Usable Policy Template Authoring for Iterative Policy Refinement

Maritza Johnson
Department of Computer Science
Columbia University
New York, NY 10027
maritzaj@cs.columbia.edu

John Karat, Clare-Marie Karat
Keith Grueneberg
IBM TJ Watson Research Center
Hawthorne, NY 10532
jkarat@us.ibm.com ckarat@gmail.com
kgruen@us.ibm.com

***Abstract*—People must have usable tools in order to author and maintain high-quality policies. In this paper we discuss policy templates as a mechanism for policy authoring. We believe that policy templates can be leveraged to make policy authoring more usable and to provide consistent policy authoring interfaces across a wide variety of policy domains. Templates provide users with a structured format for authoring policies; however, a general approach for creating policy templates has not been described in published research to date. Based on research in policy management, we propose an iterative policy refinement process that consists of three user roles and spans policy authoring, template authoring, and policy element definition. We designed a GUI-based prototype that enables users to create policy templates. In this paper we describe our proposed policy refinement process, the necessary user roles, a template authoring prototype, and the results of an empirical study of template authoring.**

I. Introduction

In this research, we consider policies as rules that define desired system behavior. A policy-driven system is able to offer flexible functionality by having policies define some of the actions to be taken by a system rather than having all actions embedded in application code. Policy-driven systems and applications can provide valuable flexibility for organizations and users but in practice the difficulties of producing high-quality policies discourages system designers from utilizing policies to their full potential. For our purposes, a high-quality policy is one that captures the intent of the user, is implementable and is understandable by the people who work with it [1]. High-quality policies are critical because the system will enforce the policy as specified and problems will arise if the policy does not match the policy authors intent. In these systems the policy author is a person or team of people tasked with translating high-level goals into implementable policies [2]. Since policies are written by people, policy-authoring tools must be designed with a human-centered approach.

Based on the related work, we researched an iterative human-centered policy authoring process to improve the usability of policy management. A primary focus of the work is defining user roles based on the tasks required in the policy refinement process and creating usable tools for the roles. We suggest the roles are policy author, policy template author, and policy element author. By isolating the technical details to the appropriate roles, our process allows a broad range of users to create high-quality policies.

In our ongoing research we have found there is a communication gap between the people in the business and technical sides within an organization when it comes to policy definition and implementation [3]. To bridge the gap, we propose that a process that uses policy templates can help facilitate communication between the business and technical users. A policy template is a structured policy statement with domain specific policy elements that enable the policy author to produce policy rules by selecting values for the policy elements. The policy element author is a technical user and isolates the technical details of the policy implementation and enforcement. The template author creates templates from the policy elements, and the policy author selects values for the policy elements to author new policies. Our process provides a more complete policy refinement process by which users and organizations can iterate on policy as requirements change. To give an example of an access control policy template with generic policy elements:

{*Subject*} can {*Action*} {*Target*} if {*Condition*}.

The following is an example from the medical domain of a policy that could be written using the template:

{*Doctors*} can {*read*} {*name*} or {*current medication*} if the {*patient*} has been {*admitted*}.

This example demonstrates how domain specific values are chosen for each policy element. *Doctors* is a role that users can be assigned to. *Read* is associated with the privilege to access the data fields *name* or *current medication*. And the *patient* is a user in the system that has a property *admitted* that has a value of true or false. The policy elements are defined such that they tie the policies to the system implementation, but these are details that do not need to be revealed to the policy author. Glue text connects policy elements to form a natural language policy statement. The glue text in the sample policy are can, or, if, the, and has been. The glue text is used in the template to support readability for the policy author and all users of the policy. Subject, action, and target are required elements for an enforceable access control policy and a condition is optional. An access control policy with these elements specifies when someone can take particular actions regarding the specified

resources.

The purpose of this research was to explore the design and effectiveness of a usable template authoring interface and to understand the user roles in policy refinement. We present the current state of our policy refinement process, discuss the design and implementation of a template authoring prototype, and briefly present results of an empirical laboratory study with the prototype. Our analysis suggests three user roles to establish an iterative policy refinement process and the study results suggest our prototype is a usable tool for template authoring.

