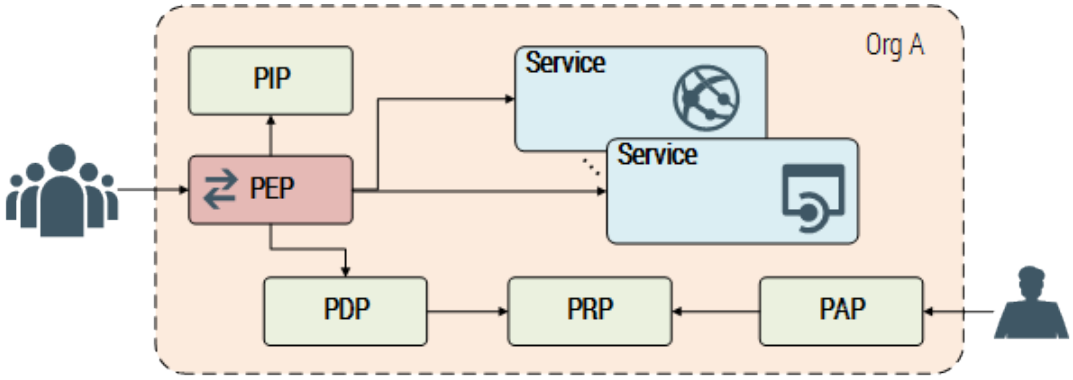

Figure 2: XACML Architecture

The security policy language established under XACML represents a more generalized, broadly adopted and standardized approach that specifies security policies in XM following the attribute-based access control model [2]. It defines the basic elements Rule, Policy and PolicySet. Rule entities implement the actual authorization logic and are the atomic units of the policy. The policy entity, in contrast, combines multiple rules for evaluation by the XACML engine. It furthermore has an associated algorithm for combining multiple rules and optionally a set of obligations or advices, which can be used to execute particular actions once a policy evaluation succeeds. A policy set is positioned one level above and incorporates multiple policies, other policy sets, obligations, and advices.

The XACML approach has evolved to the de-facto standard in the field of data-security policy languages and therefore is adapted, modified and improved extensively.

3. Authorization Management Models

We have identified three primary models covering XACML-based authorization management. We consider the existence of multiple organizations or organizational units in a single organization and the environment that includes several cloud providers. These providers act as independent entities, providing the services to the primary user (organization).

The abstract architecture of the first model is shown in Figure 3. This model considers the authorization management across multiple organizations. Here we can distinguish the cases of the management across different organizations, and the management across a single organization with multiple organizational units. In both cases, each of these entities consists of local, fully deployed XACML components.

