Semantic Policy Information Point – preliminary considerations

Michał Drozdowicz[1], Maria Ganzha[1,2], Marcin Paprzycki[1]

[1] Systems Research Institute Polish Academy of Science, Warsaw, Poland
(name.surname)@ibspan.waw.pl
[2] Department of Mathematics and Information Sciences,
Technical University of Warsaw, Warsaw, Poland

Abstract.

1 Introduction

With the rising prevalence of connected devices, including networks of sensors, there is a growing interest in providing solutions for capturing, storing and processing the vast amounts of collected data. Topics such as interoperability within the Internet of Things (IoT) also gain a lot of traction. A different issue that remains open, and at the same time growing in significance, is that of privacy and security of the data on all levels of this fast growing ecosystem.

As what concerns security of access to the data and operations exposed by the elements of the IoT, there are many similarities to the typical Web resources and services. There is an "entity," possibly described with several assigned attributes or roles, that requests access to a physical or a virtual resource(s) (or specific "services" available within such resources). In response, based on some declarative or imperative rules, such request is granted (or denied). However, in the case of IoT there are multiple reasons why the simple approaches, such as attribute or role based access control methods, may not scale well-enough and use of other solutions may be required.

The main aspects that make IoT unique when comparing to typical resources and services accessible in the Web are:

- Huge number of resources / producers ([3, 5])
- Fast growing number of consumers ([6, 11, 18])
- Enermous heterogeneity of data and service formats and descriptions ([1, 2, 8, 12])
- Unprecedented dynamics of (often short-lived) interactions between constantly changing parties ([14, 15, 17])
- Machine-machine interactions – especially on the "lower level," where it is typically one device consuming the data produced by another device, while the role of the "human" is marginalized ([4, 5])

The aim of this position paper is to briefly summarize common approaches of dealing with the aforementioned challenges and to introduce a semantically enriched access control policy system.

2 Policy based access control

In many cases, access control is embedded into the logic of the service or the resource provider, and intertwined with the business logic. However, in an environment consisting of a very large number of different services, such approach leads to an unmaintainable, inconsistent set of rules that also lack visibility. (e.g. [5, 14, 17])

A better approach would be to move access control decisions out of the services and into a centralized authorization component or a set of such components. One way to design such a subsystem would be to use an "engine" that uses declarative policies, specifying the conditions under which a given request is accepted, or when access is denied.

3 XACML

One of the most common policy specification languages in use today is the eXtensible Access Control Markup Language (XACML; [7]). It is a declarative language and a standard for implementation of processing engines developed and maintained by the OASIS group. The standard uses the XML as its serialization format, but many implementations handle information transfer in other formats, such as, for instance, the SAML.

At its core, XACML enables fine grained access control based on attribute values. The decisions are made based on policies consisting of rules. In the case when multiple policies are applicable to the same request, a policy combining algorithm, defined in the, so called, policy set that encompasses all existing / defined policies, is used to produce the final result.

The way that the XACML engines make the decisions on incoming requests, is based on two-step attribute evaluation. First, the conditions defined in the *Target Element* of the policy or rule are checked to limit the amount of rules to process. Second, the *Condition* is evaluated and, based on the result, the rule or policy decision is made.

In the XACML the attributes are grouped into four categories:

- *Subject* – the person or entity requesting access.
- *Resource* – the entity, access to which is under control.
- *Action* – the action that the Subject requests on the Resource.
- *Environment* – other attributes that bring additional context.

The reference architecture of an XACML processing system contains the following major components:

- *Policy Enforcement Point* (PEP) – responsible for the actual act of enabling or preventing access to the resource.
- *Context Handler* – which converts requests and responses between native formats and the XACML canonical representation and coordinates with the PIPs the gathering of required attribute values.

– *Policy Information Point* (PIP) – a source of attribute values.
– *Policy Decision Point* (PDP) – which evaluates policies and issues authorization decisions.
– *Policy Administration Point* (PAP) – which defines, stores and manages policies.

Diagram 1 depicts the sequence of messages in a typical access control decision that takes place in the considered architecture. Handling obligations has been omitted for brevity.

