An Authoring Framework for Security Policies: A Use-Case within the Healthcare Domain*

Thomas Trojer[1], Basel Katt[1], Florian Wozak[2], and Thomas Schabetsberger[2]

[1] Research Group Quality Engineering, University of Innsbruck, Austria
{thomas.trojer,basel.katt}@uibk.ac.at
[2] ITH-icoserve GmbH, Innsbruck, Austria
{florian.wozak,thomas.schabetsberger}@ith-icoserve.com

Abstract. Traditionally, the definition and the maintenance of security and access control policies has been the exclusive task of system administrators or security officers. In modern distributed and heterogeneous systems, there exist the need to allow different stakeholders to create and edit their security and access control preferences. In order to solve this problem two main challenges need to be met. First, authoring tools with different user interfaces should be designed and adapted to meet domain background and the degree of expertise of each stakeholder. For example, policy authoring tools for a patient or a doctor should be user friendly and not contain any technical details, while those for a security administrators can be more sophisticated, containing more details. Second, conflicts that can arise among security policies defined by different stakeholders must be considered by these authoring tools on runtime. Furthermore, warnings and assisting messages must be provided to help defining correct policies and to avoid potential security risks. Towards meeting these challenges, we propose an authoring framework for security policies. This framework enables building authoring tools that take into consideration the views of different stakeholders.

Keywords: Security policy, EHR, Policy authoring, Usability, Model-driven engineering.

1 Introduction

In recent years' conducted research and applications developed to advance the field of patient healthdata management, the trend is clearly towards data maintenance in an electronic way. Medical institutions, like hospitals or private practitioners extend their (or establish) IT infrastructures to cope with proposed architectures and governmental regulations regarding the storage and distribution of electronic health records (EHRs). Distribution of patient's medical data or the provision of shared access to it, is commonly supported and understood to lower healthcare costs and increase efficiency of individual medical

[*] This work was partially supported by the Austrian Federal Ministry of Economy as part of the Laura-Bassi – Living Models for Open Systems – project FFG 822740/QE LaB, see http://lab.q-e.at/

M. Szomszor and P. Kostkova (Eds.): E-Health 2010, LNICST 69, pp. 1–9, 2011.

treatments [1]. An example of an EHR implementations is *sense* infrastructure developed by ITH-icoserve. *ITH-icoserve technology for health-care GmbH*, as a subsidiary company of a regional hospital holding company, the *Tiroler Landeskrankenanstalten GmbH* (TILAK) and *Siemens AG*, has developed systems supporting cooperative care within Austria. The electronic health records in this scenario are aligned to the specifications of an Austrian governmental company, *ELGA GmbH* (Elektronische lebensbegleitende Gesundheitsakte, ”*electronic lifelong health record*”) which proposes the interlinked, but distributed storage of electronically acquired healthdata of patients.

In our previous work [6,7,8] we dealt with security mechanisms on the infrastructure level. In this work, on a more conceptual level, we identify sources for patient and clinician authentication as a prerequisite to further allow or deny requests or to audit (im)proper behavior. Access control policies are put in place to cope with governmental regulation on who is allowed to access what kind of health record under which circumstances. An example of such a policy in Austria is that no physician is allowed to access a record she/he created about a patient if the treatment was given earlier than 28 days in the past. This strict rule is weakened under certain conditions, e.g., if it is the intention of the patient to make her/his medical data available for a longer period or in case of a required emergency access. Our project goal is to design a framework, which provides *different stakeholders* with *security authoring tools* with a high degree of *usability*. Further these authoring tools are designed to *analyse* security measures and to generate *enforceable security artifacts*, later deployed within the security infrastructure of the system. Analysis thereby aims at ensuring consistency by resolving conflicts among active policies defined by different stakeholders and enhancing the quality and correctness of defined policies by warning users about potential security risks when new policies are created.

