Harnessing Models for Policy Conflict Analysis

Steven Davy and Brendan Jennings

Telecommunications Software & Systems Group,
Waterford Institute of Technology, Cork Road, Waterford, Ireland
{sdavy, bjennings}@tssg.org

Abstract. Policy conflict analysis processes based solely on the examination of policy language constructs can not readily discern the semantics associated with the managed system for which the policies are being defined. However, by developing analysis processes that can link the constructs of a policy language to the entities of an information model, we can harness knowledge relating to relationships and associations, constraint information, behavioural specifications codified by finite state machines, and extensive semantic information expressed via ontologies to provide powerful policy analysis processes.

1 Research Problem

Existing approaches to policy conflict detection are primarily concerned with analysing the information contained within individual policies defined for a specific managed system. However, this approach, in general, does not take into account application specific semantics. This semantic information can be represented using information models and ontologies relating to a specific managed system. By tightly coupling a policy language to a rich information model policy conflict analysis processes can begin to harness this information and use it to detect potential policy conflicts, in particular application specific conflicts.

Much research on policy conflict detection has dealt with domain independent policy conflict, which is concerned with the modality of policies, most notably by Lupu and Sloman in [1]. Dunlop et al. [2] detect possible occurrences of domain independent conflict between the modality of policies; taking into account the detection of conflict based on overlapping events to predict runtime conflict. This PhD programme is concerned not only with conflict analysis for domain independent conflict but also analysis for application specific conflict. Bandara et al. [3] propose a policy conflict analysis approach for domain independent and application specific conflicts. However, their method of application specific conflict detection is based on constraints only and the policy language is not tied directly to an explicit information model. Instead, they translate the policies into a logic program based on event calculus, and examine this to detect conflict.

Application specific conflicts that arise solely due to the behaviour of the managed system have been examined in [4, 5], where the implicit behaviour is how IP packets are processed by network interfaces for both firewalls and IPsec encapsulation and conflict is detected through examination of the individual IP rules by dedicated algorithms. In [6] Chomicki et al. describe the use of action-constraints over policy

actions to explicitly detect occurrences of conflicting actions at runtime. Their approach focuses on action cancellation and event cancellation where the actions are ordered by priority within a constraint, so that lower priority actions are prevented from being executed. This work was further extended by Bertino et al. [7], who present methods of incorporating user specified preferences for prioritisation of conflicting actions. However, this approach relies on the explicitly relating policy actions together; in contrast, we propose to automate the creation of these relationships by leveraging the information model.

The challenge is to develop algorithms and processes that take full advantage of this source of rich information to aid in the discovery and detection of conflicting policies. Four approaches will be taken in this work, where they can be used exclusively or in combination with each other to achieve the desired goal. However all algorithms developed will fit into an overarching generic process detailing the phases of policy analysis.

2 Approach

In order to tightly couple a policy language to an information model, we developed a process that enables the generation of an integrated suite of languages and tools for policy specification, analysis and deployment [8]. Basing the process on MDA (Model Driven Architecture), the information model described in UML formed the starting point for generating the policy language and related analysis tools. The tools generated can query over both the policy language, and information model thus enhancing policy specification, analysis and deployment. We believe that the refinement of policies must be closely aligned with the conflict analysis of policies.

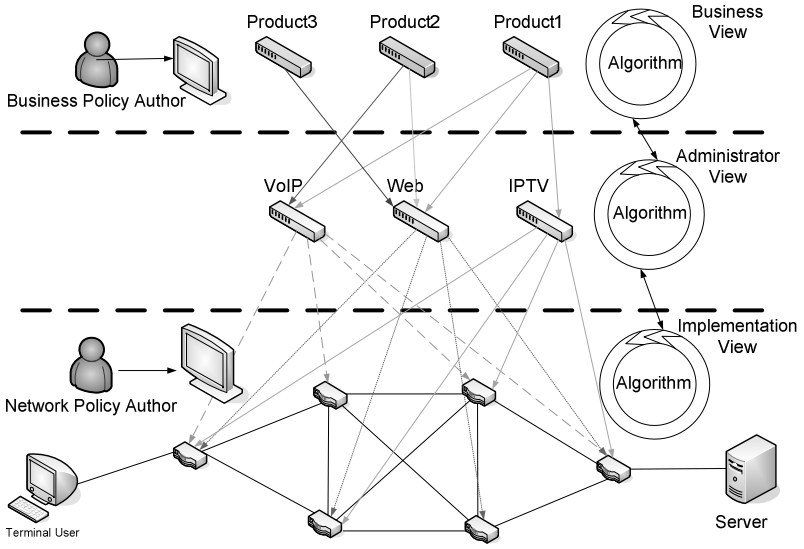

Figure 1: Policy Refinement and Analysis

The scenario our policies are based on details the provision of various internet service products to subscribing customers [8]. There are three views of the managed system: the business view, the administrator view and the implementation view. As illustrated in Figure 1 policies at the business view are transformed to policies at the implementation view. The following lists various ways in which the information model can be harnessed to provide effective policy analysis:

