Ensuring Spatio-Temporal Access Control for Real-World Applications

Manachai Toahchoodee
Computer Science Dept.
Colorado State University
Fort Collins, CO 80523
toahchoo@cs.colostate.edu

Indrakshi Ray
Computer Science Dept.
Colorado State University
Fort Collins, CO 80523
iray@cs.colostate.edu

Kyriakos Anastasakis
School of Computer Science
University of Birmingham
Edgbaston, Birmingham, UK
K.Anastasakis@cs.bham.ac.uk

Geri Georg
Computer Science Dept.
Colorado State University
Fort Collins, CO 80523
georg@cs.colostate.edu

Behzad Bordbar
School of Computer Science
University of Birmingham
Edgbaston, Birmingham, UK
B.Bordbar@cs.bham.ac.uk

ABSTRACT

Traditional access control models, such as Role-Based Access Control (RBAC), do not take into account contextual information, such as location and time, for making access decisions. Consequently, they are inadequate for specifying the access control needs of many complex real-world applications, such as the Dengue Decision Support (DDS) that we discuss in this paper. We need to ensure that such applications are adequately protected using emerging access control models. This requires us to represent the application and its access control requirements in a formal specification language. We choose the Unified Modeling Language (UML) for this purpose, since UML is becoming the defacto specification language in the software industry. We need to analyze this formal specification to get assurance that the application is adequately protected. Manual analysis is error-prone and tedious. Thus, we need automated tools for verification of UML models. Towards this end, we propose that the UML models be converted to Alloy. Alloy is based on first-order logic, has a software infrastructure that supports automated analysis, and has been used for the verification of real-world applications. We show how to convert the UML models to Alloy and verify the resulting model using the Alloy Analyzer which has embedded SAT-solvers. The results from the Alloy Analyzer will help uncover the flaws in the specification and help us refine the application and its access control requirements.

Categories and Subject Descriptors

D.2.1 [**Requirements/Specifications**]: [Languages, Methodologies]; K.6.5 [**Management of Computing and Information Systems**]: [Security and Protection]

General Terms

Design, Languages, Security

SACMAT'09, June 3–5,2009, Stresa, Italy.

Keywords

Modeling, Spatio-Temporal RBAC, UML, Alloy

1. INTRODUCTION

Traditional models, such as Discretionary Access Control (DAC) or Role-Based Access Control (RBAC) do not take into account contextual information, such as, location and time, before making access decisions. Such models are often inadequate for new types of applications that are being developed. Consequently, researchers have proposed various access control models that use contextual information, such as, location and time, for performing access control [3, 4, 5, 8, 10, 14, 21, 23, 31, 28, 29, 35]. However, when such a new access control model is being used for a novel application, we must have assurance that the application is being adequately protected. We propose a methodology that describes how we can get assurance that an application is adequately protected.

We use a real-world dengue decision support (DDS) application to illustrate our approach. DDS helps state-level public health officials respond to local outbreaks of dengue. The users of the system are assigned different roles and perform tasks that are assigned to their roles. In order to perform a specific task, a user needs to be assigned to a specific role and has to be in the specific location during the specific time. Roles of users may also be related through the hierarchical structure of the organization. Moreover, the separation of duty needs to be applied to ensure that no user can be assigned two or more conflicting tasks. Our previously proposed Spatio-Temporal Role Based Access Control (STRBAC) model is adequate for meeting these requirements, so we chose to use it.

The Spatio-Temporal RBAC (STRBAC) model [31] that we proposed is an extension of RBAC model that supports spatial and temporal constraints. We describe how each entity in this model is constrained by location and time. Our model supports the various features of RBAC including role hierarchy and separation of duty. The model is more expressive than its traditional counterparts, and has various features which the users can selectively use based on their application requirements. In this paper, we propose how to use the simplified version of STRBAC to support the Dengue Decision Support (DDS) system and gain assurance that the application is adequately protected.

In order to formally analyze the access control for the application, it is important to specify the application and their access control requirements in a formal specification language. We chose to

use the Unified Modeling Language (UML) [27] for this purpose for several reasons. First, it is the de facto modeling language used in the software industry. Second, it is easy to use and understand. Third, it is used together with Object Constraint Language (OCL), which is based on first order predicate logic; this makes it amenable to analysis. We show how the existing access control requirements for the DDS can be specified using UML.

Although formal analysis can be done on UML specifications that are augmented with OCL constraints, there is not much tool support for automated analysis. Towards this end, we advocate the use of Alloy [18] for doing automated analysis. Alloy is a specification language capable of expressing complex structural constraints and behavior. Alloy specifications can be automatically analyzed using the Alloy Analyzer which has embedded SAT-solvers. Moreover, it has been successfully used in the modeling and analysis of real-world systems [13, 39].

