# Adaptable Access Control Policies for Medical Information Systems

Tine Verhanneman, Liesbeth Jaco, Bart De Win,
Frank Piessens, and Wouter Joosen

DistriNet, Dept. Computer Science
K.U.Leuven, Celestijnenlaan 200A
3001 Leuven, Belgium
{tine,liesbeth,bartd,frank,wouter}@cs.kuleuven.ac.be

**Abstract.** IT enforced access control policies in medical information systems have to be fine-grained and dynamic. We justify this observation on the basis of legislation and on the basis of the evolution within the healthcare domain. Consequently, a reconfigurable or at least adaptable implementation of access control facilities has become extremely important. For this purpose, current technology provides insufficient support. We highlight a basic solution to address shortcomings by using interception techniques. In addition, we identify further research that is required to address the challenges of dynamic and fine-grained access control in the long run.

## 1 Introduction

The healthcare industry spends increasingly more resources on IT, driven by both the need to manage costs and by the changing structure of healthcare organizations. As a side effect, the privacy and security of health care information is a growing concern([2]). Because of these demands, access control in medical IS is a tremendous challenge ([6]). Therefore, the requirements for these systems are characterized in this paper. Moreover, we argue that access control policies are not (and will never be) static and underline the importance of fine-grained IT enforced policies. In an extended version of this paper ([16]) some example policies are described in more detail.

Access control dynamicity is driven by two forces: legislation on the one hand and system changes on the other hand.

There are rules (ethical and legislative rules) that prescribe how medical data should be handled. First of all, since these laws can change, access control policies are by nature not static. Second, the interpretation of the legislation varies. As will be shown in section 2, the applicable law in the USA states that to meet some of the requirements a "reasonable effort" should be done. The concrete interpretation of this term depends on the context, like e.g. the size of the organization and on case law.

Another cause of evolving access control policies are system changes. When for instance not only hospital personnel but also patients and relatives can login

J.-B. Stefani, I. Demeure, and D. Hagimont (Eds.): DAIS 2003, LNCS 2893, pp. 133–140, 2003.

onto the system to access their own clinical information, which can be viewed as an extension in functionality, measures need to be taken. Another example of a system change are adaptations in the IT access control enforcement itself. These are the changes that will be focussed on in this paper, and that will lead us to the conclusion that IT enforced access control policies needs to be *fine grained* and also dependent on information, such as context or application state, as well as subject, object and operation, that are involved in an invocation on the system.

As will be demonstrated, it is essential that the security component in medical information systems includes reconfigurable and/or adaptable access control technology. We now define these terms more precisely. Security policies are said to be *reconfigurable* if the deployer of the application is able to set the security measures without having to dig into the source-code written by the programmer. Access control is set through the interpretation of a configuration file, written in a "non-programming language", such as XML for example. Ideally, the format and language of this configuration file are not too complicated, so that non-programmers are able to draw it up. In the literature, this is often referred to as *declarative security*, which is often opposed to programmatic security, by which is meant that policies are hard-coded in the application.

*Adaptability* is a weaker requirement than reconfigurability. A system that is highly adaptable requires little effort to be transformed into a system that satisfies new requirements. Hereto all the concerns are to be expressed somehow separately in a modular structure, which doesn't necessarily imply that the access control policy can be changed at deployment time or at run time.

Supporting reconfigurability or adaptability often limits the access control policies that can be implemented, as expressive power is limited, either because the configuration language is limited, or because the access control module does not have access to all information that is required to enable detailed decision making. Expressiveness must be treated with caution, since the more expressive the language the more complex the enforcement mechanism will be.

Having defined these terms, we analyze the requirements for an access control module in a medical IS, first, in section 2, on the basis of the legal and regulatory framework for privacy and security of medical data in the EU and the US and secondly, in section 3, in the perspective of evolution of the healthcare organisation. In section 4, we discuss the impact of our observations when building applications using state-of-the-art platforms such as J2EE and .NET, whoms support for declarative access control proves to be insufficient. The road for future research is laid out in section 5 and a conclusion is formulated in section 6.

## 2 Access Control Policies in Healthcare

All existing threats against privacy and security of health information are contrary to a concern for patient rights and the well-functioning of health care. In respect to confidentiality this concern is formulated in the Hippocrates oath.

Nowadays, the rules for the use of medical information are extended with the right to dispose of data with guaranteed integrity and availability. Laws are constituted, describing the rights and duties imposed when processing medical data and defining the potential sanctions applied to misuse. On this basis, healthcare institutions formulate policies, containing both organizational and technical security measures. The legislation actually provides for two kinds of rights and duties.