Access to medical data is important to many stakeholders within the healthcare domain, but heavily raises the potential of having privacy and security at risk. The type of security we focus on is about patient-controlled dynamic access control. *Patient-controlled* and *dynamic*, because of patients expressing their personal privacy preferences about their identifying medical records at any time and at their free will. Patient-controlled declaration of access control regulations is a useful tool to support a patient's desire on self-determination about the usage of her/his private data. Besides that it also presents itself to be a challenging task with open questions in various ways. We have to deal with potential data risks, since non-IT experts define privacy and security measures. This in turn leads to a discussion to which extend patients shall be allowed to determine privacy aspects regarding their healthcare data. Furthermore, authoring tools have to offer functionality which guides and helps patients who are inexperienced and face difficulties when creating their privacy rules. Thus Usability of the authoring platform has to be considered. This can be partially done by the development team, but mainly, usability evaluations have to be conducted by empirical studies including patients, physicians, security experts and IT experts. Finally, general information and clarification on the purpose of patient-controlled authoring of access control policies has to be transmitted to the general public. This is especially needed to gather the required public acceptance in using such authoring tools and the trust in the method itself to be a chance for everybody to make use of the personal right of informational self-determination.

The framework is of a generic type, but we will evaluate its feasibility within a usecase scenario taken from the healthcare domain. Additional knowledge is therefore provided by our collaborating industrial partner, ITH-icoserve GmbH.

2 Related Work

Different security measures have been proposed to protect electronic healthcare environments. As of our focus, we are especially interested in authorization to secure access to electronic healthcare records of patients. The healthcare domain has certainly specific requirements to access control. Patients, for example, should be allowed to declare access control regarding their identifying health data [1]. In [11] requirements and an initial model for patient-controlled access control using *Role-based Access Control* (RBAC) [12] is presented. We build upon that work but further extend the access control model to cope with a variety of other requirements in our context.

Additionally the work in [4] discusses access control for medical records maintained by electronic information systems. The authors proposed, similar to our work, several (abstract) models which define concepts of security related to the healthcare domain.

Furthermore, delegation of access rights plays an important role to e.g., allow medical institutions and research laboratories to conduct studies in medical sciences. Of course, such transitive access rights have to be purpose and obligation-bound, as well as they have to be accompanied by protection of corresponding data through pre-enforced data privacy measures. E.g., [13] describes a method how to anonymize and share data from different sources to conduct mining and analysis. This might be just one of many methods which can be employed together with delegation mechanisms.

Regarding usability issue in security policy authoring tools, there has already been a body of work published. In [2] the authors propose the automatic generation of security-aware event-driven graphical user interfaces, by mapping RBAC entities to events triggered in order to access application resources. Similar to this approach we employ an extended policy model and deal with the authoring of corresponding policy artifacts, as it has been proposed in [5]. Our approach on the other hand covers a lot more than just the policy authoring tools themselves, as we design a framework which incorporates arbitrary types of security policies and enterprise domains. The final outcome is specifically tailored security policy authoring tools regarding the domain they are designed for. Finally, usability plays an important role in the acceptance and the success of security policy authoring tools. E.g., in [10], the authors put a focus on usability in the context of authorization methods. They defined several usability challenges with which policy authoring tools have to cope. Further a user study was conducted to evaluate their implementation of a policy authoring tool, namely the SPARCLE policy workbench.

3 Modeling Approach

In order to achieve a usable method for a patient to modify her/his access control preferences regarding personal medical data, we identify a variety of issues that must be taking into consideration.

First and foremost security requirements have to be identified. These are then reflected by appropriate security models (in our context e.g., an access control policy model). An access control policy model is used to have a domain and platform independent representation to express authorization concepts. Such a model only covers generic concepts how statements of a concrete policy are defined and therefore represents a policy specification. We can use such a specification to evaluate conformance of any concrete policy we maintain. A policy model contains security concepts independent of the domain, but can be mapped to domain entities with a certain properties. In case of RBAC we would e.g., infer properties like "*isRole*", "*isResource*" or "*isAccessMode*" for domain entities like "*pharmacist*", "*prescription*" or "*read*" respectively.