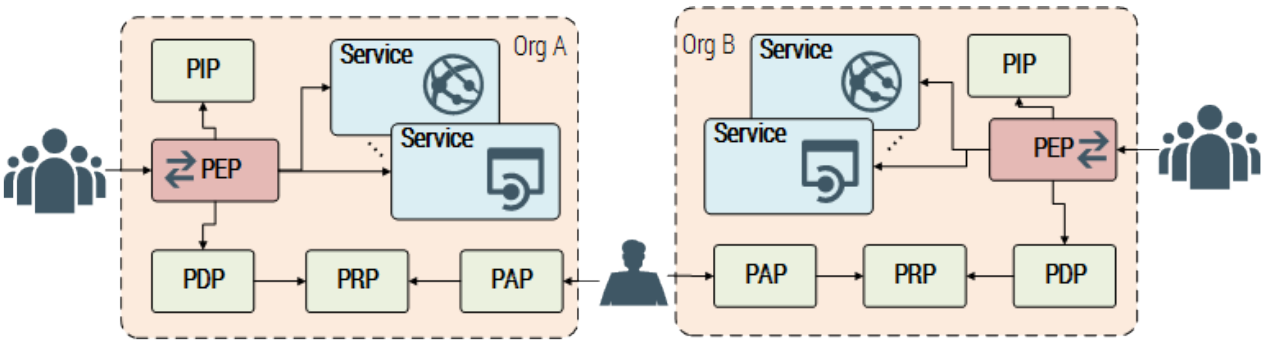

Figure 3: Managing authorizations in multi-organizational environment

In this setup, each entity deploys the services and a PEP that protects the access to these services. The policies are determined and stored locally, whereas the locally deployed PDP evaluates PEP requests based on these policies. Characteristic of this model is that, as each service and local environment are specific, they rely on a range of attributes and policies that are locally established. In a typical case, system administrators independently for each environment establish these assets. Such scenario is often applied in multi-organizational deployments, as each organization may have different practices or employs different companies. This scenario is also often found in complex organizations consisting of multiple units, where the units are geographically dispersed or exhibit a significant level of autonomy.

Considering back the architecture shown in Figure 3, we can observe that the policy management is performed by accessing the interface of each unit's specific PAP. Hence, to be able to administer the policies in both organizations in uniformly, the administrator has to derive the models of policies and attributes for each organization separately and then to harmonize them using some kind of abstract model or translators. This imposes an additional overhead, as the processes in organizations may differ and require additional manual effort to translate attribute and policy models from each organization.

In other words, this deployment does not allow the models developed in the domain of a single organization to be reusable in a different environment. Each context requires additional customizations that drive up the overall costs of integration.

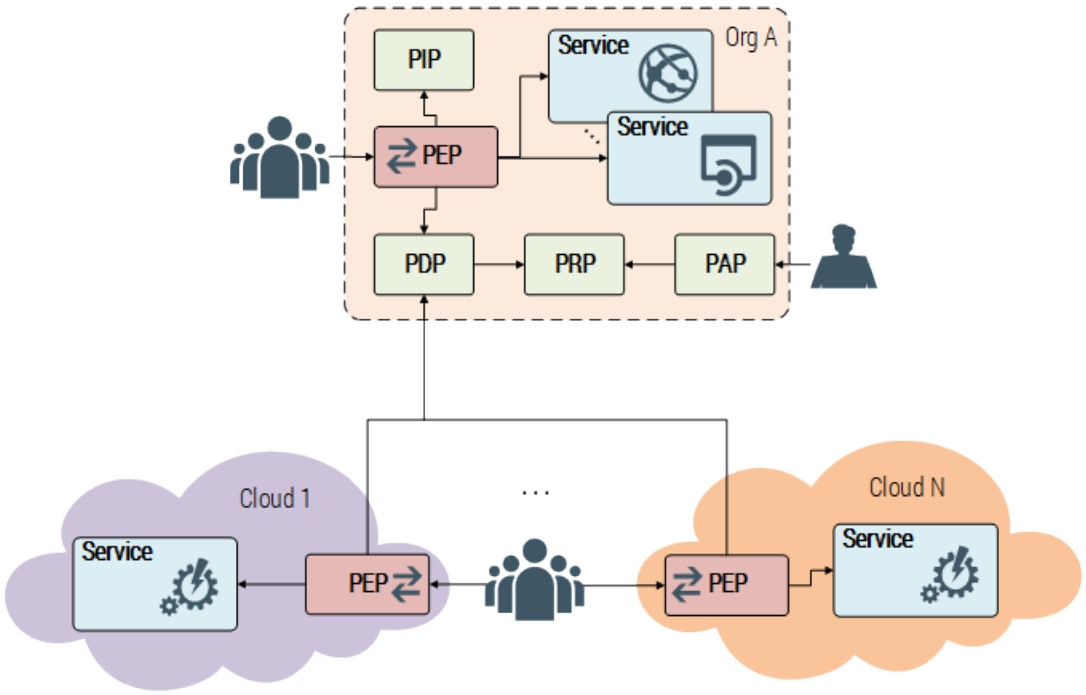

Figure 4: Centralized authorization management in multi-cloud environment

In Figure 4 we show an abstract model of authorization management across different organizations considering the use of external cloud services. In such context, clouds (Cloud 1..N) are hosted by different organizations, where each of them provides some services to the primary user entity (Org A on the figure). Org A is in this case considered as a tenant of services collocated at Cloud 1..N. While these services typically get shared across multiple tenants, for the purpose of simplicity and readability, on the figure we represent a single tenant. In this environment, each cloud hosts a PEP that contacts an authoritative PDP deployed inside the premises of Org A. This case hence combines both centralized and decentralized policy management, as the policy enforcement is based on a provider-centric infrastructure, and policy administration and evaluation rely on a user-centric (or decentralized) approach.

