A Language Driven Approach to Multi-System Access Control

Steven Davy, Jason Barron, Lei Shi, Bernard Butler and Brendan Jennings
Telecommunications Software & Systems Group,
Waterford Institute of Technology, Waterford, Ireland
{sdavy, jbarron, lshi, bbutler}@tssg.org, bjennings@ieee.org

Keith Griffin, Kevin Collins
Cisco,
Galway, Ireland
{kegriffi, kevinco}@cisco.com

***Abstract*—Resource access control policies for an organization are often derived from best practice standards or from high level business policies. To ensure that access control is enforced effectively, these business policies need to be translated into deployable system configurations or lower level policies for multiple diverse systems. These target policy representations require experts to coordinate and collaborate so that business policies are fully supported. It is difficult and cumbersome to effectively ensure that all access control policies are enforced with the desired effect and in a consistent way, particularly given that there may be many people editing policies and that business policies can change over time. We present a language driven approach that abstracts access control policies into a clear and structured set of rules defined using terms familiar to a non-systems expert, which may then be realized into multiple levels of abstraction. Our proof of concept system uses Language-Driven Development (LDD) techniques to transform high level business policies into device specific policies that can be enforced by multiple access control system types. Our scenario examines the application of access control to instant messaging communications and network server access, two systems with different access control configuration languages.**

I. Introduction

The control of access to electronic resources within an organization is typically carried out using some means of policy based management. Policies offer administrators a highly flexible means to abstract the intended behavior of a system from the data that characterizes it. This ensures that access control policies can be updated rapidly if a change in the organizations policies prescribe. Business policies may define the access control requirements for an organization at an abstract level and typically in a language that is designed to be interpreted by a human administrator and not a computer. However, many human experts must interpret these policies and define access control policies using a computer understandable language so that access control decisions can be enforced upon requests for resource access. A concern with this existing approach is that there are many different target access control policy languages for the different types of electronic resources. For example, Extensible Access Control Markup Language (XACML) policies may be used to control access to document repositories or hospital records, and IP table rules may be used to control access to network ports on a web server or file server. Often there must be strong collaboration between the authors of these disparate system's specific policy languages, otherwise security holes can occur.

Individual access control policies are authored by system experts with extensive local knowledge but have limited knowledge of other domains. This harms the ability of security experts to collaborate and ensure their policies are all fulfilling their access control requirements in a consistent way. Additionally, as business policies change, these changes must be propagated to all affected access control systems which is a cumbersome and slow process. The challenges associated to ensuring consistent access control across an organization becomes increasingly difficult as more access control systems are deployed.

A promising alternative is to define the access control policies in a structured language tailored for use by business level administrators. These high level policies can then be processed and translated into the various system specific access control policies to be validated and verified by local security experts. Another advantage is that the high level policies can be analyzed for correctness and any potential conflicts that may occur across multiple security systems can be detected and highlighted to the appropriate policy author. This paper presents a language driven framework to be used by business level policy authors, that can be extended to cater for many target access control systems. We describe a prototype solution that has a business level access control policy language, which is processed into the XACML policy language [1] and also into IP tables based firewall policies. §II presents related work in the area. §III presents the framework that underpins the implementation.[1] We evaluate our work in §IV, and finally we conclude and comment on future work in §V.

II. Related Work

The need for a policy authoring process was motivated by Davy et. al in [2]. They highlight that there are typically many policy systems deployed in an organization that require synchronization to a guiding set of business policies. The presented authoring process was defined against a formalized structuring of policy sets termed the *Policy Continuum*. First mentioned by Strassner [3], the *Policy Continuum* abstracts business policies that are defined at a high level of abstraction

[1]The work described in this paper has been illustrated in a screencast. http://www.youtube.com/watch?v=cZtzgViRg8w

978-3-901882-50-0

from device specific policies that may be in many different policy language formats. The authoring process controls the flow of authoring the policies in the policy continuum so that policy editors at any level of abstraction can be notified of indirect changes to their policies typically made due to changes in higher level policies. Also policy analysis processes are implemented to maintain the consistency and correctness of deployed policies at each level. Our work makes use of the policy authoring process and the policy continuum model to abstract the multiple device policy languages from a set of business level policies.