First, the law prescribes the circumstances for medical data to be collected, stored and used, and the authorization rules to access the data. This is input for the access control policy that a healthcare organization should manage.

Second, the law also sets some standards on how well the policy should be *enforced* and it is this kind of legislation that is especially important from the point of view of IT enforcement. Considering the protection of health information in the EU, the Data Protection Law ([8]) augmented with the Recommendation on the Protection of Medical Data ([9]) emphasize on the *appropriateness* according to the faced risks and the *periodical review* of the measures to protect personal data.

The American legislation will be discussed in more detail. The specific law concerning the protection of individually identifiable health information is the Health Insurance Portability and Accountability Act of 1996 (HIPAA). HIPAA is considered as the most significant healthcare legislation passed in years and includes rules on electronic transactions, national identifiers, patient privacy and data security. It obliges healthcare organizations to use information and communication technology to increase efficiency, but it also addresses the problems of deploying these technologies. All healthcare organizations that maintain or transmit electronic health information have to comply, and there are severe civil and criminal penalties for those that do not.

In the context of this paper, two rules of the comprehensive HIPAA regulation are important, namely the Privacy Rule ([12]) and Security Rule ([13]). The former sets forth what uses and disclosures are authorized or required and what rights patients have with respect to their health information while the latter specifies what implementation is obligatory for enforcement of this policy or what reasonable efforts should be done. It describes the necessity for standards at all stages of transmission and storage of electronic health care information to ensure integrity and confidentiality of the records at all phases of the process, before, during and after electronic transmission as well as physical and technical safeguards to protect the confidentiality, integrity and availability of electronic protected health information. Features such as context-based, role-based and user-based access control, were explicitly mentioned in a draft version, but have been deleted and replaced by the requirement that *appropriate* access control should be provided, like in the European legislation. In the following section, it will be illustrated how this requirement leads to the need of IT-enforced dynamic access control policies.

## 3 IT Enforced Policies Are Dynamic

A security policy can be enforced in many ways: through organizational and through technical measures. In healthcare, there is an old tradition of trusting the care provider. Because of the increased specialization of care providers, and the increased complexity of care procedures, the size of the team of care providers that deals with one patient grows. Teams of ten to fifty are common. Obviously the purely trust-based model does not work. Besides, the increased use of IT makes technical measures to enforce the security policy unavoidable. Many hospitals have already reached the maturity-point where hospital wards are distributed in different, separated buildings. Because each ward has its own administration, data is no longer centralized and communication networks outside the physical boundaries are used to share information. More evolved healthcare organizations offer remote access, which allows doctors or patients to access clinical information from off-site locations.

Many people have increasing (potential) access to personal clinical information of a large number of patients. Therefore IT enforcement becomes essential, organizations rely less on trust ([1]). In this context the term *regular policy* refers to the policy a healthcare organization wishes to impose, independent of the enforcement mechanism. *IT enforced policy* refers to a policy that is actually enforced by IT-based technical measures.

In this paper, the focus is on the IT enforced access control policy. It is unreasonable to suppose that these IT enforced access control policies will remain the same over long periods of time, for the following reasons.

1. *Changes in the regular policy* can be caused by changes in legislation, changes in the interpretation of legislation, or because the hospital management decides that a stricter policy will lead to a competitive advantage as patient awareness about privacy increases.
2. *Appropriate security is necessarily dynamic.* As discussed in section 2, the law states that enforcement of the access control policy should be reviewed regularly.
3. There is a shift of the trust based model towards *more IT enforcement.*
4. *IT enforced policy evolves with system changes.* As system functionality increases, the security rules need to be enforced in more and more situations.

## 4 Support for Flexible Access Control Policies

### 4.1 The Need for Expressiveness

Access control rules describe conditions for subjects to access resources. It is obviously not feasible to write a separate rule for any object or subject in the system. Hence, a good policy language should support the grouping of objects with similar access control requirements and the grouping of subjects with similar rights.

Grouping of objects is typically achieved through *policy domains*. Each of the objects in the system belongs to one or maybe several domains. This does not always depend on the class of the object alone, but also on instance properties, the relations the object is involved in, the location where the object instance is deployed, ... A powerful policy language should support fine-grained description of policy domains as well as short, generalizing definitions.