The advantage of this scenario is that the administration of policies can be centralized, using a unified set of attributes and policies for resources hosted at different premises. The similar architectural approach is proposed in UMA (User-managed access), an OAuth 2.0 [3] extension developed under the umbrella of Kantara Initiative. There are, however, several challenges that relate to this chase. First, cloud environments still determine the attributes that are relevant for the function of the services deployed in their domain. Although they partially rely on the same infrastructure, Cloud 1 and Cloud N on the figure typically do not share commonalities in attributes and hence policies. The centralized policy store and evaluation infrastructure still have to conform to and integrate diversified attributes and resource views imposed by the external infrastructures, which still involves additional overhead.

Second, the fact that PEP and PDP are deployed in different environments may cause a range of issues related to network connectivity and latency. These two components need to interact with each access coming from external clients. The failure or connection issues at the central PDP and its infrastructure automatically reflect all other cloud instances that host services for this organization.

The later issue can be mitigated by relying on decentralized architecture, as shown in Figure 5. In this scenario, PDP (with integrated policy store) is hosted in each cloud. The clouds also expose own PAPs, which are consumed by service users to administer security policies at each cloud. In this scenario, the user has to adapt and apply different attributes for each of environments. Like in the previous, this scenario also assumes that cloud organizations determine their attributes and policy models. The administrators hence need to adopt the models established at each entity, translate and coordinate the policies among different systems in order to apply integral and unified policy management.