In previous work, we examined how policy refinement and conflict analysis can be carried out in the context of a policy continuum [2] [4], in particular if the policy continuum spans multiple organizations [5]. During the authoring process, policy editors are notified while they are editing the policies, if there is a potential conflict in the current policy set. These notifications are facilitated by a conflict analysis process that takes into consideration policies deployed across all the device specific policy systems. This ensures that consistent behavior is observed in accordance with the initial business policies. A simple example would be a business policy granting access to a source code repository to a particular group of users. We would expect that the firewall policies would allow network traffic for the group of users to access the file server, and that sufficient access rights are also enabled for the group of users to read the files. By analyzing the policies before they are deployed, and taking into consideration previously deployed policies, the authoring process can maintain consistency between multiple device specific policies.

A related approach is outlined by Fitzgerald and Foley [6], who make use of Description Logic to abstract firewall policies. They then can reason over the deployed firewall policies to ensure that no conflicts arise and that high level business policies are enforced. Our approach goes further by providing feedback to a policy editor, and is extensible to support arbitrary policy languages.

Abstracting various policy languages into a formal language for analysis has been carried out previously for XACML policies [7], [8], [9] and for IP tables firewall policies [10], [11], [12],[13]. In particular, Hu et al. [8] developed a policy ontology for access control that can be used to represent both XACML policies and firewall rule policies. They make use of this ontology to represent the policies in a common format for policy analysis such as conflict and redundancy detection across both languages. Our approach also builds on a policy ontology that abstracts policies into a generic and extensible policy ontology. We make use of this policy ontology as the basis of policy analysis to be fed back to a policy author.

The approach outlined in this paper follows on from the concepts of *Federation* as presented by Jennings et al. [14]. They discuss how federation typically occurs at many levels between organizations, where legal agreements can describe specific obligations on each member of a federation. This may result in policies being deployed throughout each organization. Our approach is useful in this context in that the federation agreements can represent business policies at an abstract level which can be automatically deployed within an organization using our framework. The use of ontologies has also been highlighted as a useful approach to aid in the automation of federations between organizations [15].

Language driven approaches, where a structured language is defined for use by a specific constituency of user, follows on from model driven approaches, where an information model is used to generate tools for use by a specific constituency of user. In previous work, we examined how model driven approaches [16], [4] could be useful for aiding the automated deployment and analysis of policies for an organization. The leap to languages has been driven primarily through feedback on approaches and the maturity of toolkits available, namely the *Xtext* Framework [17] .

III. Language Driven Approach

Davy et al. [2] describes how the *policy continuum* model provides the basis of an approach for policy authoring in the domain of autonomic network management. In such scenarios, policies are specified in terms of an information model that is hierarchical in nature. The resulting policies can be arranged in a continuum, ranging from semi-abstract business policies to policies that can be implemented on devices with specific configuration parameters. Between the two extremes are policies with differing levels of detail and refinement, supporting different perspectives.

In a enterprise collaboration access control scenario, the (competing) business goals are

- *Allow* Principals to act upon resources within and between domains, so that *business operations* can proceed
- *Prevent* Principals from acting upon resources within and between domains, so that *security objectives* are met

This basic structure is more limited than that of network management policies described in [2]. However, when authoring access control policies such as those required in the collaboration scenarios above, it is very difficult to define policies that are correct both at the "macro" and at the "micro" level. Thus the policies should be

1) based on sound security principles addressing modern complex business relationships; *and*
2) specify device-level rules in sufficient detail that no weaknesses exist in their implementation.

Thus access control policies for collaboration scenarios need to be consistent across multiple perspectives, in a similar way to network management policies more generally. Consequently this paper considers how to take techniques such as *language driven development* that were applied to the network management policy continuum, and apply them to the collaboration policy continuum when multiple systems need to be managed.