- *Association and Relationships*
  A customer referenced in a policy specification can be stored as an identifier within a database, where we can access the database using the identifier to get information about the customer. However this information will not describe the customer's relationship to other entities in the system. For example, an association between classes in an information model can indicate that a customer can be associated with a set of purchased products. This information can then be used to ascertain if a given customer is related to a given product. A conflict may arise when the result of a policy causes the deactivation of a product at the implementation view, where this conflicts with the provision of that product to a customer at the business view. An approach based exclusively on analysing the constructs of the policy language may not be able to explicitly make the connection between implementation view policies, and business view policies.
- *Model Constraints*
  Another method of leveraging the information model is to analyse constraints defined over the properties of modelled elements and associations, so that we can detect if these constraint are breached by deploying a policy. For example, a constraint over the associations between ethernet interfaces in the information model can specify that the bandwidth of provisioned services that use these interfaces be limited to 80% of link capacity. By making this constraint available to the policy analysis component, we can detect a constraint breach, and thus a policy conflict when an existing customer upgrades their policy and subsequently too much bandwidth is provisioned on the associated ethernet interfaces. In [9] we demonstrated that by examining information model based constraints, that specific form of policy conflict can be prevented by further refining the specific policies to only be applicable in cases where constraints are never breached.
- *Ontologies*
  An information model ontology provides richer semantics than can be achieved with associations, and constraints. Ontologies can represent relationships between concepts, and individuals, and provides reasoning capabilities over these. When the information model is enhanced with ontological concepts, more in-depth restrictions on policy can be enforced and reasoned over for the purpose of policy analysis. For example, a high priority voice service may be assigned a PHB (per hop behaviour) of AF31 (assured forwarding), however in case this service cannot be classed as AF31 due to network restrictions we can class it to an "equivalent class" such as AF32, and re-deploy the policy. Relationships as equivalence and disjoint can be easily represented in an ontology, but not in a UML based information model. In [8], we describe how to build a base system ontology that can be enhanced to describe more extensive system semantics.

- *Finite State Machines*
  Finite state machines (FSMs) are used to describe the behaviour of an entity using input events to states causing state transitions. Using FSMs we can associate behaviour to managed entities. Future research will investigate how to best take advantage of FSMs to detect occurrences of unwanted behaviour. For example, as policy is being deployed it affects the relevant states of managed entities; the enumeration of state across the system is a snapshot of current behaviour. Since we know from the FSMs the potential next states we can devise algorithms to discover combinations of unwanted states.

3 Future Work

The usefulness of combining the above mentioned approaches to exploit information models of a managed system will be investigated, specifically combining finite state machines with ontologies. One aim is to introduce a tiered FSM where lower levels of the machine describe the behaviour of individual managed entities and higher levels of the machine describe the interaction of the system as a whole. Therefore mis-behaviour at the lower levels due to mis-configuration or policy conflict will propagate to upper levels, and the relevant policies can be flagged for analysis.

References

1. Lupu, E.C., and Sloman, M., Conflict in Policy Based Distributed Systems Management. IEEE Transactions on Software Engineering, Vol. 25, No. 6, pp 852-869 (1999)
2. Dunlop, N., Indulska, J., and Raymond, K., Dynamic Conflict Detection in Policy-Based Management Systems, in Enterprise Distributed Object Computing Conference, pp 15-26 (2002)
3. Bandara, A. K., Lupu E. C., and Russo, A., Using Event Calculus to formalize policy specification and analysis, In 4th IEEE Workshop on Policies for Distributed Systems and Networks (2003)
4. Cholvy, L. and Cuppens, F., Analyzing Consistency of Security Policies. IEEE Symposium on Security and Privacy, pp103-112 (1997)
5. Bandara, A.K., Kakas, A., Lupu, E.C., Russo, A., Using Argumentation Logic for Firewall Policy Specification and Analysis, in Proc 17th IFIP/IEEE Distributed Systems: Operations and Management (DSOM) pp185-196 (2006)
6. Chomicki, J., Lobo, J., Naqvi, S., Conflict Resolution Using Logic Programming. IEEE Trans. Knowl. Data Eng. 15(1): pp244-249 (2003)
7. Bertino, E., Mileo, A., Provetti, A., PDL with Preferences. 6th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY 2005), pp213-222 (2005)
8. Barrett, K., Davy, S., Jennings, B., van der Meer, S., Strassner, J., Model Based Generation of Integrated Suites of Languages and Tools for Policy Specification, Analysis and Deployment, submitted to IEEE Workshop on Policies for Distributed Systems and Networks (2007)
9. Davy, S., Jennings, B., Strassner, J., Policy Conflict Prevention via Model-driven Policy Refinement in Proc 17th IFIP/IEEE Distributed Systems: Operations and Management (DSOM) pp209-220 (2006)