Optimizing a Policy Authoring Framework for Security and Privacy Policies

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

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

ABSTRACT

Policies which address security and privacy are pervasive parts of both technical and social systems, and technology to enable both organizations and individuals to create and manage such policies is seen as a critical need in IT. This paper describes policy authoring as a key component to usable privacy and security systems, and advances the notions of policy templates in a policy management environment in which different roles with different skill sets are seen as important. We discuss existing guidelines and provide support for the addition of new guidelines for usable policy authoring for security and privacy systems. We describe the relationship between general policy templates and specific policies, and the skills necessary to author each of these in a way that produces high-quality policies. We also report on an experiment in which technical users with limited policy experience authored policy templates using a prototype template authoring user interface we developed.

Categories and Subject Descriptors

K.6 [**Management of Computing and Information Systems**]: System Management, Security and Protection; H.1.2 [**Models and Principles**]: User/Machine Systems—*Human Factors*

General Terms

Security, Human Factors, Design

Keywords

Policy refinement, policy management, policy authoring, security policy, privacy policy, user experience design

Symposium on Usable Privacy and Security (SOUPS) 2010, July 14–16, 2010, Redmond, WA USA
.

1. INTRODUCTION

Policies are crucial to security and privacy domains. Policy-driven systems are important because they allow a level of dynamic customization that does not require a new cycle of requirements engineering and development. Policies are especially important in security and privacy. For example, as noted by Cheswick *et al.*, "The single most important factor of your firewall's security is how you configure it" [12]. This idea extends to any system that enforces a policy, the system will enforce the specified policy so it is crucial that the specified policy matches the intended policy.

In IT, there is a general notion of policy-based systems as those whose behavior is guided by rules of the general form "If condition then action." Collections of rules are considered policies, and policies can be developed for various aspects of system behavior. In social systems, organizations have policies covering proper conduct of people and effective use of resources. Information technology systems have policies which govern access to resources aimed at protecting the integrity and confidentiality of the information and resources. Individuals have policies guiding their behavior towards others formed with the intention of guiding how they live their lives. These policies might be expressed in natural language text (common in organizations), in executable code (common for IT systems), or might be implicit (common for individuals). Such policies might be seen as including high level guidance (e.g., "to insure a safe workplace") and more specific operational rules (e.g., "don't run with scissors").

We see several aspects common across a wide range of policy types in technical and social systems. First, high level policies – generally expressed in natural language – are refined into operational rules while attempting to preserve the intent of the high level policy. This process is difficult – often subject to differences in interpretation or context. Second, the existence of multiple, possibly conflicting, policies must be accommodated. This process is also difficult, as comparison across policies requires detailed understanding of the meaning of each policy rule and is rarely straightforward. For human or technology systems there is a resulting gap – sometimes referred to as the **gulf of execution** [29] – between human intentions and technology capabilities. We believe developing approaches to closing the gulf of execution would be valuable. For example, most organizations store sensitive business and personal data in heterogeneous server systems. They do not have a unified way of defin-

ing or implementing security and privacy policies regarding the storage and use of that data throughout their organization. Changing legal requirements, social pressures and technologies are making these issues increasingly critical to organizations and society at large.

While our research can apply to policies in many areas, we are particularly interested in security and privacy policies. There are several reasons for this focus. First, there are a growing number of strict security and privacy audit and compliance requirements for healthcare, banking/finance, and government. This creates a practical need for improving the management of such information through policy-based systems. Second, there is considerable similarity between privacy and security policies. Specifically, rules for a major component of security policies, namely, access control rules defining access to resources, are nearly identical to rules which are used in the formation of privacy policies [31]. This similarity in rule structure leads us to focus on bringing together research rooted in security policy analysis with research in privacy policy authoring and implementation. Third, privacy – when viewed as appropriate use of information – relies on the security of that information in a system, and it is difficult to talk about privacy without considering security. Our principal research objective is to create an integrated privacy and security policy management framework which builds on the commonalities between the two. This includes mechanisms and tools for supporting policy authoring which have grown out of previous work centered on privacy policies [10].

We see the policy authoring challenge as consisting of both providing a mechanism for expression of policy intent by domain experts, and providing a mechanism for translating human statements to IT systems. To illustrate the policy authoring challenge, consider the following authorization policies:

> 1. Healthcare staff can forward patient medical information for the purpose of national medical research if the information is anonymized.
>
> 2. Doctors can access laboratory results.

Each of these policies *authorizes* an action if a condition is met. The first policy is considered a privacy policy, while the second a security access control policy. While it might be claimed that these policies can be understood by people, the task of making them enforceable by IT systems is complex. Basic elements of the policies might be extracted (e.g., the users, actions, data, conditions and purposes of the privacy policy in 1), but mapping them to policy elements that a system can interpret as intended by the human authors requires IT knowledge and skills not possessed by many people responsible for establishing such policies in organizations. Context dependent definitions of policy elements like "doctors", "national medical research", or "anonymized" need to be tied to policy elements an IT system can process.