A. Architecture

Three policy "levels" are defined, as depicted in Figure 1:

1) *Business:*
   These policies specify the high level rules governing

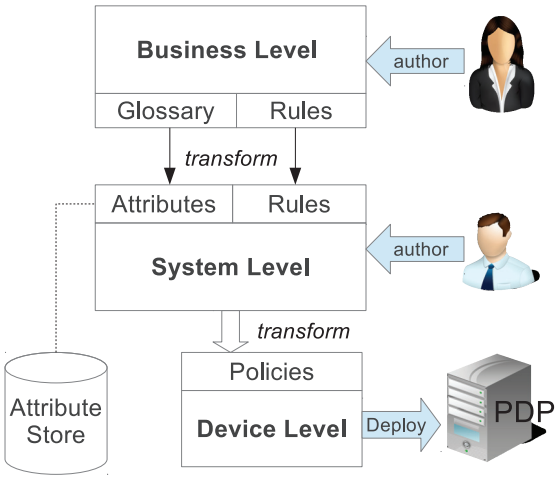

Fig. 1. Business, System and Device policy levels. Note that the (attribute) Knowledge base is separate from the Rule base. A *single* System-level policy set can be transformed into *many* Device-level policy sets, each targeting a PDP type.

```
allow CISCO UserGroup Developers
    to use CommunicationMode chat
    with CISCO UserGroup Managers
allow CISCO UserGroup Managers
    to use CommunicationMode chat
    with CISCO UserGroup Developers
allow CISCO UserGroup Developers
    to use CommunicationMode chat
    with TSSG UserGroup TSSGStudents
allow CISCO UserGroup Developers
    to use CommunicationMode chat
    with TSSG UserGroup TSSGPostDocs
allow TSSG UserGroup TSSGStudents
    to use CommunicationMode chat
    with CISCO UserGroup Developers
```

Fig. 2. Sample *Business* level policies. Identifiers specific to this domain include `CISCO`, `Developers`, `chat`, etc. Such terms have entries in a *Business* Glossary and their relationships are specified in a *System* Knowledge Base.

collaboration, such as "Group A can talk to Group B, but cannot share business reports with Group B". The focus is on conforming with good business practices, controlling information flows based on "need to know" principles, and employing security and risk principles such as *separation of duty*.

2) *System:*
These policies refine the business policies by specifying missing details such as what the groups are, who their members are, and what media types are covered by policies. Typically this is where much of the domain knowledge, captured in the form of ontologies and relational models, is added to the existing policies.
3) *Device:*
These policies are ready to be loaded into Policy Decision Points and are consistent with the requests that are generated whenever a Principal wishes to perform an action on a digital resource (example resources include a communication session, a webcam and a spreadsheet).

Following language-driven development principles, each policy level is associated with a Domain Specific Language (DSL) that balances the competing objectives of being both expressive and concise. In turn, the language hierarchy is realised in a layered architecture, see Figure 1.

We have two types of repositories, namely knowledge base and rule base, in order to store different kinds of information. Static attributes and their relationships are captured in a knowledge base. This knowledge base serves two purposes, namely maintaining relationships (write-oriented) and supporting reasoning over the rules (read-oriented). Rules defined in terms of the semi-static attributes are captured in a rule base. The rule structure is *For* (*Resources*) *IF* (*Conditions*) *THEN permit — deny*. Rule-combining algorithms combine the effects of all the rules in a policy to arrive at a final authorization decision, which include deny-overrides, ordered-deny-overrides, permit-overrides, ordered-permit-overrides, as well as first-applicable algorithms.

There are some advantages of separating the knowledge base and rule base. First, one attribute knowledge base implementation can be replaced with another that is more suitable for that use case, which offers the same functionalities. Second, the knowledge base can facilitate sharing common knowledge. For example, the business knowledge required to define firewall rules (IP traffic level) is related to that needed to define group chat and media sharing rules (XMPP traffic level). The system consistency can be achieved by sharing the business knowledge across different policy components. Third, it is easier to identify overlap and discrepancies between separate rule sets in the rule base. One or more legacy systems, each operating in its own closed world with its own administrative procedures, can have hidden problems that only become apparent when one of the systems are changed. By consolidating them into a single rule base, conflict and consistency analysis can be performed to alleviate the rule overlap and discrepancies.

