Dynamic, Secure Resource Control in the Cloud

Edward J. Nava, Viswanath Nandina, José Marcio Luna, Christopher C. Lamb,
Gregory L. Heileman, Chaouki T. Abdallah
University of New Mexico
Department of Electrical and Computer Engineering
Albuquerque, NM 87131-0001
{ejnava, vishu, jmarcio, cclamb, heileman, chaouki}@ece.unm.edu

ABSTRACT

In this paper we describe the development of a domain specific language (DSL) for expressing usage management policies and associating those policies with managed artifacts. We begin by framing a model for the language, including generalized use cases, a domain model, a general supported life-cycle, and specific extension requirements. We then develop the language from that model, demonstrating key syntactic elements and highlighting the technology behind the language while tracing features back to the initial model. We then demonstrate how the DSL supports common usage management and DRM-centric environments, including creative commons, the extensible rights markup language (XrML), and the open digital rights language (ODRL).

Categories and Subject Descriptors

D.3.0 [**Software**]: Programming Languages—*General*

General Terms

Design, Languages, Security

Keywords

Access Control, Interoperability, DRM, Usage Management

1. INTRODUCTION

In this paper, we define usage management as the ability to control actions over resources and data across and within computing environments. More than access control or digital rights management, usage management addresses with fine-grained control of all aspects of how a given digital resource is used. As digital environments become more open over time, the need for usage management for resources that span utility computational environments (e.g. cloud provider systems) will become increasingly important [17, 18].

With the advent and widespread use of cloud computing, those responsible for a given usage managed resource are almost never those responsible for the computing systems, except at edge devices like mobile phones or other small profile computing devices. Resources are regularly moved across national boundaries and regional areas without either the content owner's or creator's knowledge. Furthermore, this kind of transfer is generally according to pre-established algorithms or data routing protocols over which users have no control. Managing these issues requires new usage management capabilities that can run on platforms ranging from small, hand-held devices to nodes in large data centers.

Research in this area has been focused on developing more expressive policy languages via either different types of mathematical logics or formalisms with greater reasoning capabilities [4, 5, 7, 9, 10, 20, 24]. These approaches however fail to address interoperability challenges posed by new commercially available distributed computing environments. Interoperability efforts have resorted to translation mechanisms, where the policy is translated in its entirety to a different language [11, 19, 22]; it has been shown recently however that such techniques are infeasible and hard to perform for most policy languages [16, 21]. Other approaches have led to complex policy specification languages that have tried to establish themselves as the universal standard [1, 2, 23, 25]. This unfortunately tends to stifle both innovation and flexibility [11, 12, 13, 15].

To address these issues, we first applied the principles of system design to develop a framework for usage management in open, distributed environments that supports interoperability. These principles have been used by researchers in large network design to create a balance between interoperability and open, flexible architectures [3, 6, 8], without sacrificing innovation. Initially we standardized certain features of the framework operational semantics, and left free of standards features that necessitate choice and innovation.

We have implemented this framework, including a usage management calculus providing a platform for usage management, within a Domain Specific Language (DSL) and associated evaluation environment. The DSL and its environment implements the previously defined framework, separating various roles needed for distributed policy creation and management, provides the capability to develop executable licenses, and is extensible from both a policy and constraint definition perspective.

In this paper, we will first review the problems in usage

CCSW' 12, October 19, 2012, Raleigh, NC, USA.

management in more detail, providing the context for this DSL and the problems it helps solve. Then, in Section **??** we will first review the model we developed to guide the DSL's syntactic and semantic development. In the next section, we will cover the language itself, how it was developed, and its supporting evaluation environment. We will then close the paper with three specific implementation examples showing how the language and its runtime support usage management scenarios from three different environments — creative commons (CC), the extensible rights markup language (XRML), and the open digital rights language (ODRL).

2. MOTIVATION

Usage management incorporates specific characteristics of traditional access control and digital rights management incorporating encryption mechanisms, trust management, and trusted computing platforms [14]. In order to be effective, it must be flexible enough to provide users with opportunities for differentiation and extension, but interoperable enough to provide services across widely diverging computational environments.