In this paper we outline our directions and progress to date, and present remaining research challenges in developing a framework for policy authoring. We begin by presenting the current state of policy authoring research in Section 2. Then we describe a policy authoring framework in Section 3. The framework extends the notion of policy authoring to include the role of template author. This additional role captures the need to map human concepts to system elements and requires skills beyond the business knowledge generally associated with policy formulation in organizations. In Section 4, we present existing guidelines for security and privacy policy authoring and supplement them by suggesting additional guidelines. Our intention is to provide a framework which is comprehensive enough to enable collaboration among researchers working on policy development and implementation in different areas of security and privacy.

2. RELATED WORK

Policy authoring is an important topic for the usable security community and has received a fair amount of research attention. To frame the discussion of usable policy authoring, we categorize the prior research into work that addresses policy authoring in general and work on domain specific policy authoring interfaces. We also discuss research from the computer policy community.

2.1 General Usable Policy Authoring

SPARCLE is a policy management workbench [24]. The initial prototype focused on privacy policies [10]. Later work demonstrated the extensibility of the workbench to other domains [9]. Using SPARCLE, the policy author creates policies using their choice of either guided natural language or structured entry. The ability to switch between the two representations is a feature in SPARCLE. The interface for guided natural language displays a syntax guide above the text area where the policy author types the policy. The guide increases their ability to write correctly structured policy statements. A natural language grammar is used to extract the policy elements from the natural language statements. To author a policy using structured entry the policy author selects values for policy elements from predefined lists. Empirical studies showed when policy authors used guided natural language or structured entry, they produced higher-quality policies compared to unguided natural language [24].

Reeder *et al.* conducted a user study to identify common policy authoring errors [32]. The results of the study were used to produce a set of guidelines for designing usable security and privacy policy authoring tools. The guidelines were driven by an analysis of the errors observed and are intended to eliminate the most common errors. The guidelines include: support object grouping, enforce consistent terminology, make default rules clear, communicate and enforce rule structure, and prevent rule conflicts.

The Expandable Grid is an interactive matrix for policy visualization. For the matrix, two policy elements are chosen to be represented on the x and y-axis, the intersection displays a graphical representation of the policy decision for the pair of values. The Expandable Grid is an alternative to the usual list-of-rules approach that is commonly used. An empirical study comparing Expandable Grids and the Windows XP file permissions interface suggests the Grid is more usable [34] for file access control management. Expandable Grids can be extended to other domains, like representing P3P policies [15]. An empirical study of P3P policy management tasks suggests Expandable Grids does not improve usability beyond the level achieved by expressing the policies with natural language statements [35].

2.2 Domain Specific Usable Policy Authoring

Role-based access control (RBAC) is a commonly used mechanism for expressing authorization policies for controlling access to system resources [37]. Adage is an authorization service built on RBAC in which a primary focus was using human-centered methods to design a policy-management GUI for administrators [41]. The GUI allows the user to group objects that were then represented by a single object in the interface. The direct manipulation features and the visual representation of object groupings for subjects and targets were shown to be beneficial to users.

ESCAPE is a tool for managing file access on the web [3]. The interface allows users to configure permissions implicitly through their actions. This was accomplished by granting read privileges to users when the content owner announced new material to them. Salmon is a policy interface designed for file access control [26]. The interface was designed to reduce the number of errors users made when authoring and reading file permissions by providing the user with a more accurate depiction of the overall policy.

IAM allows users to specify the effective access control policy for WebDAV instead of individual rules to achieve the same goal [11]. IAM takes the effective policy as input and suggests ways of achieving the specified policy. This increases usability by allowing the user to specify the end result without having to figure out the individual rules necessary. Grey is a physical access control system, a primary use of it is managing access policies for rooms at a university [5]. Research with Grey has explored the implications of allowing more fine grained policies than is possible with the usual lock and key.

In the effort to design a privacy agent for P3P policies, Cranor *et al.* discussed the problems related to designing an interface for users to express their privacy preferences [17]. Most problems were related to finding the appropriate vocabulary to express the complex language and concepts found in privacy policies, and structuring the interface to group common elements.

Firmato was designed to address the usability of firewall policy management [4]. A primary feature of the tool is the ability to abstract the policy from the mechanisms and network topology. Firmato also allows policy authors to use named objects to represent groups of IP addresses and port numbers. The GUI includes features for specifying the policy, visualizing the topology, and visualizing the permitted and denied connections.

2.2.1 Policy Visualization

Empirical evidence has shown that policy visualizations have advantages over list-of-rules policy representations. A list-of-rules representation requires the user to compute the effective policy in their mind which is difficult for a large rule set [34]. With SPARCLE even a simple two-dimensional representation, where two policy elements were used as the axes, effectively communicated the overall policy [10]. Expandable Grids, a grid visualization of the effective policy, also had positive usability results [34]. The Expandable Grid was compared with a list-of-rules interface for file access control. The list-of-rules interface was less usable and less effective at communicating effective policy.