We begin by specifying the structure and security constraints of a spatio-temporal role-based access control model for DDS system using UML 2.0 and accompanying OCL constraints. We then use UML2Alloy [6, 7] to automatically transform the class diagram and OCL statements into an Alloy model which is subsequently analyzed. Alloy is supported by an automated constraint solver called Alloy Analyzer that searches instances of the model to check for satisfaction of system properties. The model is automatically translated into a Boolean expression, which is analyzed by SAT solvers embedded within the Alloy Analyzer. A user-specified scope on the model elements bounds the domain, making it possible to create finite Boolean formulas that can be evaluated by the SAT-solvers. When a property does not hold, a counterexample is produced that demonstrates how it has been violated. The analysis demonstrates how adequately the access control requirements protect the application.

The rest of the paper is organized as follows. Section 2 describes the related work. Section 3 briefly describes the simplified version of our STRBAC model. Section 4 outlines the transformation process from UML to Alloy. Section 5 provides the background information of the Dengue Decision Support system and describes the spatio-temporal security policies applied to it. Section 6 discusses how the model can be analyzed using Alloy. Section 7 concludes the paper with some pointers to future directions.

2. RELATED WORK

Role-based access control model [11] is used for addressing the access control needs of commercial organizations. Several works exist that improve RBAC functionality. Some of these focus on how RBAC can be extended to make it context aware. Sampemane et al. [34] present a new access control model for *active spaces*. Active space denotes the computing environment integrating physical spaces and embedded computing software and hardware entities. The active space allows interactive exchange of information between the user and the space. Environmental aspects are adopted into the access control model for active spaces, and the space roles are introduced into the implementation of the access control model based on RBAC.

Covington et al. [10] introduce environment roles in a generalized RBAC model (GRBAC) to help control access to private information and resources in ubiquitous computing applications. The environment roles differ from the subject roles in RBAC but do have similar properties including role activation, role hierarchy and separation of duty. In the access control framework supporting environmental roles, each element of permission assignment is associated with a set of environment roles, and environment roles are activated according to the changes specified in environmental conditions. This allows environmental properties, such as time and location, to be introduced into the access control framework. In a subsequent work [9], Covington et al. describe the Context-Aware Security Architecture (CASA) which is an implementation of the GRBAC model. In CASA, the security management service is responsible for managing the access control policies, and authentication and authorization services are used to verify user credentials and determine access to the system resources. The environmental role activation services manage environmental role activation and deactivation according to the environment variables collected by the context management services.

Other extensions to RBAC include the Temporal Role-Based Access Control Model (TRBAC) proposed by Bertino et al. [4]. This work adds the time dimension to the RBAC model. The authors introduce the concept of role enabling and disabling. Temporal constraints determine when the roles can be enabled or disabled. A role can be activated only if it has been enabled. Joshi et al.[21] extend this work by proposing the Generalized Temporal Role Based Access Control Model (GTRBAC). In this work the authors introduce the concept of time-based role hierarchy and time-based separation of duty. These works do not discuss the impact of spatial information on access control.

Researchers have also extended RBAC to incorporate spatial information. One such example is the GEO-RBAC [5]. In this model, role activation is based on the location of the user. For instance, a user can acquire the role of teacher only when he is in the school. Outside the school, he can acquire the role of citizen. The model supports role hierarchies but does not deal with separation of duties. Another work incorporating spatial information is LRBAC proposed by Ray et al. [29]. Here again, the authors propose how each component of RBAC is influenced by location. The authors define their formal model using the Z specification language. Role hierarchy and separation of duties are not addressed in this paper. None of these work discuss the impact of time on location. Location-based access control has been addressed in other works not pertaining to RBAC [14, 23, 28].

The paper proposed by Chandran et al. [8] combines the main features of GTRBAC and GEO-RBAC. Here again, a role is enabled by time constraints. The user can activate the role if the role is enabled and the user satisfies the location constraints associated with role activation. Another work which falls into this category is GST-RBAC by Samuel et al. [35]. In this work, the authors develop a framework to incorporate topological spatial constraints to the existing GTRBAC model. The authors do this by augmenting GTRBAC operations, namely, role enabling, user-role assignment, role-permission assignment, and role-activation with spatial constraints. The operations are allowed only if the spatial and temporal constraints are satisfied. The model also introduces the notion of Spatial Role Hierarchy and Spatial Separation of Duty (spSoD) constraints. Our early work [31] extends RBAC with spatial and temporal constraints. Although the goal of this work is similar to those proposed by Chandran et al. and Samuel et al., our model can express some real-world constraints that are not possible in the other ones. In our model, a user can activate a role at specific times and at designated locations. After a user's role is activated, he can access the resource only if his role has the permission to do so and the spatio-temporal constraints associated with the permissions are satisfied. The model also supports the spatio-temporal role hierarchies and separation of duties constraints. In a subsequent work [32], we extend the model to support the delegation operation.

A lot of work also appears in the area of analysis of security policies. Researchers have used formal logic for specifying authorization policies so that they can be analyzed. Many works appear

that attempt to analyze RBAC specifications. Some have used the Z modeling language for specifying RBAC [42] and LRBAC [29]. Although Z language can represent RBAC and its constraints in a formal manner, the language itself lacks the tool to support the automated analysis of the formalized model. Others have used an extension of the Unified Modeling Language (UML) [30] called parameterized UML to visualize the properties of RBAC constraints. The model describes how one can visualize the conflicts that may occur within RBAC constraints. However, it lacks the ability to perform automated model analysis.