Besides the grouping of objects, some support of grouping *subjects* will be required. An obvious example of this kind of grouping is the notion of a role. A more advanced example is implicit grouping as described in [10], where roles are assigned (dynamically) on the basis of properties of the subject.

A subject may only have limited access to an object. In object-oriented systems, there is a tendency to consider a method as the unit of *privilege*. However, in the majority of the applications in medical IS, it seems that privileges are more naturally expressed in terms of access to and modification of data. This is hard to realize, however, since it cannot always be determined beforehand whether or not a method will access and/or modify a given data-unit.

Last, since the outcome of an access control decision is often influenced by *context information*, such as time, location, etc., a policy language should have access to this kind of context information.

Given all these concerns, it becomes clear that it is in fact the *interaction* in its whole, that must be taken into consideration when describing and executing the access control decision process. An *interaction* is the chain of method invocations that are caused by an invocation of the client on the application server. An interaction is always carried out on behalf of a subject on a set of target-objects in a specific context. This subject can assume one or more roles, which can be activated when and if needed. An interaction can also be tagged with some additional contextual information. Interactions should be identified and grouped into policy domains, in a fine-grained and also flexible manner. Domains then allow for the association of conditions with each individual interaction. This is the ultimate requirement for a policy language.

### 4.2 An Assessment of J2EE and .NET

Modern application platforms such as J2EE and .NET provide support for activating infrastructural services such as access control in a declarative way. As a consequence, access control policies are reconfigurable at deployment time or even at run time. But the expressiveness of such declarative access control policies is rather limited. For more complex, fine grained policies, access control is enforced in the code, leading to policies that are harder to adapt.

J2EE offers a minimal support for RBAC. Security roles can be defined (in principle by the developer) by grouping users in categories. The mapping of these security roles to security identity is done at deployment time by the application deployer. The permissions assigned to the security roles are based on method-invocations. Each method specification needs to describe the roles that have permission to invoke the method. A J2EE container allows by default all users

("AllUsers") to invoke all methods, so access permissions definitely need to be restricted.

.NET features are comparable with J2EE. A developer can tag methods with the roles that are required to execute the method by using the PrincipalPermissionAttributes. .NET also provides a bridge to the access control support that was present in COM+. The COM+ access control technology allows the deployer to attach required roles to methods that are exposed by the application.

If one uses these declarative security systems, access control is reconfigurable, but fails to offer the expressive power as stated in section 4.1:

- The policy domain is determined by the type (class) of the target object only. This means that it is impossible to enforce access rules that are specific for instances.
- Grouping of subjects is only possible on the basis of roles that have to be assigned statically. Neither role-activation nor the enforcement of dynamic separation of duty is supported. In general, it is the application container that keeps track of the subject on whose behalf the invocation is made. There is no possibility to attach custom-defined data to the invocation.
- Access control rules are set up in terms of method-invocations. If the deployer wants to enforce a data modification access rule, the deployer needs to find out which methods may possibly modify the data.
- Further context information, such as external factors (context, time, ...) can not be used in the access control decision.

The only way to deal with the above mentioned limitations is by hard coding security into the application. But by using such *programmatic security*, reconfigurability and even adaptability are lost.

### 4.3 A Solution Based on Extensible Interceptor Chains

A first step toward *interaction*-based access control consists of supporting *extensible interceptor chains*. An interceptor is a construct that intercepts method-invocations and that carries out some extra functionality before passing on, blocking, or redirecting the invocation. Depending on the considered technology, interceptor-instances can be deployed per container, class, object instance and/or context. It is fair to say that both J2EE and .NET provide some preliminary support.

For example, the open source J2EE-compliant JBoss application server tries to disentangle security code and application-logic by the definition of so-called security proxies ([14]). These proxies enforce the security policy outside the bean, but they still have to be coded programmatically.

.NET apparently has configurable interceptor chains built in already, but this is an undocumented feature rather than a solution. So-called context bound objects in .NET can be decorated with attributes that specify what interceptor should handle all method calls on the object. This interceptor can programmed in any of the .NET languages.

Our team has been experimenting with application architectures that support flexible interceptor chains since about five years, and we have been applying this approach in security frameworks for electronic commerce ([7]). Clearly, if an interceptor is used to implement access control, adaptability can be achieved: the access control logic is encapsulated in the interceptor, and relatively easily replaceable (at least at development time) by another interceptor. This solution has its limitations: reconfigurability is not possible, and access control is certainly not specified at a high level of abstraction. Rules need to be programmed in a general purpose programming language with a reified method call as primary input. Expressiveness however is significantly better, since the interceptor can base its decision on object state, application state and available context information.