Audience View is an policy authoring tool that gives the user visual feedback as they make policy modifications. With Audience View, the policy author configures privacy settings for a profile on a social networking website [25]. The user is able to view their profile as different groups of users would see it. Rather than manage a list of rules, the user clicks on the profile to show and hide information. An empirical study shows the visual feedback contributes to better usability than the list-of-rules interface on social networking websites.

2.3 Policy Languages

The design of policy languages that allow flexibility and maximal expressivity is a popular research direction. Flexible policy languages are useful to demonstrate that a wide range of enforceable policies can be specified. In practice, their usability is limited by whether a usable policy authoring interface is available. For an in-depth survey of policy specification approaches, including policy languages, see [18]. Ponder is a strongly typed, declarative, object-oriented language for specifying network management policies for heterogeneous systems [19]. Flexible policy languages are useful to demonstrate that a wide range of enforceable policies can be specified. In practice, their usability is limited by whether a usable policy authoring interface is available. Keynote and XACML are other examples of policy languages [8, 30]. Policy languages tend to look very similar to programming languages.

A subarea of policy-based systems research investigates methods for automation throughout the policy lifecycle. The ultimate goal is to eliminate human-intervention. One example is a machine learning approach to assigning roles in an RBAC system [28].

3. TEMPLATE-BASED POLICY AUTHORING FRAMEWORK

In our previous research, we defined a template-based policy authoring framework and empirically evaluate a policy template authoring prototype [22]. The core idea of the framework is that policy authoring is a complex task which requires user participation and more than one area of expertise. The complexity of the task suggests the need to divide policy authoring into subtasks. It is also necessary to identify the roles and the skills needed to complete the tasks. Our template-based framework defines three subtasks and describes the user who will accomplish each one. Each subtask in the framework is paired with a user interface to help the user complete their tasks. The roles are policy author, template author, and policy element author (see Figure 1).

The *policy author* creates policies from templates, which are structured natural language representations of policy statements. The policy author has knowledge of the policy domain and is assumed to have a high-level understanding of the intended policy. Policy authors are not expected to have in-depth technical knowledge of how the policy will be implemented.

The *template author* creates templates by combining predefined policy elements to form abstractions of the policies the policy author will create. The template author is assumed to be familiar with the policies that may be written but does not need to know the exact policies that will be authored. Template authors also have some knowledge of the technical aspects of the policy domain. Templates offer a syntax guide for the policy author.

The *policy element author* creates policy elements for the

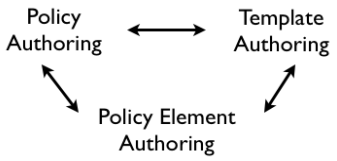

Figure 1: User roles in iterative policy refinement

objects that will be used in policy statements. The policy element author has a thorough understanding of the technical aspects of the policy domain. They must have the knowledge to specify policy elements for the domain specific objects. The objects include users, resources, actions, system variables that can be used in conditions, and security mechanisms. Policy element authors must also specify the relationships between the objects. Policy elements form the vocabulary that will be used for the policy statements.

This framework offers an iterative policy authoring process. The policy author can request new templates or policy elements as needed. Policy authors can also adjust their policies based on the available templates and policy elements. And, the template author can request new policy elements. The policy author's ability to request new templates eliminates the expectation that the template author must predict every policy that will be written. The ability to request new policy elements may help the policy author write higher-quality policies, especially when the policy element author can identify an existing element that satisfies their request.

The following example is an access control template:

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

The following policy is an example from the medical domain. It demonstrates 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 replace the policy elements in a template to form policy statements. An access control policy specifies when someone can perform an action on a resource. Subject, action, and target are required elements for an enforceable access control policy, a condition is optional.

The template-based framework was designed based using results from prior research [24]. Empirical evaluations of SPARCLE show natural language and structured-entry methods are both more usable compared to unguided natural language. To extend SPARCLE to a new domain a robust grammar is required to implement SPARCLE's natural language features. In addition to the grammar, structured lists must be generated for each domain. At this time it is more efficient to implement a method of generating structured lists than it is to implement a robust natural language grammar for each policy domain [24]. Since the usability of the two methods is comparable, and superior to unguided natural language, the template-based policy authoring framework focuses on satisfying the requirement to generate domain specific structured lists.

3.1 Empirical Evaluation of a Template Authoring Prototype

Since the template-based authoring framework calls for a role not specifically considered in previous research – the template author – we conducted a user study specifically to evaluate the usability of a template authoring prototype and to observe how users understand the three roles defined. We argue that policy authoring from natural language guides has been studied previously [10, 32] and that policy element authoring resembles a standard programming task not unique to policy authoring. A detailed description of the template authoring prototype, the user study, and the primary results, those related to template authoring, are reported in another paper [22].

In the template authoring user study, twenty participants completed two template authoring tasks using a template authoring prototype. The participants were technical users with limited policy experience. Each task presented a short scenario and a list of policies. The participant was asked to create templates that a policy author could use to author a set of policies. One task involved a web merchant and writing policies for access to their order database. The other task was for a fictitious social networking site where users could share personal information.