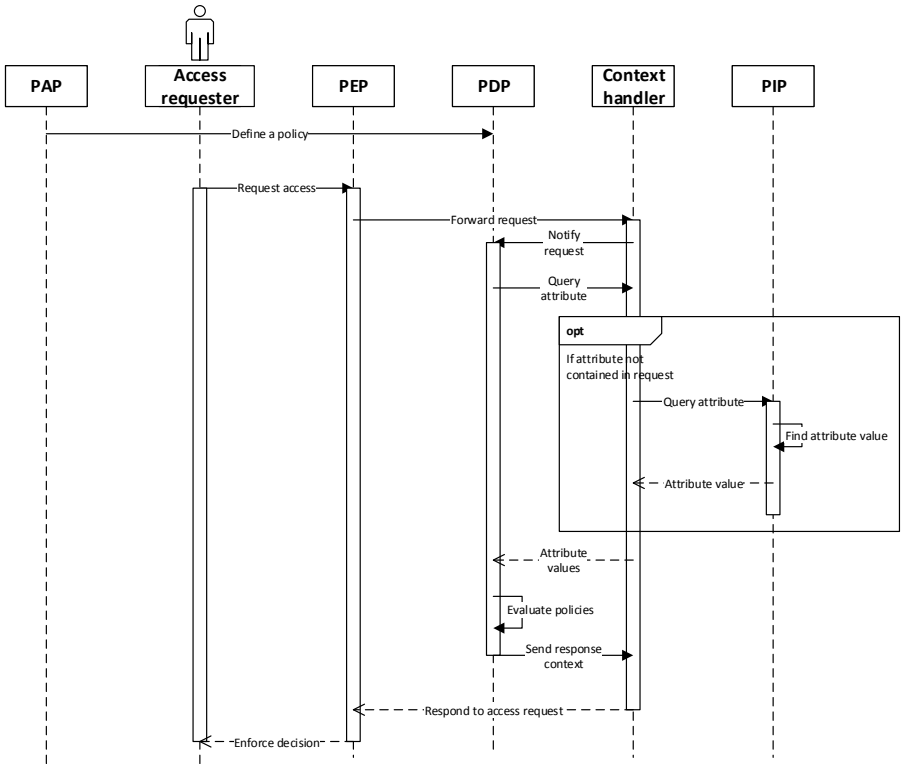

Fig. 1. Sequence diagram in an access control decision????

While the XACML has many advantages, such as expressiveness, extensibility it also has some serious drawbacks. More complex policies that involve relations between the attributes or calculation of values based on other attributes, can be extremely convoluted and hard to maintain. In this respect, what XACML lacks most is the possibility to reason about the domain containing the attributes and infer additional data in an automatic way.

4 Semantic approaches to access control policies

The functionality that the XACML is missing can be naturally handled using techniques from the area of semantic data processing. Indeed, there have been projects tackling the challenge of access control by defining policies using special ontologies and basing the decision engines on semantic reasoners.

The Rei project [20] was created to address security concerns in Semantic Web, mobile and pervasive computing scenarios. Its engine utilized policies defined in OWL-Lite and provided ability to reason about permissions and obligations, policy conflict resolution, permission delegation and possibility to define variables that could be referenced in the policy rules. The project was later modified as Rein to use the N3 language as its base and provide extensible, reusable meta-policies for federation networks.

KAoS framework [19] is similar to Rei in its language choices, using OWL-DL for policy specification. It differs in its primary application area - being multi-agent systems and distributed systems. As a matter of fact, the policy authorization engine was but a part of a more general agent platform, including also directory services and simplified GUI tools for policy creation and management.

In contrast to the previous two projects, Ponder ([16]), which actually predated them and laid grounds for many of their features, does not any ontological language for its policies. Instead, it utilizes a custom declarative language that is later processed by the Java based engine. It provides reasoning capabilities, however, not as powerful as OWL based reasoners used in Rei and KAoS.

Unfortunately, these projects never gained enough traction and did not achieve significant adoption. In this respect the plain XACML was much more successful.

5 Semantic extensions to the XACML

As a consequence of its popularity, XACML has also become the base for many research projects dealing with various aspects of access control. Some were centered around the challenges of spatio-temporal constraints applied to policies , others were dealing with deficiencies of the language in some more sophisticated elements of Role Based Access Control.

There were also some more general extensions of the language aimed at taking advantage of the semantic reasoning capabilities of the description logic languages.