ALLOY [16, 17, 18, 43] is a textual language developed at MIT by Daniel Jackson and his team. Alloy is a fully declarative first-order logic language designed for modeling and analyzing complex systems. Alloy is supported by a fully automated constraint solver, called Alloy Analyzer, which allows analysis of system properties by searching for instances of the model. Alloy has been used in the verification of real world systems and protocols [13, 12, 39].

Bordbar et al. propose UML2Alloy [2, 6, 7], a tool to automatically transform UML model to Alloy specification. The transformation makes use of Model Driven Architecture (MDA) [22] techniques for defining and implementing the transformations from models captured in the UML class diagram and OCL into Alloy. The theoretical foundations and the challenges incurred in doing this transformation appear in one of our more recent works [1].

Researchers have also advocated the use of Alloy for modeling RBAC specifications. Schaad et al. [37] model user-role assignment, role-permission assignment, role hierarchy, and static separation of duties features of RBAC extension using Alloy. The authors do not model role activation hierarchy or the dynamic separation of duties. The authors briefly describe how to analyze conflicts in the context of the model. Zao et al. [43] model basic features of RBAC, role hierarchy, and static separation of duties in Alloy.

Hu and Ahn [15] propose an Enhanced Assurance Management Framework to verify and test the access control system. In this work, the authors provide algorithms to show how RBAC models can be specified in Alloy. The Alloy Analyzer will automatically analyze this model to ensure the correctness of the access control specification. Later, the test cases generated from the verified model will be used to perform a conformance testing, where the actual result from model implementation will be compared with the expected result derived from the formal specification. Our work focuses on automated translations of UML specifications to Alloy. We use OCL for specifying constraints instead of using a specialized constraint language RCL2000. Our biggest difference is in describing how the spatio-temporal aspects can be verified using Alloy.

Samuel et al. [35] also illustrate how GST-RBAC can be specified in Alloy. They describe how the various GST-RBAC functionalities, that is, user-role assignment, role-permission assignment, and user-role activation, can be specified by Alloy. In our recent work [40], we propose using Alloy to study the interaction of the various features of our previously proposed spatio-temporal RBAC model [31]. The analysis reveals 13 types of conflict that may arise due to the interaction of the model features. In another work [41], we adapt this approach to verify a more complex model–a spatio-temporal RBAC with delegation. However, both of these works focus on analyzing the model in isolation and are useful in describing potential conflicts that may occur between the different features of the model. Such analysis is independent of the application. The current work, on the other hand, analyzes the behavior of the application using our spatio-temporal access control model. Such analysis will help illustrate whether any conflicts are produced by the access control requirements of the given application.

3. STRBAC MODEL

In this section we describe the simplified version of our STRBAC model that is needed for the DDS application. For the full version of our model, we refer the interested reader to [31, 32, 40].

3.1 STRBAC Components

The simplified STRBAC model consists of the different entities corresponding to RBAC entities which are *Users*, *Roles*, and *Permissions*. In this section, we describe how the entities in RBAC are associated with location and time. Throughout this paper, location is represented as a *physical location*, which is a set of points in three dimensional space, and time is represented as a *time interval* which is a set of time instants. For more detail about the location and time representation, please refer to our previous STRBAC paper [31].

3.1.1 Users

We assume that each valid user, interested in doing some location-sensitive operation, carries a locating device which is able to track his location. The location of a user may change with time. The relation $UserLocation(u,t)$ gives the location of the user at any given time instant t. Since a user can be associated with only one location at any given point of time, the following constraint must be true. Note that in this and all the subsequent formulae, we omit the quantification symbols.

$$UserLocation(u,t) = l_i \wedge UserLocation(u,t) = l_j \Leftrightarrow (l_i \subseteq l_j) \vee (l_j \subseteq l_i)$$

3.1.2 Roles

We have two types of relations with roles. These are user-role assignment, and permission-role assignment. We discuss user-role assignment in this section, and later we discuss permission-role assignment. Often times, the assignment of a user to a role is location and time dependent. For instance, a person can be assigned the role of State Epidemiologist only in certain designated locations and at certain times only. To get the role of Jurisdiction Epidemiologist, a person must be at the jurisdiction office during regular hours. Thus, for a user to be assigned a role, he must be in designated locations during specific time intervals. In our model, a user must satisfy spatial and temporal constraints before roles can be assigned. We capture this with the concept of *role allocation*. A role is said to be *allocated* when it satisfies the temporal and spatial constraints needed for role assignment. A role can be assigned once it has been allocated. $RoleAllocLoc(r)$ gives the set of locations where the role can be allocated. $RoleAllocDur(r)$ gives the time interval where the role can be allocated. If some role s can be allocated anywhere, then $RoleAllocLoc(s) = universe$. Similarly, if role p can be assigned at any time, we specify $RoleAllocDur(p) = always$.