Before beginning the tasks, the participants were given instructions that explained the relationship between the three roles (policy element author, template author, and policy author). Then the participants completed a policy authoring task to give them experience authoring a policy using a template, and demonstrate the difference between the template author and policy author. The participants were asked to think-aloud during the template-authoring tasks, completed post-task questionnaires, and answered debriefing questions. In addition, the study coordinator took notes from an observation room while the participants completed the tasks. This brief description of the user study is included here because some of the data collected provide insight specific to security and privacy policy authoring. The relevant data will be presented in the discussion of new guidelines.

3.2 Policy Authoring Tool Extensibility

The template-based framework was designed to address general policy authoring. By providing an interface for each role that is designed and evaluated with human-centered processes, and defining the expectations of the user in each role, the framework can be extended to any policy domain. Two policy authoring tools discussed above – SPARCLE and Expandable Grids – have been designed with the goal of being extensible to any security and privacy policy domain.

SPARCLE was developed for privacy policies, but can be extended to other policy domains if a natural language (NL) grammar is written and if the corresponding elements of the structured lists are defined. NL parsing of policy statements is a valuable goal but the development of new grammars is more difficult than the specification of templates (structured lists). The difficulty can be attributed to the usability of NL grammar tools which are complex to learn and generally tied to particular language parsers with complex recognition performance characteristics. Some of the difficulty is associated with the desirability of maintaining a structured authoring mechanism to augment the NL approach. For SPARCLE, a new policy type required a new structured entry dialog and a new grammar.

Expandable Grids was also designed to be applied to many security and privacy domains. Empirical results suggest Expandable Grids is a usable policy authoring interface for file access control policies [34]. However, results from a user study of applying Expandable Grids to P3P privacy policies suggest the tool is not more usable than P3P policies expressed in natural language [35]. A description of Expandable Grid's extensibility describes a customization process for tailoring the interface for new policy domains [33]. The customization process has several design steps: designing visual indicators of policy decisions, choosing the appropriate values for the two axes, choosing effective short policy element names, and deciding how to display metadata in a limited space. The difference in performance between file access control and P3P policies suggests applying the customization process is not trivial, though the effectiveness of Expandable Grids depends on a successfully implementing each step. The guidelines for the design of security indicators demonstrate the difficulty of designing effective security indicators [16].

The extensibility of a policy authoring tool should not rely on modifications to the UI design that must be empirically evaluated for each new policy domain. An empirical evaluation of each new domain is not feasible given the continual growth of the number of policy-based systems and the expectations on end-users to manage policies. We believe that the template-based framework provides a better approach by isolating the aspects that are unique to a policy domain to the policy elements. Policy element authors write code specific to the domain using mechanisms that are standard to the framework's implementation. Policy templates provide the appropriate interface between the policy elements and the policies that will be authored.

4. OPTIMIZING FOR SECURITY AND PRIVACY POLICY AUTHORING

Policy-driven systems are used in many domains but here we focus the discussion on security and privacy policy authoring. There is a need to identify design requirements for security and policy authoring tools, but in this early stage of the research it is best to start with guidelines. We begin by presenting existing guidelines for security and privacy policy authoring and discuss how the template-based policy authoring framework meets existing guidelines. We suggest new guidelines based on prior research and present support for the new guidelines using data from our empirical study of a template authoring prototype.

4.1 Guidelines for Usable Security and Privacy Policy Authoring

There are unique usability challenges in authoring security and privacy policies [32]. The following guidelines have been identified for a usable policy authoring system:

- Support object grouping
- Enforce consistent terminology
- Communicate and enforce rule structure
- Make default rules clear
- Prevent rule conflicts

The template-based policy authoring framework satisfies these guidelines, with the exception of making default rules clear and preventing rule conflicts. Policy elements address the first two guidelines. Policy elements are displayed using metadata from their definitions that show the relationships between elements. Policy authors and template authors are able to view the groupings by clicking on policy elements in the GUI for more information or hovering over elements for more detail. These features make object groupings clear [22]. Policy elements also enforce consistent terminology. The policy author is presented with a limited set of values for each policy element in the template and does not have the opportunity to introduce inconsistent terminology. The template authoring prototype was designed so the template author can easily search for and discover the policy elements they may need for a template.

Policy templates enforce and communicate rule structure to the policy author. The template author has the knowledge and experience to know what types of policies will be written. Using this knowledge they create templates that represent the natural language policy statements that should be written. The template authoring interface also stores examples of master templates. Master templates can be modified to create new templates. Authoring policies from templates guarantees the policy author will only be presented with valid rule structures. If the policy author needs to write a policy that cannot be written from the existing templates, they communicate this to the template author. The template author has more experience with the policy domain and can help the policy author refine the rule to fit an existing template, or can create a new template using the policy author's input.

For making the default rule clear, the template author and the policy author must understand whether the system is default-allow or default-deny. In a default-deny system, the policy author must understand that only those actions explicitly allowed will be permitted and that it is unnecessary to write deny policies. For default-deny, the template authoring interface should communicate to the template author that deny templates are unnecessary and may create confusion or introduce conflicts. Additionally, the interface must communicate to the policy author why templates to author deny policies are not available.