B. Language driven development

The *Xtext* Framework [17] is used to automate the policy specification. State transitions are defined by rules that are specified using a grammar. Transforming from an abstract language to a less abstract language (*refinement*) can be performed by inspecting both grammars and by defining the transformation rules that map one grammar element into one or more grammar element in the less abstract language. Closer integration with static attributes reduces the abstraction level and leads to more concrete rules. Figure 2 illustrates a sample policy that was created using the XText framework. Notice that the language used is very close to natural structured sentences, ensuring maximum accessibility to non-experts.

The generic business level polices are translated into more concrete and meaningful system level policies by making use of the knowledge base. For example, in the business policy level, resource access polices are generic. More detailed

```
MATCH [
    REF [
        TYPE: 'Local',
        DESCRIPTION: 'Federation Communication
            between CISCO and TSSG'
        ]
    SUBJECT [
        LU_UserGroup.UserGroup_id = Developers
        AND
        LU_UserGroup.UserGroup_type_ds = CISCO
    ]
    RESOURCE [
        LU_UserGroup.UserGroup_id = Managers
        AND
        LU_UserGroup.UserGroup_type_ds = CISCO
    ]
    ACTION [
        LU_CommunicationMode.CommunicationMode_id
            = chat
    ]
]
DECISION : PERMIT
```

Fig. 3. Sample *System* level policies, derived from those in Figure 2. Note the same set of identifiers is used, but the *System*-level Rule Base is structured more like a query language.

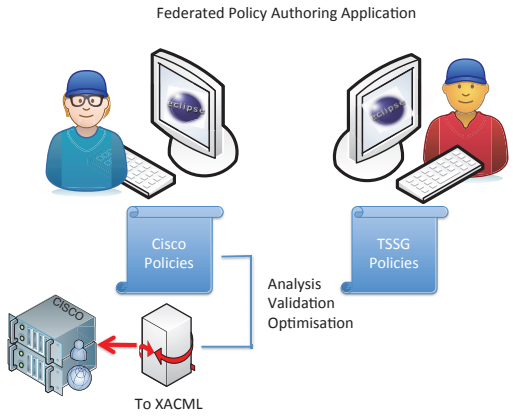

Fig. 4. Policy authoring across two domains. This is a still from Page 6 of the video demo described in footnote 1.

information will be taken from the knowledge base and considered in the lower level polices to complement the business-level generic "access" rules. We take two examples: one is used to "access" the communication method and the other one to "access" a code repository. When allowing members of a specific group to access a "communication method", e.g, video conference, the information from the knowledge base about membership information, the associated firewall rules and QoS policies should be taken into account. When accessing the code repository, for example a Git-based code repository, the system level policies must be compatible with the access policies associated with the code repository and policies related to system securities. Figure 3 illustrates the intermediate language we developed to make the high level business policies more concrete. The language should typically be used by a policy systems expert having knowledge of policy priorities and conflict handling. Many forms of conflict resolution are supported in this intermediate language, such as deny-overrides, first-match, first-permit etc.

The policy management tool chain enables authors to isolate where changes (inserts, updates, deletes) are made, and thereafter to generate artefacts that are consistent with the new policy state, removing the need for tedious and error-prone editing of complex and user-unfriendly editing of multiple device-level policy artefacts.

IV. Evaluation

We evaluated our work in the context of a federated access control scenario, where two organizations agree to federated in accordance with a pre-determined contract. Within this contract are the terms on which resources can be shared. These resources may be source code repositories, document servers, or communication services. We have developed the *Federated Access Control Authoring Application* which is used by non-system experts to create and modify access control policies for a number of systems under influence by the terms of the federation contract.