II. Related Work

Human-centered design processes have not been extensively applied to policy refinement, the task of mapping high-level policy goals to a low-level implementable representation [4]. Some usable security researchers have begun to consider the policy refinement process [5]. Though most usable policy authoring research is domain specific. Zurko *et al.* developed an interactive GUI for RBAC access control policies [6]. Firmato was designed for managing firewall policies [7]. SPARCLE was initially designed for organizational privacy policies [3]. Expandable Grids is a policy visualization tool, its use has been evaluated for file access control and privacy policies [8]. Grey was designed for physical access control [9].Each of these was developed with an emphasis on usability and each made a contribution to usable policy authoring. But, it is important to research more general policy authoring techniques because solutions needed that extend to a distributed or heterogeneous system.

General policy authoring was considered to identify common challenges in authoring security and privacy policies [10]. This work identifies the following guidelines for policy authoring systems: support object grouping, enforce consistent terminology, make default rules clear, communicate and enforce rule structure, and prevent rule conflicts. These guidelines and the benefits gained from structured-entry demonstrated by empirical evaluations of SPARCLE suggest policy templates are a strong base for a policy authoring process.

SPARCLE is a policy management workbench for privacy, security, and other domains [3]. Users author policies using either guided natural language or structured-entry. Guided natural language (NL) allows users to author policies by typing free-text sentences in a word processor window. A syntax guide above the window provides information about the required policy elements and communicates the rule structure. For natural language entry, SPARCLE employs a shallow natural language parser which employs a grammar to guide it in extracting policy elements from the sentence. Structured-entry presents users with a predefined set of policy elements where the policy author selects values for the policy elements from dropdown lists; this method is similar to using a policy template. Karat *et al.* describe the policy elements common to privacy policy rules and discuss how the values are populated using domain knowledge. Empirical results show guided NL and structured-entry produce higher-quality policies compared to unguided natural language.

The extensibility of SPARCLE is an area in need of further research. Extending SPARCLE to a new domain requires generating a NL grammar and defining structured element lists specific to that domain. Both are required because the tool provides NL and structured-text versions of each policy, and allows either authoring method to be used by the policy author at any time. Generating a robust NL is a significant amount of work [3]. For this reason, the research presented in this paper focuses on a template-based approach that will provide the policy author with the same user experience as authoring a policy with the structured-entry method in SPARCLE. A tool for template authoring - i.e., creating templates from which policies will be authored - can meet the need to define structured-entry lists for new policy domains.

Mont et al. proposed POWER which used policy templates based on domain specific policy elements [11]. The paper is the first to discuss two separate user roles for policy refinement: the expert and the consultant. The "expert" has technical knowledge of the system to create policy templates. The "consultant" has experience in business policies and must be capable of expressing high-level policies using the existing templates. The prototype is similar to our template-based process but the prototype was not implemented nor was it tested. We take the work further by defining the roles required for the process, defining the tasks related to the roles, and implementing a prototype.

III. User Roles in Iterative Policy Refinement

Previous research defined a policy management framework that identifies abstract steps for translating high-level goals to runtime policies [5]. The framework identifies policy authoring, conflict and coverage analysis, and deployment as the activities in a policy lifecycle. the framework divides the translation steps into several layers where the policy has a different representation at each layer. The policy moves from natural language in the specification layer, through an intermediate representation in the abstract layer, to an implementation-specific representation in the execution layer. Policy authoring is a stage in this framework but it has received little attention. There is no work that defines the *user roles* associated with the refinement of policies from high-level goals through executable representations.

This paper builds on the existing policy management framework by defining user roles and components that form an iterative policy refinement process. Based on our user research in the policy authoring domain, we propose that an iterative policy refinement process has at least three user roles. The process is based on people authoring policies using policy templates. In the general case the interactions between the roles will be: the policy author creates policies using templates, the template author creates templates using policy elements, and the policy elements are authored to represent the system objects and resources that are unique to the domain or the system. The process is divided into three roles based on the

technical or domain knowledge needed to complete each role's tasks.

The policy element author is responsible for defining policy elements specific to the domain and system implementation. A user who is familiar with the domain and technical details of the system should complete this task. There should be a policy element to represent everything in the system that might be used in a policy. This includes all users in the system, devices in the system and possible actions. The policy elements are abstractions of low-level system operations and are used by the template author. The values of the policy elements cross into the specification layer since the policy author will be choosing actual values for the policy elements when they author a policy. The primary benefit of policy elements is the policy author will only see values that are meaningful in the domain while masking the technical complexity of the system.