Policy conflict detection, prevention, and resolution are active areas of research [27, 13, 2]. A simple method of preventing conflicts is to only allow the policy author to write deny-policies or allow-policies, but to disallow both types of policies within one policy set. This is useful for avoiding conflicts but the restriction could make the set of policy rules unnecessarily large. User interface features that present conflicts and provide conflict resolution guidance should be researched as solutions continue to progress.

In addition to the existing guidelines [32], we propose the following guidelines:

- Support appropriate limitation of expressivity
- Communicate risk and threats
- Provide access to metadata

We demonstrate the importance of each new guideline, provide support from the literature, and present relevant data from our template authoring user study [22].

4.1.1 Support Appropriate Limitation of Expressivity

The guideline to support appropriate limitation of expressivity is related to limiting the set of policies that can be written. This guideline affects usability and, in turn, also affects security. In a template-based policy authoring framework, the responsibility of limiting expressivity falls on the policy element author or the template author. To illustrate when it would be desirable to limit the policies that can be written, imagine the policy author plans to write this policy:

> Order processors can read order fields in the database.

With this template:

> {*Internal Users*} can {*Action*} {*Database Record Fields*} in the database.

In this template, the policy element {*Internal Users*} has the values: database administrators, order processors and customer service. {*Action*} has the values: read, write and edit. And the policy element {*Database Record Fields*} has the values: all fields, order fields and non-private fields.

The policy author could create the policy by choosing the following values for the policy elements:

> {*Internal Users*} = order processors
> {*Action*} = read
> {*Database Record Fields*} = order fields

However, the policy author could also write this policy from the template:

> Order processors can edit all fields in the database.

Even without knowing the details of the policy domain or the threat model, one can assume based on the principle of separation of duties (or separation of privileges) that the second policy is undesirable [36]. Users in the order processors group should be able to view the order. but they should not be able to modify the order. From a security standpoint, only an administrator should have direct access to modifying the database. Or, if it is acceptable for other roles to modify the database, the template should force a policy where auditing or another security mechanism is enabled.

In the template authoring user study described earlier, nine of the twenty participants expressed concerns about template flexibility while completing the template authoring tasks. For example, one participant commented, "But a policy author could say the order processors can access all fields in the database, I want it more specific." Another participant spent a few extra minutes checking their work and looking over their templates for the task. Finally, they expressed their concern, "I think I'm done but I don't like what I have ... they can write what's on the paper but also a lot more than that. Is that right? They can write a lot of policies that probably shouldn't be allowed."

When participants expressed feelings related to this theme they were reminded that the task was to generate policy templates that a policy author could use to write the sample policies on their task sheet. After the participant completed the tasks, their concerns were discussed in the debriefing session. One participant described their perspective in terms of the user roles when they talked about how they chose which policy elements to use, "The template author should limit the choices for the policy author. Depending on the policy author, I would choose different policy elements. Like for a new person, I'd want to limit the choices." It was unexpected that the participants would embrace their role as the template author in this way and consider the policy author as they created the templates. It was most certainly welcome, as their concerns provided additional insights on the relationships between the roles especially for security and privacy policies.

It might seem that specifying the additional relationships between policy element values is an unacceptable amount of work to add to the policy authoring process. Within a template-based policy authoring framework, however, the relationships between policy element values is specified once then is uniformly enforced by the user interface such that the template author is limited by the restrictions and the policy author is as well. If such a limitation is necessary, it is better to specify it one time rather than rely on the policy author to remember to enforce it each time they author new policies or modify existing policies.

The responsibility of specifying these restrictions could be assigned to the policy element author, the template author, or both. If the policy element author must specify the additional relationships their user interface must support such features and make it clear how the changes affect the template author and policy author. It may be best for the policy element author to assume the task because they are familiar with the technical details of the system and have the best understanding of the security mechanisms. However, the template authoring interface chould also provide a way to restrict the values that can be combined to create a policy. As the participants in our study suggested, when the policy elements are put together to form a template there may be combinations that should not be allowed.

We have discussed how building restrictions into the templates may reduce the policy author's ability to author insecure or undesirable policies; we have not addressed how this will affect the policy author's user experience. The end result of this guideline influences the policies the policy author can create from a template. This will change how they are able to interact with some templates. Prior work discusses the user frustration that results when their intended policy cannot be authored, and their willingness to work around policy mechanism when this happens [21]. This must be addressed in the policy author's user interface by making it clear why certain policies cannot be written. When users do not understand the security reasons behind a mechanism they are more likely to subvert it [1].

4.1.2 Communicate Risk and Threats

This guideline represents the need to capture and communicate information about risk and threats. Computer security can be thought of in terms of requirements, security policies, and mechanisms [6]. The goal of computer security is to prevent undesirable events from occurring. The requirements specify the undesirable events and the policies represent decisions for how to satisfy the requirements by stating what is and is not allowed. The mechanisms enforce the policy. In this context, policies can be technical or procedural policies, security mechanisms enforce the technical policies and users in the system are expected to execute the procedural policies.