The predicate $UserRoleAssign(u,r,d,l)$ states that the user u is assigned to role r during the time interval d and location l. For this predicate to hold, the location of the user when the role was assigned must be in one of the locations where the role allocation can take place. Moreover, the time of role assignment must be in the interval when role allocation can take place.

$$UserRoleAssign(u,r,d,l) \Rightarrow (UserLocation(u,d) = l) \wedge (l \subseteq RoleAllocLoc(r)) \wedge (d \subseteq RoleAllocDur(r))$$

3.1.3 Permissions

Permission is the ability to perform a corresponding task. Permissions are associated with roles. A user can acquire all permissions associated with his role only when he is in the designated location and time while his role is active. We define another predicate which we term $PermRoleAcquire(p,r,d,l)$. This predicate is

true if role r has permission p for duration d at location l. Note that, for this predicate to be true, the time interval d must be contained in the duration where the role is active. Similarly, the location l must be contained in the places where the role is active.

3.2 Role Hierarchy

The structure of an organization in terms of lines of authority can be modeled as an hierarchy. This organization structure is reflected in RBAC in the form of a role hierarchy [36]. Role hierarchy is a relation among roles. This relation is transitive and anti-symmetric. Roles higher up in the hierarchy are referred to as senior roles and those lower down are junior roles. The major motivation for adding role hierarchy to RBAC was to simplify role management. Senior roles can inherit the permissions of junior roles, or a senior role can activate a junior role, or do both depending on the nature of the hierarchy. This obviates the need for separately assigning the same permissions to all members belonging to a hierarchy.

Joshi et al. [21] identify two basic types of hierarchy. The first is the permission inheritance hierarchy where a senior role x inherits the permission of a junior role y. The second is the role activation hierarchy where a user assigned to a senior role can activate a junior role. Each of these hierarchies may be constrained by location and temporal constraints. Consequently, we have a number of different hierarchical relationships in our model. In this section, we will consider only the spatio-temporal permission inheritance hierarchy. For the spatio-temporal role activation hierarchy, we refer to [31].

Definition 1 [Unrestricted Permission Inheritance Hierarchy] Let x and y be roles such that $x \geq y$, that is, senior role x has an unrestricted permission-inheritance relation over junior role y. In such a case, x inherits y's permissions but not the locations and time associated with it. In other words, the permission can be applied wherever the senior role is at that time. This is formalized as follows:

$$(x \geq y) \wedge PermRoleAcquire(p,y,d,l) \Rightarrow PermRoleAcquire(p,x,d',l')$$

In the above hierarchy, a senior role inherits the junior roles permissions. However, unlike the junior role, these permissions are not restricted to time and location. Account auditor role inherits the permissions from the accountant role. He can use the permissions at any time and at any place.

Definition 2 [Time Restricted Permission Inheritance Hierarchy] Let x and y be roles such that $x \geq_t y$, that is, senior role x has a time restricted permission-inheritance relation over junior role y. In such a case, x inherits y's permissions together with the temporal constraints associated with the permission. This is formalized as follows:

$$(x \geq_t y) \wedge PermRoleAcquire(p,y,d,l) \Rightarrow PermRoleAcquire(p,x,d,l')$$

In the above hierarchy, a senior role inherits the junior roles permissions. However, the duration when the permissions are valid are those that are associated with the junior roles. A contact author can inherit the permissions of the author until the paper is submitted.

Definition 3 [Location Restricted Permission Inheritance Hierarchy] Let x and y be roles such that $x \geq_l y$, that is, senior role x has a location restricted permission-inheritance relation over junior role y. In such a case, x inherits y's permissions together with the location constraints associated with the permission. This is formalized as follows:

$$(x \geq_l y) \wedge PermRoleAcquire(p,y,d,l) \Rightarrow PermRoleAcquire(p,x,d',l)$$

In the above hierarchy, a senior role inherits the junior roles permissions. These permissions are restricted to the locations imposed on the junior roles. A top secret scientist inherits the permission of top secret citizen only when he is in top secret locations.

Definition 4 [Time Location Restricted Permission Inheritance Hierarchy] Let x and y be roles such that $x \geq_{tl} y$, that is, senior role x has a time-location restricted permission-inheritance relation over junior role y. In such a case, x inherits y's permissions together with the temporal and location constraints associated with the permission. This is formalized as follows:

$$(x \geq_{tl} y) \wedge PermRoleAcquire(p,y,d,l) \Rightarrow PermRoleAcquire(p,x,d,l)$$

In the above hierarchy, a senior role inherits the junior roles permissions. These permissions are restricted to time and locations imposed on the junior roles. Daytime doctor role inherits permission of daytime nurse role only when he is in the hospital during the daytime.

3.3 Separation of Duty

Separation of duties (SoD) enables the protection of the fraud that might be caused by the user [38]. Separation of Duty (SoD) comes in two varieties. First one is with respect to user role assignment. The second one is with respect to permission role assignment. In this paper, we will focus on the Static Separation of Duty for permission role assignment. The idea is that the same role should not acquire conflicting permissions. Due to the presence of temporal and spatial constraints, we can have different flavors of separation of duties – some that are constrained by temporal and spatial constraints and others that are not. In the following we describe the different separation of duty constraints.