## 5 The Way Forward

*Towards reconfigurable, expressive and high level access control.*

Extensible interceptor chains as discussed above are only a first step towards achieving the goal of reconfigurable, expressive and high level access control. Some issues still need to be solved.

- Programming interceptors is too low-level: a more developer-friendly programming model is necessary. Most probably, the interceptor should be interpreting a high-level policy language. Therefore, research about policy languages ([5]) is relevant. Then the actual policy can be configured at run time.
- Current application containers do not pass much context information to the objects inside. For instance, context information about the current interaction, such as where the interaction was started (on a remote network or a local network) might influence an access control decision, but this information is not available to an interceptor.

Both issues are (at least in part) addressed by the Aspect Oriented Software Development (AOSD) community ([3]). AOSD has as objective to modularize concerns, such as security, into slices of behavior. This modularization not only consists of the implementation of the required functionality but also the composition of these slices into the overall application and the management of the relations between slices. A variety of techniques are being investigated, varying from the weaving of extra code at compile time (AspectJ [4]) to the (runtime) deployment of wrappers (Lasagne, JAC [11,15]). The main difference between these AOSD systems and the previously described extensible interceptor chains is that there is considerably more support for the creation and composition of these slices of behavior. Often new language primitives are introduced to support the definition of these aspects. This addresses the first issue raised above. In order to support the consistent activation of aspects, and to support client-specific views, Truyen et al. [15] have suggested to attach metadata to interactions to drive the activation. Such techniques can address the second issue raised above: contextual information about the interaction has to travel with the interaction as metadata.

## 6 Conclusion

This paper has argued, on the basis of legislation and on the basis of the evolution of healthcare, that IT enforced access control policies in medical information systems will be fine-grained and dynamic. As a consequence a reconfigurable or at least adaptable implementation of access control is very important. Current application servers were shown to provide insufficient support for this, and some indications of how these shortcomings could be remedied were given.

## References

1. R. J. Anderson, *Patient Confidentiality – At Risk from NHS Wide Networking.* Proceedings of Health Care 96, March 96
2. R. J. Anderson, *A Security Policy Model for Clinical Information Systems.* IEEE Symposium on Security and Privacy, Oakland, CA, pp 30-43, May 1996
3. `http://www.aosd.net`
4. `http://aspectj.org`
5. N. Damianou, *A Policy Framework for Management of Distributed Systems*, PhD thesis, Februari 2002
6. I. Denley, S. Weston Smith, *Privacy in Clinical Information Systems in Secondary Care*, British Medical Journal, 318:1328–30, May 1999
7. B. De Win, J. Van den Bergh, F. Matthijs, B. De Decker, and W. Joosen, *A security architecture for electronic commerce applications*, Information Security for Global Information Infrastructures (S. Qing and J. Eloff, eds.), Kluwer Academic Publishers, 2000, pp. 491-500
8. European Parliament and Council of Europe *Directive 95/46/EC, on the protection of individuals with regard to the processing of personal data and on the free movement of such data*, October 24, 1995
9. Council of Europe, *Recommendation R(97)5, On the protection of medical data*, February 12, 1997
10. R. Goodwin, S.F. Foh, F.Y. Wu, *Instance-level access control for business-to-business electronic commerce*, IBM Systems Journal, Volume 41, number 2, Januari 2002
11. R. Pawlak, L. Duchien, G. Florin, L. Seinturier *JAC : a Flexible Solution for Aspect Oriented Programming in Java*, Reflection 2001, Kyoto, Japan, September 2001
12. Secretary of the Department of Health and Human Services, *Final Privacy Rule*, August 14, 2002
13. Secretary of the Department of Health and Human Services, *Final Security Rule*, February 20, 2003
14. L. Taylor *Customized EJB security in JBoss. Separate your security policy from your business logic*, JavaWorld, February 2002
15. E. Truyen, B. Vanhaute, W. Joosen, P. Verbaeten, B. N. Joergensen, *A Dynamic Customization Model for Distributed Component-Based Applications*, accepted for International Workshop on Dynamic and Distributed Multiservice Architectures (DDMA 2001)
16. T. Verhanneman, L. Jaco, B. De Win, F. Piessens, W. Joosen, *Adaptable Access Control Policies for Medical Information Systems: requirements analysis and case studies*, technical report, Katholieke Universiteit Leuven, CW363, August 2003