6 Semantic Policy Information Point

Due to the aforementioned strengths of the approach proposed in [10], we have decided to follow on the general (top-level) ideas outlined in that paper.

The result is an implementation of the Policy Information Point of the XACML reference architecture that is capable of providing values to unknown attributes by inferring them from the ontologies describing the domain of the system.

The general algorithm used by the component is described with the following steps:

1. When the Context Handler queries for an attribute value, we translate the request context into a temporary ontology.
2. The temporary ontology is merged with the existing ontologies specifying the details of the domain.
3. Pellet [9] reasoner is used to reason about all ontology properties that are not explicitly specified.
4. A SPARQL query is issued on the ontology to retrieve the attribute value and type.
5. The response from SPARQL is converted into the format acceptable by the Context Handler.

In comparison to the solution developed by Priebe et al. [21] our solution does not change the reference architecture of XACML system, it merely implements the contract of the PIP component. This is also reflected in the fact that we only query for and return the attribute values that are requested by the Context Handler, reducing the burden on the SPARQL engine.

The component is built as an attribute finder extension to the Balana [13] XACML engine developed by WSO2 as a continuation of the popular Sun's XACML Implementation. The engine was chosen due to its maturity and widespread use as part of the WSO2 Identity Server product package.

7 Conclusions and future work

References

1. Bringing big data to the enterprise. http://www-01.ibm.com/software/data/bigdata/, 2015.
2. Collaborative open market to place objects at your service. http://www.compose-project.eu/, 2015.
3. CSA for Global RFID-related Activities and Standardisation (CASAGRAS2). http://www.iot-casagras.org/, 2015.
4. iCore: Empovering IoT Through Cognitive Technologies. http://www.iot-icore.eu/, 2015.
5. Internet-of-Things Architecture. http://www.iot-a.eu/public, 2015.
6. Iot@work. https://www.iot-at-work.eu/, 2015.
7. OASIS Standard. http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html, 2015.
8. Open source cloud solution for the internet of things. http://www.openiot.eu/, 2015.
9. Pellet. https://github.com/Complexible/pellet, 2015.
10. Trzeba znalezc. https://community.emc.com/docs/DOC-7410, 2015.
11. uBiquitous, secUre inTernet-of-things with Location and contEx-awaReness (BUTLER: Smart life. http://www.iot-butler.eu/, 2015.
12. "web of objects" ITEA 2 Project. www.web-of-objects.com/, 2015.

13. Xacml for autorisation. http://xacmlinfo.org/category/xacml/, 2015.
14. Erin Anzelmo, Alex Bassi, Dan Caprio, Sean Dodson, Rob van Kranenburg, and Matt Ratto. Internet of things. discussion paper. Institute for Internet and Society, Berlin, October 2011.
15. L. Atzori, A. Iera, and G. Morabito. The internet of things: A survey. In *Computer Networks: The International Journal of Computer and Telecommunications Networking*, volume 54, page 2787âĂŞ2805. Elsevier North-Holland, Inc., 2010.
16. N. Damianou, N. Dulay, E. C. Lupu, and M. Sloman. Ponder: A language for specifying security and management policies for distributed systems. Technical report, Imperial College, UK, Department of Computing, 2001.
17. D. Evans. The internet of things: How the next evolution of the internet is changing everything. Technical report, Cisco Internet Business Solutions Group (IBSG), 2011.
18. FASyS. Absolutely safe and healthy factory. http://www.fasys.es/en/index.php, 2015.
19. M. Johnson, P. Chang, R. Jeffers, J. Bradsha, V-W. Soo, M. Breedy, L. Bunch, S. Kulkarni, J. Lott, N. Suri, and A. Uszok. KAoS semantic policy and domain services: An application of DAML to web-services-based grid architectures. In *Proceedings of the AAMAS 03: Workshop on Web Services and Agent-Based Engineering*, 2003.
20. L. Kagal, T. Finin, and A. Johshi. A policy language for pervasive computing environment. In *Proceedings of IEEE Fourth International Workshop on Policy (Policy 2003)*, pages 63–76. Los Alamitos, CA: IEEE Computer Society, 2003.
21. T. Priebe, W. Dobmeier, and N. Kamprath. Supporting attribute-based access control with ontologies. In *Availability, Reliability and Security, 2006. ARES 2006. The First International Conference on*, pages 8 pp.–, April 2006.