Definition 5 [Weak Form of SSoD - Permission Role Assignment] Let p and q be two permissions such that $p \neq q$. $(p,q) \in SSOD\_PRA_w$ if the following condition holds:

$$PermRoleAcquire(p,x,d,l) \Rightarrow \neg\, PermRoleAcquire(q,x,d,l))$$

The above definition says that if permissions p and q are related through weak SSoD Permission Role Assignment and x has permission p at time d and location l, then x should not be given permission q at the same time and location.

Definition 6 [Strong Temporal Form of SSoD - Permission Role Assignment] Let p and q be two permissions such that $p \neq q$. $(p,q) \in SSOD\_PRA_t$ if the following condition holds:

$$PermRoleAcquire(p,x,d,l) \Rightarrow \\ \neg\, (\exists d' \subseteq always \bullet PermRoleAcquire(q,x,d',l))$$

The above definition says that if permissions p and q are related through strong temporal SSoD Permission Role Assignment and x has permission p at time d and location l, then x should not get permission q at any time in location l.

Definition 7 [Strong Spatial Form of SSoD - Permission Role Assignment] Let p and q be two permissions such that $p \neq q$. $(p,q) \in SSOD\_PRA_l$ if the following condition holds:

$$PermRoleAcquire(p,x,d,l) \Rightarrow \\ \neg\, (\exists l' \subset universe \bullet PermRoleAcquire(q,x,d,l'))$$

The above definition says that if permissions p and q are related through strong spatial SSoD Permission Role Assignment and x has permission p at time d and location l, then x should not be given permission q at the same time at any location l'.

Definition 8 [Strong Form of SSoD - Permission Role Assignment] Let p and q be two permissions such that $p \neq q$. $(p,q) \in SSOD\_PRA_S$ if the following condition holds:

$$PermRoleAcquire(p,x,d,l) \Rightarrow \neg\,(\exists l' \subset universe, \exists d' \subseteq always \bullet PermRoleAcquire(q,x,d',l'))$$

The above definition says that if permissions p and q are related through strong SSoD Permission Role Assignment, then the same role should never be given the two conflicting permissions.

4. UML TO ALLOY TRANSFORMATION

Since the applications are generally specified in UML, we use UML to specify our application and access control constraints as well. UML can be used in conjunction with OCL which is based on formal logic; this allows us to formally specify the constraints in our model. However, in order to get assurance that our application is adequately protected, we need to analyze our application together with the access control constraints. Manual analysis is tedious and error-prone, so we need to automate the verification process. Existing tools for automated analysis of UML models, such as USE and OCLE, cannot verify behavioral properties and so are inadequate. We propose an approach that will transform UML models with OCL constraints into an Alloy specification. We discuss the details of the transformation process after giving some background in Alloy.

4.1 ALLOY Lightweight Modeling System

ALLOY [16, 17, 18, 43] is a fully declarative first-order logic language designed for modeling and analyzing complex systems. An Alloy model consists of a number of signature and relation declarations. A signature specifies entities used to model the system, and relation declarations specify the dependencies between such entities, allowing the designer to capture complex structures. Alloy is supported by a fully automated constraint solver, called Alloy Analyzer, that analyzes system properties by searching for model instances that violate assertions about them. Alloy Analyzer translates the model into a Boolean expression, and analyzes it using embedded SAT-solvers. The user specifies a scope to the tool, which is an integer number used to bound the domain of model elements. Bounding enables the tool to create finite Boolean formulas for evaluation by the SAT-solver. If Alloy Analyzer produces an instance that violates the assertion (a counterexample), we can infer that the specified property is not satisfied. However, for a chosen scope, if no counterexample emerges, it is possible that the property is violated in a larger scope. Choosing a larger scope provides greater assurance, but the analysis takes longer time to complete [19]. However, design flaws are often discovered in small scopes. This is known as "small scope hypothesis" [19]. Choosing the right scope, and the degree of confidence a given scope provides, depends on the problem being analyzed. Currently, we do not provide any guidelines on how to choose the scope for verifying access control requirements. We propose to use the Alloy Analyzer as a first line of defense to discover flaws in the access control requirements. If the analyzer does not produce a counterexample, other techniques such as Model Checking and Theorem Proving can be used to provide a higher level of assurance. Such techniques are more time consuming and require human intervention and expertise. Our approach can therefore save time and resources by using the Alloy Analyzer to rapidly discover a number of flaws that would otherwise require much more time and resources to uncover. For more details on Alloy and its comparison with other formal methods please refer to [17, 18, 19].

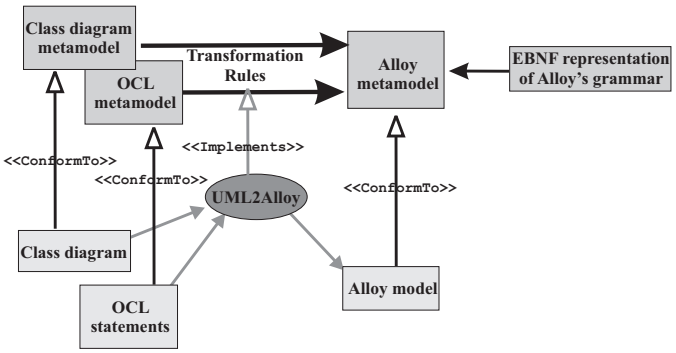

Figure 1: Outline of the transformation method.