Specifically, we define policies that constrain instant messaging between two organization through the use of a policy-controlled message interceptor. The interceptor receives IM communications leaving and entering each organization and can query and enforce XACML based policies on these messages. We also constrain access to network attached storage devices by blocking access to network address ranges and ports at the IP level using a linux based firewall. Thus we can enforce policies that can constrain communication via instant messaging and access to network based resources across the federation.

Figure 4 illustrates the authoring application which is based in the Eclipse XText framework [17]. XText allows a system expert to specify a Domain Specific Language that can be used by non-system experts to describe relatively complex statements in a constrained and informed manner. The System experts in our case are individuals that have in-depth experience in defining communication access control policies to be applied using XACML and firewall policies. They abstract the concepts from these low level policy languages into concepts amenable to non-expert users such as "Allow" and "Permit" keywords and identifying groups of users and resources based on common name references that can later be resolved to concrete references.

The authoring application guides the non-system experts while they are defining the high level policies by providing prompt and informative feedback at runtime. The feedback, facilitated by XText warning and error notification system, can alert the authors to inconsistencies or anomalies in the policies they defined. We have augmented the *Xtext* warning and error notification system with linkages to our policy analysis processes. Subsequently, we can query knowledges bases, stored in OWL based ontologies, to ascertain if there are 1) existing deployed policies rules that may be negatively

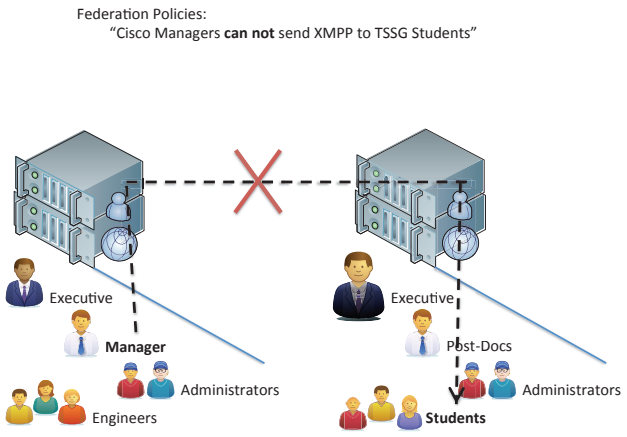

Fig. 5. An example *Business*-level federation policy denying XMPP communication between two named groups, one in each domain. This policy is transformed within each domain, using local knowledge and shared capabilities, to suitable device level policies.

affected by the deployment of these policies, or 2) semantic based inconsistencies are asserted if the policies are deployed. Our analysis processes are highly extensible and can be readily upgraded to detect for more elaborate forms of policy anomalies.

An example policy that may be defined between the two organisations is depicted in figure 5. This policy describes that within two organisations, there are particular teams that must not communication over XMPP. This can be facilitated using XACML and IP firewall rules. However, the firewall rules cannot be too stringent at they need to be flexible enough to allow other groups to communicate within the federation. This policy may also be incompatible with the existing policy deployment, or may even be redundant if it is fulfilled by an existing set of policies. Our policy analysis techniques will be designed to cater for these types of scenarios.

V. Conclusions and Future Work

Defining policies for a large organisation that need to be deployed onto multiple target policy systems is a cumbersome process that involves the participation of many domain experts. Our system aims to alleviate the need for the involvement of many domain experts as we separate out the various policy languages into distinct levels of the policy continuum. We implemented a proof of concept system that can generate validated XACML and firewall policies in a controlled scenario involving communication across administrative boundaries. A Language Driven Approach was adopted and facilitated by the XText framework. We see this as a very promising approach towards realising the policy continuum concept for targeted scenarios such at those presented in this paper. Future work will progress techniques for multiple system policy analysis and widen the scope of supported target policy systems.

Acknowledgment

The authors acknowledge funding from Science Foundation Ireland grant 08/SRC/I1403 "Federated Autonomic Management of End-to-End Communication Services". This work is also sponsored by the Irish Research Council in association with Marie Curie FP7.

