A new approach for WS-Policy Intersection using Partial Ordered Sets

Abeer Elsafie, Christian Mainka, and Jörg Schwenk

Horst Görtz Institute for IT-Security, Ruhr-University Bochum, Germany
{abeer.elsafie, christian.mainka, joerg.schwenk}@rub.de

Abstract. WS-Policy is a framework that can be used to describe assertions for web services message exchange. In the context of Service Oriented Architectures and Clouds, where web services are belonging to, machine-to-machine communication is one of its core ideas. When those machines try to apply WS-Policy, mainly two events can occur: First, the machine-exchanged policies have common assertions – there is an intersection. Second, there is no direct intersection and the participants must reach an agreement by minimal adjustments to the policies. This paper introduces a new approach for reaching intersection by computing adjustments to the policies using partial ordering.

Keywords: WS-Policy Intersection, Partial Ordered Sets, Hasse Diagram

1 Introduction

In the field of web services, requirements and capabilities can be described using XML according to the WS-Policy specification [1]. The policies can be applied to the web services message exchange, which is commonly machine-to-machine communication with multiple participants, for assuring security goals. This leads to the need for WS-Policy intersection, a technique used when two or more web services want to communicate and fulfill each others policy. Currently, this approach can only handle the case that intersection within the participating policies exists [2]. Otherwise it fails and the further communication cannot be achieved.

Hence, our motivation is to find a way to make intersection possible even in the case that there is no direct intersection by adjusting one or both party's policy, e.g. by adding some policy aspects. This is achieved by a multi-layer approach: First, every WS-Policy, which can be seen as a set of Boolean terms, is converted into its *disjunctive normal form* ($\mathcal{DNF}$), so that policies are easy to compare and finding matching terms is simple. In the case that there is a match, the decision for the participants is obviously done. If there is no direct intersection, this paper introduces a model for an arbitrary number of parties, that computes these adjustments using partial order sets to enforce policy intersection for all participants.

2 Foundations

2.1 WS-Policy and Policy Intersection

WS-Policy is a framework for describing policies using XML [3, 1]. In the context of web services, it is commonly used to specify which parts of a message should be signed or encrypted using WS-SecurityPolicy [4]. The structure of a WS-Policy can be seen as a Boolean term, but written in XML. It consists of an enveloping `<Policy/>` element which can contain arbitrary *AND* (element: `<All/>`) and *XOR* (element: `<ExactlyOne/>`) expressions. For each term, there exists a *disjunktive normal form* ($\mathcal{DNF}$). It is an XOR-junction of propositions derived from the compact form using boolean algebra [5]. Consider the following example, which does not use any XML for simplicity:

$$\mathcal{A}_1 \wedge (\mathcal{A}_2 \oplus \mathcal{A}_3) \overset{\mathcal{DNF}}{=} \underbrace{\mathcal{A}_1 \wedge \mathcal{A}_2}_{\text{Alternative 1}} \oplus \underbrace{\mathcal{A}_1 \wedge \mathcal{A}_3}_{\text{Alternative 1}}$$

From the $\mathcal{DNF}$, one can easily see the *policy alternative*: They are a bundle of assertions which must be fulfilled.

The *WS-Policy Intersection* process identifies compatible policy alternatives included in all parties policies or returns nothing if there are no matches [6]. Two alternatives are compatible, if the sets of included assertions are identical.

2.2 Ordered Sets and Hasse Diagrams

A *partially ordered set* (poset) is a mathematic tool generalizing the concept of arranging and ordering elements. In a poset, there exists a relation between pairs of elements, e.g. the "$\leq$"-relation, so that the elements can be compared. When this relation exists for each possible pair, then the poset is called a chain (or total ordered set). In addition a poset in which no two distinct elements are comparable is called *antichain*.

A *Lattice* is an ordered set where every pair of elements has a *least upper bound* (LUB) and a *greatest lower bound* (GLB). In our approach we assume that the posets are all Lattices.

A *Hasse or Lattice diagram* is a visualization of the finite poset in the form of a drawing, in which nodes are elements of the poset and arrows between related nodes represent the order relation between these elements [7, 8]. In the next section we introduce an example providing a detailed overview of the usage of Hasse diagram.

3 WS-Policy Intersection Model

The evaluation of WS-Policy Intersection consists of two main layers as shown in Figure 1:

The *preparation* layer is responsible for converting each policy into its corresponding $\mathcal{DNF}$. This is achieved either manually or using an software-tool [9]

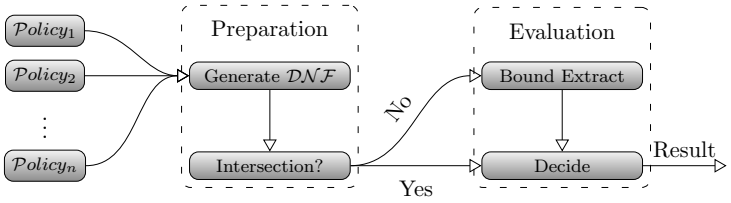

Fig. 1. Evaluating WS-Policy Intersection Model.

and is outside the scope of this research. Afterwards, the policy intersection examination unit compares the $\mathcal{DNF}$ policies and forwards the results to the *evaluation* layer.