4.2 Model Transformation from UML to Alloy

There are clear similarities between Alloy and UML languages such as class diagrams and OCL. From a semantic point of view both Alloy and UML can be interpreted by sets of tuples [20, 33]. Alloy is based on first-order logic and is well suited for expressing constraints on Object Oriented models. Similarly, OCL has extensive constructs for expressing constraints as first order logic formulas. Considering such similarities, model transformation from UML class diagrams and OCL to Alloy seems straightforward. However, UML and Alloy have fundamental differences, which are deeply rooted in their underlying design decisions. For example Alloy makes no distinction between sets, scalars and relations, while the UML makes a clear distinction between the three. Other examples include that UML supports a number of primitive types, whereas Alloy only supports integers. UML also supports aggregation and composition, but there is no counterpart in Alloy. All of this makes the transformation from UML to Alloy challenging.

Figure 1 depicts an outline of our approach. Using the Extended Backus-Naur Form (EBNF) representation of the Alloy grammar [20], we shall first generate a Meta Object Facility (MOF) compliant [24] metamodel for Alloy. We then select a subset of the class diagrams [26] and OCL [25] metamodels. To conduct the model transformation, a set of transformation rules has been defined. The rules map elements of the metamodels of class diagram and OCL into the elements of the metamodel of Alloy. The rules have been implemented into a prototype tool called UML2Alloy. If a UML class diagram, which conforms to the subset of UML we support, is provided as input to UML2Alloy, it automatically generates an Alloy model. For lack of space, we do not show how the EBNF representation of Alloy's grammar is transformed into a MOF compliant metamodel but refer the interested reader to [2].

4.3 Mapping Class diagram and OCL to Alloy

The transformation rules map elements of the UML class diagram and OCL metamodels to the Alloy metamodel. Due to space limitations the UML and OCL metamodels are not presented here, but can be found in the respective specification documents [26, p. 29], [25].

Table 1 presents a table which provides an informal mapping

UML+OCL metamodel element	Alloy metamodel element
Class	ExtendsSigDecl
Property	DeclExp
Operation	Predicate
Parameter	Decl
Enumeration	ExntedsSigDecl
EnumerationLiteral	ExtendsSigDecl
Constraint	Expression

Table 1: Informal mapping between UML and Alloy metamodel elements

between the most important elements of the UML and OCL metamodels and Alloy. More specifically a UML *Class* is translated to an Alloy signature declaration (*ExtendsSigDecl*), which defines a *SigId* with the same name. If the class is not a specialization the Alloy signature is not related to any *SigRef*. Otherwise it might be related to a *SigRef*, which references the signature it might extend.

A *Property* is translated to a declaration expression (*declExp*), which is used to define a field in an Alloy model. An *Operation* is transformed to a *Predicate* and the *Parameters* of the operation are transformed to declarations (*Decl*). An Enumeration [26, p. 63] is transformed to a signature declaration *SigDecl*, which declares an abstract signature. An *EnumerationLiteral* is transformed to a sub signature. A more complete transformation rules from UML to Alloy and their implementation are explained in our previous work [2].

5. DENGUE DECISION SUPPORT SYSTEM

We illustrate our approach using a real-world Dengue Decision Support (DDS) system. The DDS helps state-level public health officials respond to local outbreaks of dengue. Response consists of vector control and vector surveillance, namely, spraying (control) and investigating locations where mosquitoes might be breeding and living (surveillance) and the level of confirmed dengue cases has increased above a prescribed threshold. Public health officials are organized in jurisdictions, based on population, and multiple jurisdictions are included in a single state. When the threshold is reached, officials at both levels respond. The jurisdiction officer activates vector control and surveillance teams that are local to the jurisdiction, with instructions regarding the specific control and surveillance protocols to follow and the locations where they are to be performed. The state officer releases materials for control to the team, and the local team then performs the controls and surveillance ordered. The jurisdiction and state vector control officials are often located in different buildings, although the vector control team is co-located with the jurisdiction officer. All control materials are located in warehouses elsewhere, and for coordination reasons are controlled by the state officer. Information about specific cases of dengue is retained in what is called an epidemiological study. This data includes information about the patient, the location where the patient lives (the premise), the case, and control and surveillance actions performed at the premise. The patient and case data are considered private information, and are only available to epidemiologists at the jurisdiction and state levels. The vector control team receives premise information along with orders for control and surveillance. However, the team also needs to have names associated with the premises in order to validate the location. The team therefore needs access to some of the patient data for a fixed period of time, in order to perform control and surveillance duties. For lack of space, we omit giving the full specification.