The template author is responsible for creating policy templates that will enable the policy author to create specific, implementable policies. The templates are a level of abstraction higher than the actual policies. The template author must be familiar with the policies that will be written for the domain, the domain itself, and have some understanding of the technical side of the organization. They will create abstract policy statements from the available policy elements.

The policy author is responsible for creating policies using the policy templates. The policy author's goal is to create high-quality implementable policies based on their domain expertise and business knowledge of the organization. Templates ensure the resulting policies are valid in for the domain.

The process is iterative as the organization and policies change over time. New policies can influence the creation of new templates and policy elements. Additionally, the policy author can request new policy elements or templates, and the template author can request new policy elements. Alternatively, the policy elements and their details may help shape the policies that are authored. To give an example, imagine a policy author would like to create a policy but finds the existing policy elements and templates are insufficient. The policy author can communicate this to the policy element author to request new ones, collaborate with the template author to create a new template, or seek advice on how to adjust their intended policy so it is expressible using the existing templates and policy elements.

IV. Template Authoring Prototype

The template authoring prototype was designed to support the template author in their role as the link between the policy author and the policy element author. The template author must write templates using a vocabulary of available policy elements that support the policy author's goal of authoring high-quality policies. Their understanding of the business goals and the technical capabilities of the system allows them to author templates. Based on task analysis and previous research in policy authoring [3], critical features include the presentation of policy elements, the entry method for creating a policy template, and the ability to preview a template from the policy author's perspective. We were conscious of scalability in respect to the number of policy elements and templates; in a real deployment the number of policy elements could be quite large. Other identified requirements are to communicate context information through the presentation of the policy elements and to provide a search mechanism.

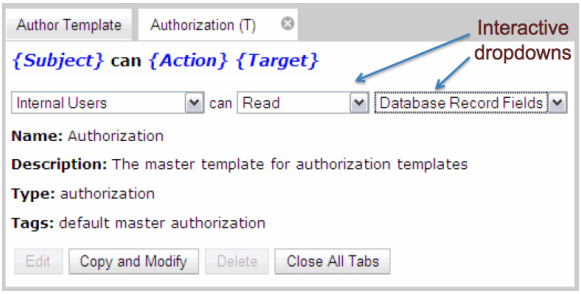

Fig. 1. The master template for authorization templates

In our prototype a template has the fields: template text, name, description, tags, type, and author. The policy author can use the values for name, description, tags, and type to understand which template to use. The template text is a natural language statement composed of policy elements and glue text. A policy element has the fields: name, description, type, tags, values, parent policy element, and related policy elements. The only required fields are name and values. The other fields are used by the template author for context information or by the interface for displaying the policy elements.

The template authoring interface has two panes. On the right, the template author creates templates in the template authoring tab by typing in a text editor or using a button to select and add policy elements. The tool provides a few master templates that the user can copy and modify to suit their needs. On the left, the policy elements and existing templates are displayed. The Elements tab displays a tree in the left pane. The user can click on a policy element to view the details. The Templates tab lists the master templates and saved templates. The user can click on a template to view the details. The details tab also shows the template as the policy author would see it (see Figure 1). The template author can interact with the dropdowns to experience what the policy author will see.

The Search Results tab displays the policy elements and templates that satisfy the search query. The tab dynamically changes to suggest related policy elements and templates based on the user's actions. For example, when a policy element is selected from the Elements tab the search results show the policy elements that are listed as related policy elements. This feature will help the template author discover related templates and policy elements. In the left-hand pane, the hierarchical organization of the policy elements presents the template author with varying levels of abstraction or specificity.

A. Empirical Evaluation

We conducted an empirical laboratory study with twenty participants to explore the usability of the template authoring

prototype and to better understand the user roles in iterative policy refinement. The participants were experienced computer users who had little to no policy authoring experience. Ideally, our participants would have been representative of policy experts in the appropriate domains but it is difficult or prohibitively costly to recruit such participants.

Before interacting with the prototype, the participant completed a short tutorial on policy authoring and read instructions for the template authoring prototype. Then the participant completed two template authoring tasks. For each task, a brief scenario was presented with a set of sample policies. The participant was asked to create templates that a policy author could use to create the sample policies in the scenario. One scenario described a web merchant who needed to specify access to their database. The other described a fictitious social networking website and asked the participant to create templates for allowing access to personal data.