Presumably the requirements are defined based on some type of risk analysis [7]. A careful quantitative risk analysis

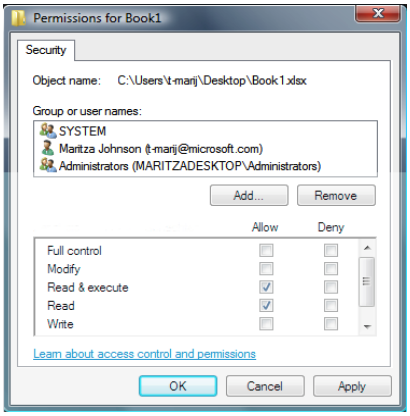

Figure 2: Windows File Permissions Interface

is best [38]. Though, even the results of a rudimentary or qualitative evaluation can benefit the policy author. The policy author may be more capable of authoring high-quality policies if they understand the relevant threats and risks. The inclusion of this additional information could also be used during an audit.

Applying this guideline to the template-based policy authoring framework means communicating risk and threat information between the three user roles. There should be a mechanism such that the policy element author can indicate to the template author that certain elements should be used with special considerations in mind.

For example, if the policy author was writing access control policies for internal and external users, there should be a way for the policy element author to indicate that the templates involving external users should make careful use of security mechanisms when access is allowed. Similarly, the templates could include information to make the policy author aware of the security implications of different policy element values. In most policy authoring interfaces all settings are treated the same by visually presenting the options uniformly. It may benefit users if policy decisions with a higher impact were presented in a different manner.

For a simple situation of where this guideline could be used, consider the Windows file permissions interface as an example(see Figure 2). The security implications of allowing full privileges to a group with many members may be quite different than giving one user read access. However, the user experience of clicking the checkbox to grant the permissions is the same. The optimal way to manage the user interactions in this case is a subject for future research but the first step is noting that the user experience should be different.

This guideline would be useful for firewall policy management interfaces. Correct firewall policy management is crucial since an incorrect policy could deny important traffic that should be let through or allow malicious traffic that should be blocked. In a quantitative evaluation of firewall configurations, 37 rule sets were evaluated to measure configuration quality [40]. The evaluators looked for 12 general configuration errors to assess the quality of the rule set. The configuration errors were chosen based on industry standards at the time. The errors were the kind of errors a general security audit would identify.

One error in this study was allowing connections from the outside to "Any" service to enter the network. This was considered a mistake because there are many high-risk services that should not be used and allowing any incoming connection leaves the network open to known vulnerabilities. About 80% of the rule sets evaluated in the study had this error. To demonstrate how a template-based system could have avoid this misconfiguration. A general template for a firewall policy rule is:

> {*Action*} {*Source IP addresses*} to {*Destination IP addresses*} {*Destination Port*}

The policy element {*Destination Port*} could be defined such that the value "Any" is not an available choice in the list of values if the value for {*Source IP addresses*} is on the outside. This would strongly encourage the policy author to write more restrictive policies for incoming traffic. Another option is to communicate the risk of this policy to the policy author if they select "Any" for the policy element {*Destination Port*}.

In a study of SSL warnings, Sunshine *et al.* found that warnings that described the potential risk of visiting a website increased awareness of risk among study participants [39]. However, policy authoring research has not considered how to incorporate this type of information with the user experience. A similar idea was discussed in a case study of designing a privacy preference interface [14]. Since specifying privacy preferences is a daunting task for the end-user, the work suggested organizing the display of the preferences with the more critical items near the top. This benefits those users who spend a limited amount of time managing their preferences – if they only make a few decisions, at least they've made the important ones.

The policy authoring interface could also provide the policy author with a space to enter comments about the policy. This space could hold information about why the policy was written, what threat it was intended to protect against, or conditions under which the policy can be deleted. Such features may help the policy author manage their policies more effectively.

4.1.3 *Provide Access to Metadata*

It should be straightforward for the policy author to locate information that allows them to understand what terms mean and understand the relationships between policy elements and values.

Policy-based systems tend to use jargon or concepts that are unfamiliar to the policy author. For this reason, all available metadata for policies and policy elements should be easy to access through the user interface. In the template-based framework, the policy elements are associated with metadata to provide as much context information to the policy author as possible and to support the implementation of the features mentioned above. Ontologies have been proposed in the semantic web domain as a way to structure and manage policy metadata [23]. In the semantic web domain, web services are being researched with the end-goal that they will eventually negotiate privacy and security decisions on behalf of the user. Representing the nuances of the user's intended policy is especially important when the system is negotiating policy decisions for the user.

Prior work highlights the necessity of providing clarification for terminology not understood by the user. It's been noted that there is a lack of standardized language even for the most commonly used security concepts [20]. Cranor *et al.* discussed the difficulty of finding the right terminology in their study of designing privacy policies agents [17]. A user study of P3P policy authoring also supports this guideline, participants were confused by the terminology used in the user study tasks, and the user interface did not have room to provide sufficient metadata [35].