Table 2: DDS Tasks List

	Task		Task
1	Read Premise	10	Read VControl
2	Change Premise	11	Change VControl
3	Read Case	12	Read Work Record
4	Change Case	13	Change Work Record
5	Read Patient	14	Read VC Materials
6	Change Patient	15	Change VC Materials
7	Read Patient Names	16	Signal VC Need for DV
8	Read Schedule Work	17	Signal VC Need for DHF
9	Change Schedule Work		

5.1 Security Policies

5.1.1 Entities

DDS system consists of the following roles: *State Epidemiologist*, *Jurisdiction Epidemiologist*, *Clinic Epidemiologist*, *Clinician*, *State Vector Control*, *Jurisdiction Vector Control*, and *Local Jurisdiction VC Team*. Tasks user can perform are listed in Table 2. Each role can perform their own set of tasks in the designated location and time summarized in Table 3.

5.1.2 Role Hierarchy

Some roles in the DDS are related using unrestricted permission inheritance hierarchy. Using the STRBAC model, these relationships can be define as follow: *State Epi* $\geq$ *Juris Epi*, *Clinic Epi* $\geq$ *Clinician*, and *State VC* $\geq$ *Juris VC*.

5.1.3 Separation of Duty

There are two separation of duty constraints in DDS system. Both are the strong spatial form of static separation of duty. These permissions should not be assigned to the same user at the same time at any location. Note however, unlike traditional separation of duty, these permissions can be assigned to the same user at different times.

1. User should not have permission to change VC protocols at the same time as he has permission to change VC materials.

2. User should not have permission to signal DV at the same time as signal DHF.

These can be represented in STRBAC as follow: $(11, 15) \in SSOD\_PRA_l$ and $(16, 17) \in SSOD\_PRA_l$.

6. MODEL ANALYSIS

The first step in formal security analysis is to abstract and transform the STRBAC model in the context of DDS into a UML class diagram and accompanying OCL. The class diagram depicts the entities that take part in the model, and defines their attributes related in the access control operations, such as the time and location attribute. OCL statements specify the invariants of the model such as the tasks assigned to role and security constraints that all entities in the model must satisfy. In the next step, we use UML2Alloy [6, 7] to automatically transform the class diagram and OCL statements into an Alloy model, which we subsequently analyze using Alloy Analyzer.

Table 3: DDS Role Constraints

Role	Tasks	Location Constraint	Time Constraint
State Epi	16	A–State Office	a–Regular Hours
Juris Epi	1, 3 17	B–Juris Office B–Juris Office	a–Regular Hours b–Any Time
Clinic Epi	17	C–Clinic	b–Any Time
Clinician	1, 2, 3, 4, 5, 6	C–Clinic	a–Regular Hours
State VC	11, 15	A–State Office	a–Regular Hours
Juris VC	1, 8, 9, 10, 12, 14	B–Juris Office	a–Regular Hours
Local VC Team	7 1, 9, 13	B–Juris Office, E–Emergency Location B–Juris Office, D–Field	c–24 Hours Window after signal to begin work received a–Regular Hours

6.1 Stage 1: Model Abstraction

The first step of the abstraction is to simplify the original model by removing non-essential elements so that the translation to Alloy produces a model that only contains items necessary to reason about its security properties. For example, we remove the attributes which are not related with the security such as, *gender*, *birthdate*, *ssid* from the *Person* entity since these attributes are not related with the access control model. The resulting UML class diagram is shown in Figure 2.

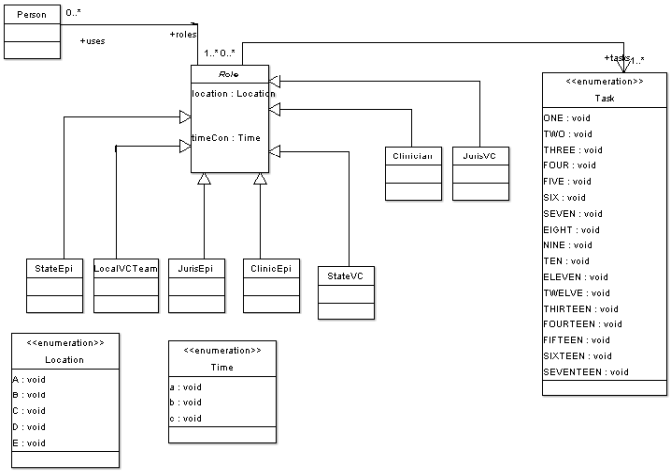

Figure 2: UML Model for the DDS's STRBAC

The permission role assignments are expressed as OCL constraints. The following OCL depicts the constraints for the permission role assignment for *Juris Epi* role.

```
context JurisEpi
inv jurisEpiCon : (self.tasks = (Task :: ONE ->
including (Task :: THREE)) and
self.location = Location :: B and
self.timeCon = Time :: a) or
(self.tasks = (Task :: SEVENTEEN -> including
(Task :: SEVENTEEN)) and
self.location = Location :: B and
self.timeCon = Time :: b )
```

The effect of permission inheritance hierarchy can also be expressed as OCL. The following OCL depicts the constraints for the permission role assignment for *State Epi* role.

```
context StateEpi
inv stateEpiCon : (self.tasks = (Task :: SIXTEEN ->
including (Task::SIXTEEN)) and
self.location = Location :: A and
self.timeCon = Time :: a) or
(self.tasks = (Task :: ONE -> including
(Task :: THREE)) and self.location = Location :: B and
self.timeCon = Time :: a) or
(self.tasks = (Task :: SEVENTEEN -> including
(Task :: SEVENTEEN)) and self.location = Location :: B
and self.timeCon = Time :: b)
```

Note that all permissions assigned to *Juris Epi*, which is the junior role of *State Epi* role are appended to the set of permissions assigned to *State Epi* role.