For quantitative data, we collected the participant's templates and recorded the time to complete a task. For qualitative data, the participants completed post-task questionnaires and a debriefing was conducted. In addition, participants were asked to think-aloud as the study coordinator took notes from an observation room while the participants completed the tasks. A detailed description of the study design and a complete discussion of the results can be found in [12].

Overall, the participants were satisfied with the template authoring prototype and found it was easy to create policy templates. The most used feature was the ability to create new templates from the master templates provided in the prototype (see Figure 1).

All twenty participants completed both tasks in less than an hour. In addition, all twenty participants successfully authored functional templates. Functional templates, for the purpose of the study, is a set of templates that can be used to author all the sample policies provided for the task. Three categories of templates emerged from our analysis of the participant authored templates: generic, mixed, specific. A discussion of the trade-offs between each category of templates and potential future work can be found with the description of the study [12].

V. Discussion and Future Research

Template authors play a crucial role in the policy authoring process as they are the link between policy element authors and policy authors. The data collected in our empirical study indicate non-policy experts are capable of authoring functional templates when given policy elements and sample policies in the context of a scenario. The participants had little experience with policy authoring and were able to author functional policy templates for two different policy domains after completing just a short tutorial. Our results support the claim that the template authoring prototype is usable.

Furthermore, although the study's primary goal was to evaluate the participant's ability to author templates, the data collected also provide important insight to develop guidelines for a usable security and privacy policy authoring framework [13].

We plan to evaluate template authoring more thoroughly by empirically evaluating the quality of the templates authored. One way to do this is to have a second participant author policies using the new templates and then evaluate the quality of the policies authored. The goal of evaluating the prototype described here was to explore how to facilitate the creation of templates from policy elements. As mentioned previously, empirical evaluation is needed to determine what characteristics make a template most effective. We may find there are situational demand characteristics that dictate the level of template specificity in real world use.

Acknowledgment

This research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the author(s) and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

References

[1] C.-M. Karat, J. Karat, C. Brodie, and J. Feng, "Evaluating interfaces for privacy policy rule authoring," in *SIGCHI*, 2006.

[2] C. Brodie, C.-M. Karat, J. Karat, and J. Feng, "Usable security and privacy: A case study of developing privacy management tools," in *SOUPS*, 2005.

[3] J. Karat, C.-M. Karat, C. Brodie, and J. Feng, "Privacy in information technology: Designing to enable privacy policy management in organizations," in *Int. Journal of Human-Computer Studies*, 2005.

[4] J. D. Moffett and M. S. Sloman, "Policy hierarchies for distributed systems management," *IEEE Journal on Selected Areas in Comms.*, 1993.

[5] J. Karat, C.-M. Karat, E. Bertino, N. Li, Q. Ni, C. Brodie, J. Lobo, S. Calo, L. Cranor, P. Kumaraguru, and R. W. Reeder, "A policy framework for security and privacy management," in *IBM Journal Research & Development*, 2009.

[6] M. E. Zurko, R. T. Simon, and T. Sanfilippo, "A user-centered, modular authorization service built on an RBAC foundation," *IEEE Security and Privacy*, 1999.

[7] Y. Bartal, A. Mayer, K. Nissim, and A. Wool, "Firmato: A novel firewall management toolkit," *ACM Trans. Comput. Syst.*, 2004.

[8] R. W. Reeder, L. Bauer, L. F. Cranor, M. K. Reiter, K. Bacon, K. How, and H. Strong, "Expandable grids for visualizing and authoring computer security policies," in *SIGCHI*, 2008.

[9] L. Bauer, L. F. Cranor, R. W. Reeder, M. K. Reiter, and K. Vaniea, "A user study of policy creation in a flexible access-control system," in *SIGCHI*, 2008.

[10] R. Reeder, C.-M. Karat, J. Karat, and C. Brodie, "Usability challenges in security and privacy policy-authoring interfaces," in *INTERACT*, 2007.

[11] M. Mont, A. Baldwin, and C. Goh, "POWER prototype: Towards integrated policy-based management," *IEEE NOMS*, 2000.

[12] M. Johnson, J. Karat, C.-M. Karat, and K. Grueneberg, "An empirical study of policy template authoring," in *submission to ACITA*, 2010.

[13] M. Johnson, J. Karat, C. M. Karat, and K. Grueneberg, "Optimizing a policy authoring framework for security and privacy policies," in *SOUPS*, 2010.