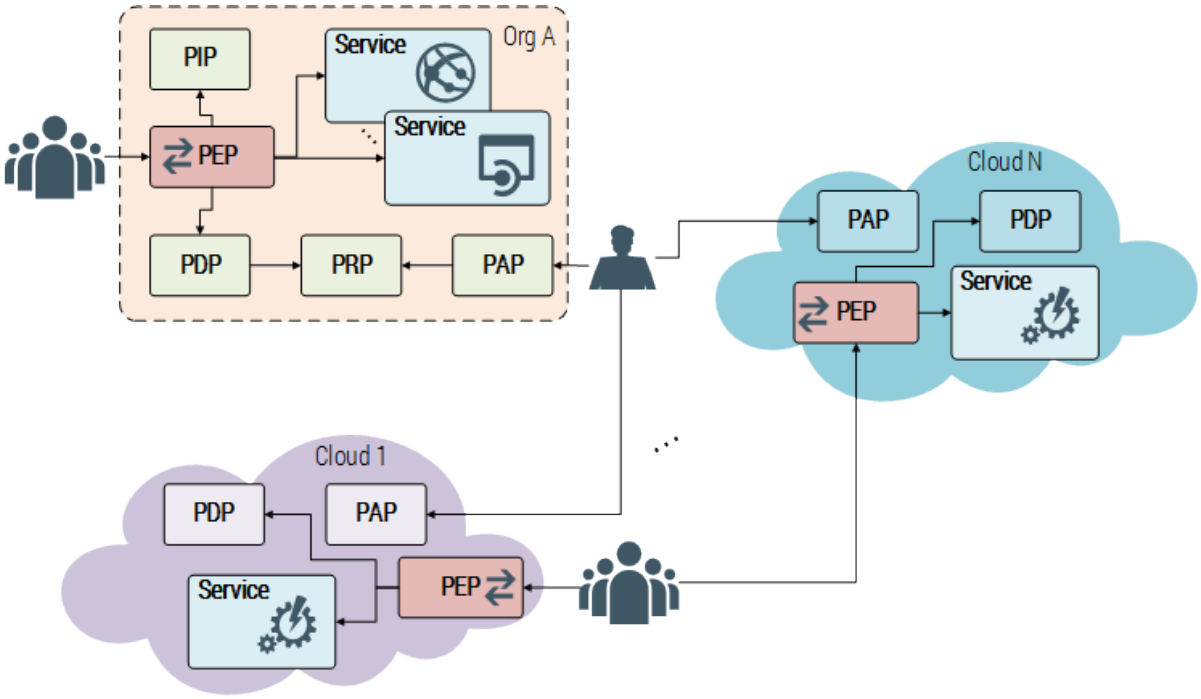

Figure 5: Decentralized authorization management in multi-cloud environment

The heterogeneity of environments is one of the leading factors that impose the overhead for the orchestration and management of security policies in both of following scenarios. While it may be seen as a neglecting issue of technical implementation, its practical effect is far more significant. This is due to the complexity of building blocks introduced with XACML, and primarily, the complexity and criticality of security policies and attributes that have to be designed and maintained during the product lifecycle. These properties impose the requirements, human skills and overhead on implementing users so that most larger organizations can be found among the adopters. Other companies that do not possess the necessary resources have choice to adopt the provider-specific access control and interfaces without the possibility to implement advanced security controls or to integrate the security management of cloud resources in its local, unified management environment.

4. XACML Adaptor for RESTful Services

Considering the fact that many modern services rely on a RESTful architecture, and that these services basically share common traits, we have designed the adaptor that enables the integrated management of security policies across different systems based on RESTful web services.

As elaborated in more detail in OpenAPI service description language [4], the primary building block of service descriptions are endpoints, which represent resources, and HTTP methods, which together with endpoints abstract different kinds of operations applicable over a specific resource. These include the operations such as adding, retrieving, updating or deleting a resource.

Instead of redefining attributes at each environment to allow describing of exposed web services, their resources and interactions with external entities, in our approach we use a specialized description model for RESTful services [5] and implement its translation to XACML policies that conform to a particular environment. This translation is supported by XAd policy adaptor, whose overall deployment is shown in Figure 6. For more details on the service description model, we refer to [6, 7].