This DSL and runtime provide flexibility through the inclusion of easily pluggable constraint and policy evaluators. Both types of evaluators simply need to be locatable by the runtime, via standard classpath-style semantics. If they are, they can be loaded and used with no other configuration required. A policy file can simply reference the evaluator via a predefined language element, and the runtime will find it and load it automatically, using it to evaluate either the policy itself or the defined constraints (depending on whichever type of evaluator is specified).

The policy runtime elements can be dynamically loaded on host systems when required. This provides interoperability by forcing the policy system itself to accept the responsibility of ensuring it is installed wherever needed. The technology upon which the runtime is based is widely available as well. Additionally, the policies can be transferred from one host to another and can be directly executed upon delivery. Furthermore, this language and system can express the semantics of common rights management environments, providing additional semantic flexibility and interoperability.

The DSL we describe, in tandem with its runtime, embodies an interoperable framework for usage management, unlike Coral and Marlin architectures, that implements a formal calculus to reason about the relationship between a license, a computing environment, and interoperability between them. It incorporates concepts such as programmable licenses and common ecosystems used by Coral and Marlin architectures respectively. The DSL design is based on the principles of *design for choice*, eloquently described by Clarke et al. with reference to "tussles" in cyberspace [8]. They explain the importance of identifying the locations in the architecture where standards need to be introduced to enable interoperability, and locations where they should *not* be applied to enable innovation and differentiation. By supporting pluggable evaluators that allow users to extend the basic language syntax, semantics, and runtime arbitrarily, this language system provides space for innovation.

3. CONCLUSIONS AND FUTURE WORKS

Usage management is a common problem set with features embodied in domains ranging from security systems to video games to music production and retail. The ability to provide management of resources with regard to authorized subjects is being addressed in multiple different forums, many of which are taking remarkably different approaches. Common features however generally include the need for either ubiquitous rights expression language acceptance or for extensive translation between all supported rights languages.

In this paper, we first demonstrated the development of the initial model used to define the problem space. Here, we described the general use of the system, who the primary users were, what the expected life-cycle of policies was, and what the domain model looked like. We then implemented the syntax of the DSL, in Ruby, as an internal DSL with specific examples. We concluded the paper with demonstrations of equivalence to common rights management frameworks like the creative commons, ODRL, and XrML.

We have only begun to specify and use this particular DSL. Future focus on this effort will include additional language elaboration, exploration, and use in specific scenarios. We need to spend additional time engineering the underlying software as well, so we can ensure that policies are in fact platform and environment agnostic, portable, and executable. Finally, this implementation is an internal DSL within the Ruby language; we need to explore the application of external DSL techniques to this domain to better understand the required compromises between expressiveness and development difficulty and begin to apply more stringent security models to the system itself.

4. REFERENCES

[1] Enabler release definition for DRM V2.0. Technical report, Open Mobile Alliance, 2003. `xml.coverpages.org/OMA-ERELD_DRM-V2_0_0-20040401-D.pdf`.

[2] Open digital rights language ODRL version 2 requirements. ODRL, Feb. 2005. `odrl.net/2.0/v2req.html`.

[3] H. Alverstrand. The role of the standards process in shaping the internet. *Proceeding of the IEEE*, 92(9):1371–1374, 2004.

[4] A. Arnab and A. Hutchison. Persistent access control: A formal model for drm. In *DRM '07: Proceedings of the 2007 ACM workshop on Digital Rights Management*, pages 41–53, New York, NY, USA, 2007. ACM.

[5] A. Barth and J. C. Mitchell. Managing digital rights using linear logic. In *LICS '06: Proceedings of the 21st Annual IEEE Symposium on Logic in Computer Science*, pages 127–136, Washington, DC, USA, 2006. IEEE Computer Society.

[6] M. S. Blumenthal and D. D. Clark. Rethinking the design of the Internet: The end-to-end arguments vs. the brave new world. *ACM Transactions on Internet Technology*, 1(1):70–109, Aug. 2001.

[7] C. N. Chong, R. Corin, S. Etalle, P. Hartel, W. Jonker, and Y. W. Law. LicenseScript: A novel digital rights language and its semantics. In *Third International Conference on the Web Delivery of Music*, pages 122–129, Los Alamitos, CA, Sept. 2003.