The template authoring prototype that was discussed previously was designed such that additional information about a policy template or policy element is accessible by clicking on the element in the user interface. The participants from the template authoring user study found the ability to view the details of the policy elements made it much easier to understand what the policy elements were and how they could be assembled to form templates.

5. CONCLUSIONS AND FUTURE WORK

In this paper we discuss remaining research challenges for security and privacy policy authoring interfaces. We focus our work on the importance of guidelines for usable policy authoring. We propose new guidelines to add to existing guidelines [32], and discuss their benefits. We demonstrate the guidelines a template-based policy authoring framework intrinsically meets and discuss features that can be added to the user experience of policy element authoring and policy template authoring to meet the remaining guidelines. Also, where applicable, we present empirical evidence from a user study on template authoring that indicates the user demand, and utility, of these features [22]. Throughout the paper we outline the direction of the template-based policy authoring framework and discuss the progress to date.

We suggest the following new guidelines for security and privacy policy authoring:

- Support appropriate limitation of expressivity
- Communicate risk and threats
- Provide access to metadata

The next step for this research is to design and evaluate a policy element authoring prototype that satisfies the new set of guidelines. It is also necessary to design and evaluate a policy authoring prototype that follows the new guidelines. Empirical evaluation will reveal whether the guidelines are sufficient for achieving a policy authoring tool that is more usable than existing tools.

When the design and evaluation of the template-based framework's stages are complete the extensibility of the framework will depend on the policy element author's ability to capture the policy domain's unique vocabulary. The feasibility of this goal needs to be tested with empirical evaluations with policy experts from a range of domains.

It is important to continue researching better tools and mechanisms for security and privacy policy-authoring, and to establish guidelines for better interfaces as we learn more. To achieve security goals, it is crucial that policy authors are able to author high-quality policies and to ensure the specified policy matches their intended policy.

6. ACKNOWLEDGMENTS

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.

7. REFERENCES

[1] A. Adams and M. A. Sasse. Users are not the enemy. *Commun. ACM*, 42(12):40–46, 1999.

[2] D. Agrawal, S. Calo, J. Giles, K.-W. Lee, and D. Verma. Policy management for networked systems and applications. *9th IFIP/IEEE International Symposium on Integrated Network Management, IM 2005.*, pages 455–468, 2005.

[3] D. Balfanz. Usable access control for the world wide web. In *ACSAC 19th Annual Computer Security Applications Conference*, pages 406–415, 2003.

[4] Y. Bartal, A. Mayer, K. Nissim, and A. Wool. Firmato: A novel firewall management toolkit. *ACM Trans. Comput. Syst.*, 22(4):381–420, 2004.

[5] 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 *CHI '08: Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 543–552, NY, NY, USA, 2008. ACM.

[6] M. Bishop. What is computer security? *IEEE Security and Privacy*, 1:67–69, 2003.

[7] B. Blakley, E. McDermott, and D. Geer. Information security is information risk management. In *Proceedings of the 2001 Workshop on New Security Paradigms*, pages 97–104. ACM, 2001.

[8] M. Blaze, J. Feigenbaum, J. Ioannidis, and A. Keromytis. The KeyNote trust-management system version 2. `http://tools.ietf.org/rfc/rfc2704.txt`, September 1999.

[9] C. Brodie, D. George, C. Karat, J. Karat, J. Lobo, M. Beigi, X. Wang, S. Calo, D. Verma, A. Schaeffer-Filho, et al. The Coalition Policy Management Portal for Policy Authoring, Verification, and Deployment. In *IEEE Workshop on Policies for Distributed Systems and Networks*, pages 247–249, 2008.

[10] C. Brodie, C.-M. Karat, J. Karat, and J. Feng. Usable security and privacy: A case study of developing privacy management tools. In *SOUPS '05: Proceedings of the 2005 symposium on usable privacy and security*, pages 35 – 43, 2005.

[11] X. Cao and L. Iverson. Intentional access management: making access control usable for end-users. In *SOUPS '06: Proceedings of the second symposium on Usable privacy and security*, pages 20–31, NY, NY, USA, 2006. ACM.

[12] W. Cheswick, S. Bellovin, and A. Rubin. *Firewalls and Internet security: repelling the wily hacker.*

Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 2003.

[13] J. Chomicki, J. Lobo, and S. Naqvi. A logic programming approach to conflict resolution in policy management. In *Principles of Knowledge Representation and Reasoning: Proceedings of the 7th International Conference (KR2000)*, 2000.

[14] L. Cranor. Designing a privacy preference specification interface: A case study. In *CHI 2003 Workshop on Human-Computer Interaction and Security Systems*, 2003.

[15] L. Cranor, M. Langheinrich, M. Marchiori, M. Presler-Marshall, and J. Reagle. The Platform for Privacy Preferences 1.0 (P3P 1.0) specification. W3C Recommendation, April 2002.