References

[1] S. Godik, T. Moses, and Others, "eXtensible Access Control Markup Language (XACML) Version 1.0," *OASIS Standard, February*, 2003.
[2] S. Davy, B. Jennings, and J. Strassner, "The policy continuum - policy authoring and conflict analysis," *Computer Communications*, vol. 31, no. 13, pp. 2981–2995, Aug. 2008.
[3] J. Strassner, *Policy-Based Network Management*. Morgan Kaufmann, 2003.
[4] K. Barrett, J. Strassner, S. van der Meer, W. Donnelly, B. Jennings, and S. Davy, "A policy representation format domain ontology for policy transformation," in *2nd IEEE International Workshop on Modelling Autonomic Communications Environments (MACE2007)*, San Jose, CA, USA, Oct. 2007.
[5] J. Barron, S. Davy, and B. Jennings, "Conflict analysis during authoring of management policies for federations," in *Proc. 1st IFIP/IEEE International Workshop on Managing Federations and Cooperative Management (ManFed.com 2011)*. IEEE, 2011, pp. 1176–1183.
[6] W. M. Fitzgerald, S. N. Foley, and M. Ó Foghlú, "Network access control configuration management using semantic web techniques," *Journal of Research and Practice in Information Technology*, vol. 41, no. 2, May 2009.
[7] V. Kolovski and J. Hendler, "XACML Policy Analysis Using Description Logics," in *15th International World Wide Web Conference*, vol. 44, no. Www, 2007, pp. 494–497.
[8] H. Hu, G.-J. Ahn, and K. Kulkarni, "Ontology-based policy anomaly management for autonomic computing," in *Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom), 2011 7th International Conference on*, 2011, pp. 487–494.
[9] W. M. Fitzgerald, S. N. Foley, and M. Ó Foghlú, "Network access control interoperation using semantic web techniques," in *6th International Workshop on Security in Information Systems (WOSIS 2008)*, June 2008.
[10] Z. Fu, S. F. Wu, H. Huang, K. Loh, F. Gong, I. Baldine, and C. Xu, "IPSec/VPN Security Policy: Correctness, Conflict Detection and Resolution," *in Proc. of the IEEE International Workshop on Policies for Distributed Systems and Networks, POLICY*, 2001.
[11] E. Al-Shaer and H. Hamed, "Firewall policy advisor for anomaly detection and rule editing," *in Proc. of the Eight IEEE/IFIP International Symposium on Integrated Network Management, IM*, pp. 17–30, 2003.
[12] W. Fitzgerald, S. Foley, and M. Ó Foghlú, "Confident firewall policy configuration management using description logic," in *12th Nordic Workshop on Secure IT Systems*, Reykjavik, Iceland, Oct. 2007.
[13] M. Abedin, S. Nessa, L. Khan, and B. Thuraisingham, "Detection and Resolution of Anomalies in Firewall Policy Rules," *in Proc. of the 20th Annual IFIP WG 11.3 Working Conference on Data and Applications Security*, pp. 15–29, 2006.
[14] B. Jennings, R. Brennan, W. Donnelly, S. N. Foley, D. Lewis, D. O'Sullivan, J. Strassner, and S. Van Der Meer, "Challenges for federated, autonomic network management in the Future Internet," *2009 IFIPIEEE International Symposium on Integrated Network ManagementWorkshops*, pp. 87–92, 2009.
[15] D. O'Sullivan, J. Strassner, and S. van der Meer, "A snapshot of ontological approaches for network and service management," *Journal for Network and Systems Management*, vol. 17, no. 3, pp. 231–233, Sep. 2009.
[16] S. van der Meer, S. Davy, A. Davy, R. Carroll, B. Jennings, and J. Strassner, "Autonomic networking: Prototype implementation of the policy continuum," in *Proceeding of the 1st IEEE International Workshop on Broadband Convergence Networks*, Vancouver, Canada, Apr. 2006.
[17] S. Efftinge, "oAW xText: A framework for textual DSLs," *Workshop on Modeling Symposium at Eclipse*, 2006.