If there is intersection, which means compatible alternatives exist, they are directly forwarded to the *decision making* unit, which chooses the strongest alternative.

In the case of no intersection, the *bound extraction* unit takes part. It first identifies all ordered sets, which can be chains like $AES_{128} < AES_{256}$ or anti-chains which cannot be compared, e.g. $Sign_{\mathcal{H}\text{eader}}$ and $Sign_{\mathcal{B}\text{ody}}$. Afterwards, all sets are combined to one Hasse diagram as shown in Figure 2.

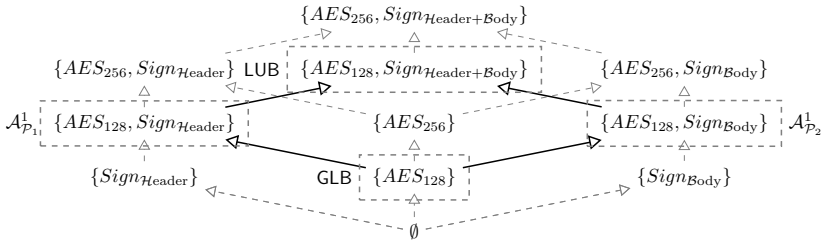

Fig. 2. $\mathcal{S}$igned $\mathcal{P}$art and cryptographic suite combined into one Hasse diagram.

Consider the two policies $\mathcal{P}_1$ and $\mathcal{P}_2$, having the alternatives $\mathcal{A}^1_{\mathcal{P}_1}$ and $\mathcal{A}^1_{\mathcal{P}_2}$ as shown. Obviously, they are not compatible. Using the Hasse diagram, the *least upper bound* (LUB) and the *greatest lower bound* (GLB) can be easily extracted. In general if we consider that the posets used are all lattices, where each two elements have a LUB/GLB, then we can easily use the meet and join for finding these bounds [8]. Finally the bounds are forwarded to the *decision* unit, which has to decide if the GLB or either the LUB should be used.

4 Related Work

Researchers in [10] investigated a mechanism for calculating compatibility of alternatives. An approach for comparing policies and checking compatibility between alternatives in terms of its assertions to reach intersection is shown in [11]

and [12]. Policy reconciliation algorithm, a technique to reach policy agreement between two party communication, is introduced in [13]. Another research using a web ontology language (OWL-DL) is based on the idea that policy assertions and alternatives are mapped in to program classes using OWL to measure compatibility [6]. Our research focuses on how to examine intersection and find solution for policy agreement by means of partial ordering.

5 Conclusions and Future Work

This paper presents a model for WS-Policy Intersection using Partial ordered sets. It is the first solution which is able to (1) handle more than two parties and (2) makes proposals for the case that the policies are not directly compatible.

For future work we plan to investigate a real protocol for multi-party negotiation and additional, an implementation which will show the practical usability.

References

1. W3C Recommendation, "Web Service Policy 1.5 - Framework," http://www.w3.org/TR/ws-policy/, Sep. 2007.
2. ——, "Web Service Policy Intersection," http://www.w3.org/TR/ws-policy/, Sep. 2007.
3. ——, "Web Service Policy 1.5 - Primer," http://www.w3.org/TR/ws-policy-primer/, Nov. 2007.
4. OASIS Standard, "Web Service Security Policy," http://docs.oasis-open.org/ws-sx/ws-securitypolicy/, Feb. 2009.
5. J. Eldon Whitesitt, *Boolean algebra and its applications*. Courier Dover, 1995.
6. V. Kolovski, B. Parsia, Y. Katz, and J. Hendler, "Representing Web Service Policies in OWL-DL," in *In International Semantic Web Conference (ISWC)*. Springer, Nov. 2005, pp. 461 – 475.
7. W. Strunk, Jr. and E. B. White, *Order Relation*, 3rd ed. Macmillan, 1979.
8. M.-C. van Leunen, *Partial order*. Knopf, 1979.
9. T. A. S. F. Group, "The Apache Software Foundation," https://ws.apache.org/neethi/, Jul. 2012.
10. T. Lavarack and M. Coetzee, "Considering Web Services Security Policy Compatibility," in *The 9th Annual Information Security for South Africa Conference, (ISSA 2010)*. IEEE Press, Aug. 2010, pp. 1 – 8.
11. B. Hollunder, "Domain-Specific Processing of Policies or: WS-Policy Intersection Revisited," in *IEEE 7th International Conference on Web Service (ICWS2009)*. IEEE Press, Jul. 2009, pp. 246 – 253.
12. S. Hudert, T. Eymann, H. Ludwig, and G. Wirtz, "A Negotiation Protocol Description Language for Automated Service Level Agreement Negotiations ," in *Commerce and Enterprise Computing, 2009. CEC '09.* . IEEE Press, Aug 2009, pp. 162 – 169.
13. A. P. P McDaniel, "Methods and limitations of Security Policy Reconciliation," in *2002 IEEE Symposium on Security and Privacy*. IEEE Press, May 2002, pp. 73 – 87.