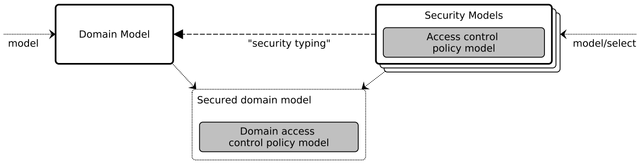

Fig. 1. Domain and security model according to a security requirements analysis

Fig. 1 shows how we incorporate the policy model with the actual domain. Note that we use the term *security model* to emphasize that all types of security-related measures are potentially modeled there. In this paper we will only focus on access control policies. In order to use the access control policy model within a specific domain, we have to first identify all concepts which occur in the real domain environment. This is entitled as the *domain model* and is also depicted in the referred figure.

The mapping of model elements in the context of security to domain model elements introduces a method we call "*security typing*". Security typing simply relates security model elements as being properties of the domain model entities.

4 Entity Models

In this paper we deal with two entity models, the *domain model* and the security model (*domain access control policy model*). In this section we want to elaborate our viewpoint on the importance of model-driven engineering to design and develop access control authoring tools in the healthcare scenario. We do this by describing the two entity models we introduced briefly within Sec. 3.

4.1 Domain Model

The *domain model* is the core resource for which certain authoring aspects are made available. The domain model describes, as mentioned earlier, all concepts which occur in the real environment, whereas *domain model instances* describe the actual entities which conform to the domain model. The healthcare domain, e.g., can be described by a set of healthcare related roles, medical documents (EHRs) of different types, healthcare institutions and care activities. Furthermore, relationships between the various constituents of this domain can be identified.

The general domain model, which describes the real domain environment (e.g., healthcare), can be connected with other models tackling a specific aspects in that domain (e.g., security), which we call a *aspect model.* By mapping both models, we can develop or generate executable applications dealing with certain aspects of the domain.

4.2 Access Control Aspect Model

Fig. 1 depicts the modeling process performed by domain experts, which step-by-step considers the (*i*) *analysis* of the domain model in order to gather knowledge about certain aspects and requirements; the (*ii*) *selection* of security properties which are appropriate to the targeted security requirements; finally, the (*iii*) *mapping* of the domain model and the aspect model in order to highlight how arbitrary domain model elements have to be represented during an authoring process.

The *access control policy model* is the aspect model we focus on in this work. It covers role/subject entities for which access to resource entities shall be (dis)allowed to a set of conditions. Conditions are also expressible within the model and consist of context attributes which have to be verifiable.

During our research we identified types of potential access control rules. The policy model has to be expressive enough to cover all of the following concepts:

- Permissions based on *hierarchical roles* allowed or denied to access resources
- Permissions based on the identity of *individuals*
- (Restricted) *delegation* of permission
- Permissions targeting *resources*, either by their *type* or *identifiers*
- Permissions bound to *conditions*, possibly of the following types or any combination of them: temporal, location-based, (medical) session-specific, purpose-based and obligation-bound.

Hierarchical roles [12] are used to express institutional roles and how they relate to the domain. E.g., clinicians are part of a healthcare institution (*parent role*) and consists of surgeons, internists, anesthetists and others (*child roles*).

A role used within a permission assignment implies that all child roles are assigned with that permission as well. This is since their parent role is used to generically cover the group of roles with certain attributes in common.

Permissions for subjects or individuals are meant to be assigned to individual persons, rather than roles. Such assignments are especially practical e.g., to declare a specific doctor as being someone's family practitioner. Such a practitioner may receive extended permissions compared to permissions she/he would get granted because of the general domain role that is assigned to her/him.

Delegation of permissions is useful if, for a specifically stated purpose, an actor needs access to resources she/he normally doesn't hold. A case is if the maintenance of medical records and access rules to them is not feasible e.g., due to a certain disability of the identified individual. Delegates thereby may include e.g., family members or nursing staff.

Resources, which are the target of permissions may be referenced in two different ways: Each single EHR is labelled with a type. Permissions can therefore target all EHRs of a common type. Such types include e.g., prescriptions, medical treatment reports or discharge letters. Further specific EHRs can be the target of a permission by explicitly referring to their unique document identifiers.