[16] L. F. Cranor. What do they indicate?: evaluating security and privacy indicators. *interactions*, 13(3):45–47, 2006.

[17] L. F. Cranor, P. Guduru, and M. Arjula. User interfaces for privacy agents. *ACM Trans. Comput.-Hum. Interact.*, 13(2):135–178, 2006.

[18] N. Damianou, A. K. Bandara, M. S. Sloman, and E. C. Lupu. A survey of policy specification approaches. Technical report, Department of Computing, Imperial College, 2002.

[19] N. Damianou, N. Dulay, E. Lupu, and M. Sloman. The Ponder policy specification language. In *Lecture Notes in Computer Science*, pages 18–38. Springer-Verlag, 2001.

[20] S. L. Garfinkel. *Design Principles and Patterns for Computer Systems That Are Simultaneously Secure and Usable*. PhD thesis, MIT, 2005.

[21] M. Johnson, S. M. Bellovin, R. W. Reeder, and S. Schechter. Laissez-faire file sharing: access control designed for individuals at the endpoints. In *NSPW '09: Proceedings of the New Security Paradigms Workshop*, pages 1 – 10, September 2009.

[22] M. Johnson, J. Karat, C.-M. Karat, and K. Grueneberg. Usable policy template authoring for iterative policy refinement. In *submission to Annual Conference of ITA, ACITA*, 2010.

[23] L. Kagal, M. Paolucci, N. Srinivasan, G. Denker, T. Finin, and K. Sycara. Authorization and privacy for semantic web services. *IEEE Intelligent Systems*, 19(4):50–56, July/August 2004.

[24] J. Karat, C.-M. Karat, C. Brodie, and J. Feng. Privacy in information technology: Designing to enable privacy policy management in organizations. In *International Journal of Human-Computer Studies*, volume 63, pages 153–174. Elsevier, 2005.

[25] H. R. Lipford, A. Besmer, and J. Watson. Understanding privacy settings in Facebook with an audience view. In *UPSEC'08: Proceedings of the 1st Conference on Usability, Psychology, and Security*, pages 1–8, Berkeley, CA, USA, 2008.

[26] R. A. Maxion and R. W. Reeder. Improving user-interface dependability through mitigation of human error. *International Journal of Human-Computer Studies*, 63(1-2):25 – 50, 2005.

[27] J. Moffett and M. Sloman. Policy conflict analysis in distributed system management. *Journal of Organizational Computing*, 4(1):1–22, 1994.

[28] I. Molloy, H. Chen, T. Li, Q. Wang, N. Li, E. Bertino, S. Calo, and J. Lobo. Mining roles with semantic meanings. In *SACMAT '08: Proceedings of the 13th ACM symposium on access control models and technologies*, pages 21–30. ACM, 2008.

[29] D. A. Norman. *The Design of Everyday Things*. New York, DoubleDay, 1988.

[30] OASIS. *eXtendible Access Control Markup Language Committee Specification 2.0*. Security services technical committee, 2005.

[31] Organization for Economic Co-operation and Development. *OECD Guidelines on the Protection of Privacy and Transborder Flow of Personal Data*. Paris, France, 1980.

[32] R. Reeder, C.-M. Karat, J. Karat, and C. Brodie. Usability challenges in security and privacy policy-authoring interfaces. In *Human-Computer Interaction – INTERACT 2007*, pages 141–155. Springer Berlin / Heidelberg, 2007.

[33] R. W. Reeder. *Expandable Grids: A user interface visualization technique and a policy semantics to support fast, accurate security and privacy policy authoring*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, July 2008.

[34] 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 *CHI '08: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 1473–1482, NY, NY, USA, 2008. ACM.

[35] R. W. Reeder, P. G. Kelley, A. M. McDonald, and L. F. Cranor. A user study of the expandable grid applied to p3p privacy policy visualization. In *WPES '08: Proceedings of the 7th ACM workshop on Privacy in the electronic society*, pages 45–54, New York, NY, USA, 2008. ACM.

[36] J. H. Saltzer and M. D. Schroeder. The protection of information in computer systems. *Proceedings of the IEEE*, 63(9):1278–1308, Sept. 1975.

[37] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman. Role-based access control models. *Computer*, 29(2):38–47, 1996.

[38] S. E. Schechter. *Computer Security Strength & Risk: A Quantitative Approach*. PhD thesis, Harvard University, Cambridge, Massachusetts, May 2004.

[39] J. Sunshine, S. Egelman, H. Almuhimedi, N. Atri, and L. F. Cranor. Crying wolf: An empirical study of SSL warning effectiveness. In *18th USENIX Security Symposium*, 2009.

[40] A. Wool. A quantitative study of firewall configuration errors. *IEEE Computer*, 37(6):62–67, 2004.

[41] M. E. Zurko, R. T. Simon, and T. Sanfilippo. A user-centered, modular authorization service built on an RBAC foundation. *Proceedings of the 1999 IEEE Symposium on Security and Privacy*, pages 57–71, 1999.