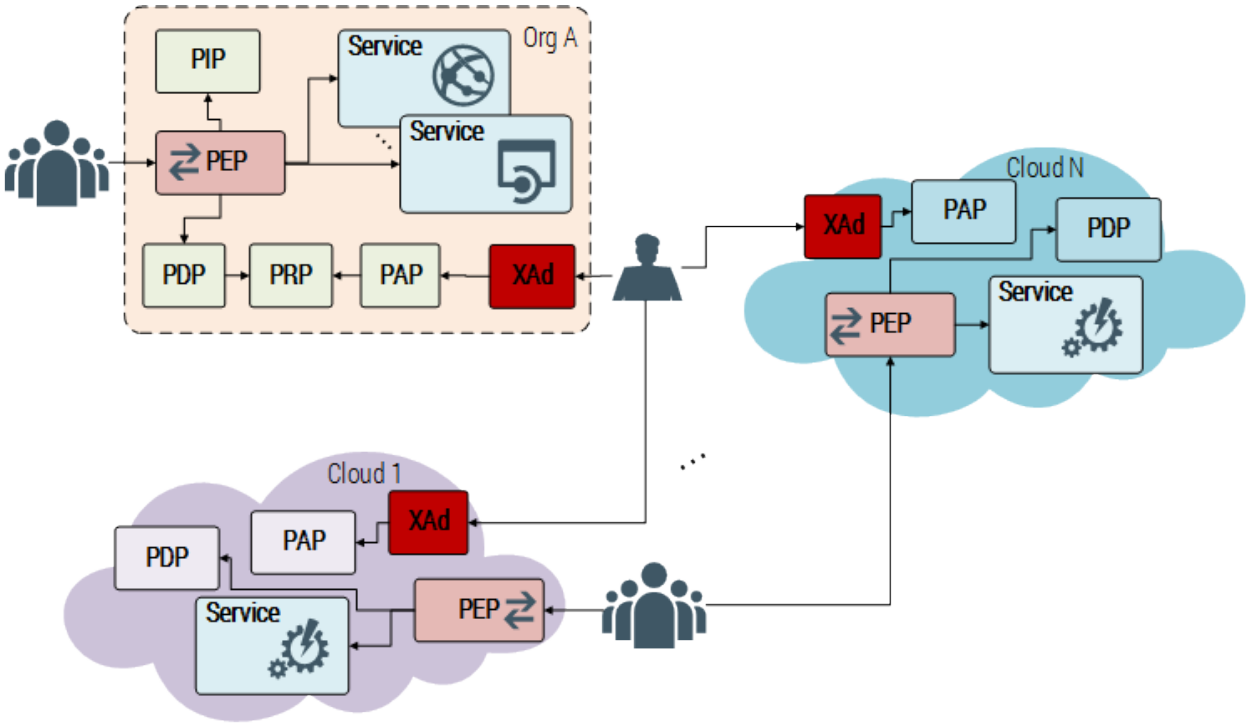

Figure 6: Deployment of the XAd adaptor

Hence, instead of interacting with each cloud service using its language and specific interface, XAd provides system administrators a uniform and consolidated interface that relies on common service

vocabularies. This interface allows gathering and updating the integrated picture of security allowances available at different services.

Acting as middleware, XAd exposes the RESTful interface to external clients with the dual functionality. Firstly, XAd exposes a model of an underlying service using the approach described in [7]. The service description can be determined manually, using the concepts defined under DASP-Service [6], or derived and adjusted from OpenAPI descriptions. Secondly, the same interface exposes endpoints for management of security policies using LEAR vocabulary [5].

On the other side, XAd connects with local PAP and accesses its policy repository. This is necessary in order to be able to translate from XACML to DASP-conform policies. The definition of attribute and concept mappings is done using a configuration file present in the conf folder of XAd. For further details on the configuration and deployment of XAd we refer to the package provided under its web site[1].

The advantage of the approach introduced with this model is that the user can use a single management model and infrastructure to control and orchestrate access control policies in several different cloud-based applications. The model assumes that different cloud providers deploy these applications, whereas each of them applies XACML for policy-based security management and enforcement. The need to implement and integrate solution specific for each cloud is mitigated by relying on a single, intermediary language based on DASP framework [6].

It should be noted that this approach at the time supports only the management of RESTful web services.

5. Summary

In this project, we extended the management model of XACML to support the transparent application in different, interacting and integrating environments. We have first identified and described most prominent authorization management models for XACML-based infrastructure considering the environment that consists of multiple organizations and relies on cloud-based services. Our proposal deals with RESTful services, as they are currently mostly proliferated form of web APIs, forming a backbone of data and resource sharing on today's web. While XACML provides the advanced and powerful model to implement attribute-based access control at the level of a single enterprise, the lack of standardized approach in the definition of attributes used for security policies impose the interoperability obstacles when it comes to the integrated management of services exposed at different systems or organizations. With our approach based on XAd middleware and abstract policies and service descriptions based on DASP framework, we support the transparent and integrated management of security policies that govern the access of different RESTful services using a unified approach.

[1] http://demo.a-sit.at/xad

References

[1] OASIS, "eXtensible Access Control Markup Language (XACML) Version 3.0 Plus Errata 01," OASIS, 2017.

[2] J. Xin, R. Krishnan and R. Sandhu, "A Unified Attribute-Based Access Control Model Covering DAC, MAC and RBAC," in *IFIP WG 11.3 Working Conference on Data and Applications Security and Privacy*, 2012.

[3] D. Hardt, "The OAuth 2.0 authorization framework," 2012.

[4] Linux Foundation, "OpenAPI Specification, Version 3.0.0.," 2017. [Online]. Available: https://github.com/OAI/OpenAPI-Specification/blob/master/versions/3.0.0.md.

[5] S. Bojan, "LOD and LOV for Authorization Concepts," 2017.

[6] "Data Sharing and Processing Framework," 2017. [Online]. Available: http://daspsec.org/.

[7] B. Suzic, "User-centered Security Management of API-based Data Integration Workflows," in *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, 2016.

[8] R. Fielding, R. Taylor, Erenkrantz, Gorlick and Whitehead, "Reflections on the REST architectural style and principled design of the modern web architecture," in *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*, 2017.