Various types of *conditions* increase the expressiveness of permissions. We identify four different types of conditions: *Temporal* to express time and date constraints according to access requests. *Location-based* may declare permitted access only at certain locations. Such a condition can e.g. express, "only at the employing healthcare institution" in order to prevent physicians to access medical records from home. Another example of location-based conditioned access is to establish the "four-eye principle", in which a physician is only allowed to access patient's medical data if the corresponding patient is attending in a medical session. *(Medical) session-specific*, which declares EHRs to be (un)available to the practitioner performing in a medical treatment session. This condition may internally represented by temporal and location-based conditions covering the session's appointed date and place, respectively. *Purpose-based* conditions are set in order to be able to verify if the purpose of a usage of EHRs stated by the patient matches the intended usage by the practitioner or researcher. Different kinds of purposes may be related within a hierarchy [12]. A chosen purpose of usage for an EHR represents the "maximal" allowed purpose (i.e. regarding the severity of e.g., risked privacy) the data is intended to be accessed, processed or distributed. *Obligation-bound* access declares certain actions to be fulfilled prior to granted access. E.g., patient notification, if demanded by the patient, or request for permission by a physician can be established that way.

5 Policy Authoring Environment

Continuing from the *secured domain model*, where we relate security entities to domain entities, we declare editing functionality through user interface (UI) input controls. This method is what we call "*interface typing*" and bases on domain entities which have been previously associated with security properties

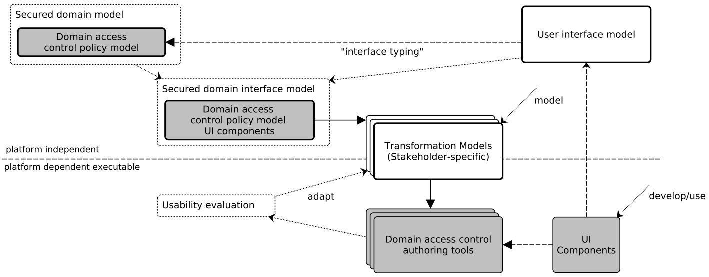
Secured domain model
Domain access control policy model
"interface typing"
User interface model
Secured domain interface model
Domain access control policy model UI components
model
Transformation Models (Stakeholder-specific)
platform independent
platform dependent executable
Usability evaluation
adapt
develop/use
Domain access control authoring tools
UI Components

Fig. 2. Security domain with user interface annotations. Generation of authoring tools according to transformation models.

through security typing. Fig. 2 shows this by assigning UI input controls to the *domain access control policy model.* Such input controls are defined within the *user interface model* and realized by a concrete UI development kit (see *UI components* as an instance of the *user interface model*). Typically textual input fields/areas, item and option selections as well as command buttons are provided. Besides that, a general style and textual output elements are provided to increase the structural quality, readability and usability of the UI.

According to the *secured domain interface model* we have now gathered security-relevant domain entities with additional information on how input data can be acquired. Still, in order to complete a transformation process from the extended domain model to an authoring tool for security properties of the domain, a *transformation model* [3] is put in place. The following tasks are carried out by using this transformation model: (*i*) UI layout and UI component grouping, (*ii*) interface realization to auxiliary system components, and (*iii*) stakeholder-centric views.

UI component layout is performed for both, input controls and output components according to the policy constructs and their semantical meaning. Further a usable UI has to be generated by taking into account certain layout rules and coded best practices. E.g., semantical meanings like "*subject* is allowed to *perform action* on *target EHR*, if *condition* is satisfied and private health data is at most used for the purpose of *purpose*." can be inferred from our access control policy model. UI components, i.e. input controls for domain entity placeholders (emphasized by italic font in the access control statement) as well as text labels to complement a natural language sentence, have to be logically grouped.

The transformation model, which is defined closely related to the selected security model, is responsible for realizing interfaces to perform communication with auxiliary system components. E.g., queries to a policy repository and the provisioning of enforceable rules to a security engine have to be implemented. Further a source for retrieving domain model instances has to be defined there.