[8] D. D. Clark, J. Wroclawski, K. R. Sollins, and R. Braden. Tussle in cyberspace: Defining tomorrow's internet. In *SIGCOMM*, pages 347–356, Pittsburg, Pennsylvania, USA, Aug. 2002.

[9] J. Y. Halpern and V. Weissman. A formal foundation for XrML licenses. In *Proceedings of the 17th IEEE Computer Security Foundations Workshop*, pages 251–265, Asilomar, CA, June 2004.

[10] J. Y. Halpern and V. Weissman. A formal foundation for XrML. *J. ACM*, 55(1):1–42, 2008.

[11] G. L. Heileman and P. A. Jamkhedkar. DRM interoperability analysis from the perspective of a layered framework. In *Proceedings of the Fifth ACM Workshop on Digital Rights Management*, pages 17–26, Alexandria, VA, Nov. 2005.

[12] P. A. Jamkhedkar and G. L. Heileman. DRM as a layered system. In *Proceedings of the Fourth ACM Workshop on Digital Rights Management*, pages 11–21, Washington, DC, Oct. 2004.

[13] P. A. Jamkhedkar and G. L. Heileman. *Handbook of Research on Secure Multimedia Distribution*, chapter Rights Expression Languages. IGI Publishing, 2008.

[14] P. A. Jamkhedkar, G. L. Heileman, and C. C. Lamb. An interoperable usage management framework. In *Proceedings of the tenth annual ACM workshop on Digital rights management*, DRM '10, pages 73–88, New York, NY, USA, 2010. ACM.

[15] P. A. Jamkhedkar, G. L. Heileman, and I. Martinez-Ortiz. The problem with rights expression languages. In *Proceedings of the Sixth ACM Workshop on Digital Rights Management*, pages 59–67, Alexandria, VA, Nov. 2006.

[16] R. H. Koenen, J. Lacy, M. MacKay, and S. Mitchell. The long march to interoperable digital rights management. *Proceedings of the IEEE*, 92(6):883–897, 2004.

[17] C. C. Lamb, P. A. Jamkhedkar, G. L. Heileman, and C. T. Abdallah. Managed control of composite cloud systems. In *6th IEEE International Conference on System of Systems Engineering (SOSE)*. IEEE.

[18] C. C. Lamb, P. A. Jamkhedkar, G. L. Heileman, and C. T. Abdallah. Managed control of composite cloud systems. In *System of Systems Engineering (SoSE), 2011 6th International Conference on*, pages 167 –172, june 2011.

[19] J. Polo, J. Prados, and J. Delgado. Interoperability between ODRL and MPEG-21 REL. In *Proceedings of the first international ODRL workshop*, Vienna, Austria, Apr. 2004.

[20] R. Pucella and V. Weissman. A logic for reasoning about digital rights. In *Proceedings of the 15th IEEE Computer Security Foundations Workshop*, pages 282–294, Nova Scotia, Canada, June 2002.

[21] R. Safavi-Naini, N. P. Sheppard, and T. Uehara. Import/export in digital rights management. In *Proceedings of the Fourth ACM Workshop on Digital Rights Management*, pages 99–110, Washington, DC, Oct. 2004.

[22] A. U. Schmidt, O. Tafreschi, and R. Wolf. Interoperability challenges for DRM systems. In *IFIP/GI Workshop on Virtual Goods*, Ilmenau, Germany, 2004. `http://virtualgoods.tu-ilmenau.de/2004/program.html`.

[23] X. Wang. MPEG-21 rights expression language: Enabling interoperable digital rights management. *IEEE Multimedia*, 11(4):84–87, Oct./Dec. 2004.

[24] J. Xiang, D. Bjorner, and K. Futatsugi. Formal digital license language with OTS/CafeOBJ method. In *Proceedings of the sixth ACS/IEEE International Conference on Computer Systems and Applications*, Doha, Qatar, Apr. 2008.

[25] eXtensible Rights Markup Language (XrML) 2.0 Specification, November 2001. `www.xrml.org`.