The transformation process is performed with regards to the stakeholder, which leads to customized authoring environments with full or limited functionality available to certain types of users. Additionally, each stakeholder has only access to a restricted set of the overall domain entity instances usable for permission creation. According to this subset, specific queries are generated to fetch all accessible domain entity instances.

Finally it has to be easily possible to integrate the generated authoring applications to existing healthcare portal applications, which are already in place in many healthcare institutions.

6 Conclusion

This paper presents an overview of a conceptual framework to author certain domain aspects. Tackled domain aspects are defined by aspect models. A security model, e.g., is the aspect model of our healthcare domain model, while a user interface model is the aspect model of the secured domain model. Transformation models are used to layout the generated authoring applications according to usability best practices.

Our future research will highlight each step performed within this framework. A realistic domain model is currently discussed with our partner from the healthcare industry. Once the domain model is established we will decide on a security model which fits the access control requirements of the given domain. Our focus will be entirely put on access control in EHR-maintaining infrastructures, which is performed by patients identified by such health records. Further we will conduct real usability studies with different domain stakeholders in order to establish best practices in designing patient-controlled authoring tools regarding access control of EHRs.

References

1. IBM Austria, Feasibility study for implementing the electronic health record (ELGA) in the Austrian health system, IBM (November 2006)
2. Basin, D., Clavel, M., Egea, M., Schläpfer, M.: Automatic generation of Smart, Security-aware GUI Models. In: Massacci, F., Wallach, D., Zannone, N. (eds.) ESSoS 2010. LNCS, vol. 5965, pp. 201–217. Springer, Heidelberg (2010)
3. Bézivin, J., Büttner, F., Gogolla, M., Jouault, F., Kurtev, I., Lindow, A.: Model transformations? Transformation models! In: Wang, J., Whittle, J., Harel, D., Reggio, G. (eds.) MoDELS 2006. LNCS, vol. 4199, pp. 440–453. Springer, Heidelberg (2006)
4. Blobel, B.: Authorisation and access control for electronic health record systems. International Journal of Medical Informatics 73(3) (2004)
5. Karat, C., Karat, J., Brodie, C., Feng, J.: Evaluating interfaces for privacy policy rule authoring. In: SIGCHI 2006. ACM, New York (2006)
6. Katt, B., Breu, R., Hafner, M., Schabetsberger, T., Mair, R., Wozak, F.: Privacy and Access Control for IHE-Based Systems. In: Weerasinghe, D. (ed.) eHealth 2008. LNICST, vol. 1, pp. 145–153. Springer, Heidelberg (2009)
7. Katt, B., Trojer, T., Breu, R., Schabetsberger, T., Wozak, F.: Meeting EHR Security Requirements: SeaaS approach. In: EFMI STC 2010, Reykjavik, Iceland (June 2010)

8. Katt, B., Trojer, T., Breu, R., Schabetsberger, T., Wozak, F.: Meeting EHR Security Requirements: Athentication as a Security Service. In: Perspegktive Workshop, GMDS 2010, Mannheim, Germany (September 2010)
9. Lodderstedt, T., Basin, D., Doser, J.: SecureUML: A UML-based modeling language for model-driven security. In: Jézéquel, J.-M., Hussmann, H., Cook, S. (eds.) UML 2002. LNCS, vol. 2460, p. 426. Springer, Heidelberg (2002)
10. Reeder, R.W., Karat, C., Karat, J., Brodie, C.: Usability challenges in security and privacy policy-authoring interfaces. In: Baranauskas, C., Abascal, J., Barbosa, S.D.J. (eds.) INTERACT 2007. LNCS, vol. 4663, pp. 141–155. Springer, Heidelberg (2007)
11. Røstad, L.: An initial model and a discussion of access control in patient controlled health records. In: ARES 2008. IEEE Computer Society, Washington, DC, USA (2008)
12. Sandhu, R., Coyne, E., Feinstein, H., Youman, C.: Role-based access control models. Computer 29 (1996)
13. Trojer, T., Fung, B., Hung, P.: Service-oriented architecture for privacy-preserving data mashup. In: IEEE International Conference on Web Services (2009)