The separation of duty can also be modeled using OCL constraint. For instance, the constraint expressing that user should not have permission to change VC protocols at the same time as he has permission to change VC materials can be modeled as follows:

```
context Person
inv no_eleven_fifteen : self.roles ->
forAll (r1 , r2 : Role |
(r1.tasks -> includes (Task :: ELEVEN) implies
(r2.tasks -> excludes (Task :: FIFTEEN))) and
(r1.tasks -> includes (Task :: FIFTEEN) implies
r2.tasks -> excludes (Task :: ELEVEN)))
```

6.2 Stage 2: Model Transformation

The UML2Alloy tool is used to create an Alloy model from the class diagram and associated OCL specification.

When we apply UML2Alloy to the UML class diagram and its OCL specification, the class diagram will be transformed to the following *signatures* in Alloy corresponding to each class shown in Figure 2.

```
abstract  sig Role{
location:one Location,
timeCon:one Time,
tasks:some Task,
uses:set Person}

one sig StateEpi extends Role{}
one sig JurisEpi extends Role{}
one sig ClinicEpi extends Role{}
one sig Clinician extends Role{}
one sig StateVC extends Role{}
one sig JurisVC extends Role{}
one sig LocalVCTeam extends Role{}

some sig Person{roles:some Role}
```

```
abstract  sig Location{}
one sig A extends Location{}
one sig B extends Location{}
one sig C extends Location{}
one sig D extends Location{}
one sig E extends Location{}

sig Time{}
sig a in Time{}
sig b in Time{}
sig c in Time{}

abstract  sig Task{}
one sig ONE extends Task{}
one sig TWO extends Task{}
one sig THREE extends Task{}
one sig FOUR extends Task{}
one sig FIVE extends Task{}
one sig SIX extends Task{}
one sig SEVEN extends Task{}
one sig EIGHT extends Task{}
one sig NINE extends Task{}
one sig TEN extends Task{}
one sig ELEVEN extends Task{}
one sig TWELVE extends Task{}
one sig THIRTEEN extends Task{}
one sig FOURTEEN extends Task{}
one sig FIFTEEN extends Task{}
one sig SIXTEEN extends Task{}
one sig SEVENTEEN extends Task{}
```

The OCL constraint for the permission role assignment will be transformed to *fact* and *predicate* in Alloy. For example, the OCL constraint for the permission role assignment of the *Juris Epi* role will be transformed to the following Alloy code.

```
fact JurisEpi_jurisEpiCon_fact{
all self: JurisEpi | JurisEpi_jurisEpiCon[self]}

pred JurisEpi_jurisEpiCon[self: JurisEpi]{
((self.tasks = ONE+THREE) && (self.location = B) &&
(self.timeCon = a)) || ((self.tasks = SEVENTEEN) &&
(self.location = B) && (self.timeCon in Time))}
```

The effect of role hierarchy represented in the OCL constraint will also be transformed to *fact* and *predicate* in Alloy. For example, the OCL constraint for the set of permissions that assigned to the *State Epi* role through the role hierarchy will be transformed to the following Alloy code.

```
fact StateEpi_stateEpiCon_fact{
all self: StateEpi | StateEpi_stateEpiCon[self]}

pred StateEpi_stateEpiCon[self: StateEpi]{
(self.tasks = SIXTEEN + ONE + THREE + SEVENTEEN) &&
(self.location = A) && (self.timeCon = a)}
```

The OCL constraint for the separation of duty constraint will be transformed to *predicate* in Alloy. For instance, the OCL constraint says that user should not have permission to change VC protocols at the same time as he has permission to change VC materials will be transformed to the following Alloy code.

```
pred Person_no_eleven_fifteen[self: Person]{
all r1, r2: self.roles |
((ELEVEN in r1.tasks) => (FIFTEEN !in r2.tasks)) &&
((FIFTEEN in r1.tasks) => (ELEVEN !in r2.tasks))}
```

6.3 Stage 3: Model Analysis

Alloy assertions must be formulated prior to analysis by Alloy Analyzer. Assertions are statements that capture properties we wish to verify. Alloy Analyzer automatically checks such assertions and if they fail it produces a counterexample. We have checked several assertions regarding the security properties of the example system. For example, it is crucial to ensure that no user can change VC protocols (task 11) at the same time as he has permission to change VC materials (task 15). To verify this, we create the following assertion:

```
assert NoConflictPermsSTVCAssigned{
all r: Person.roles, d: Time, l: Location|
((ELEVEN in r.tasks) && (d in r.timeCon) &&
(l in r.location)) =>
((FIFTEEN !in r.tasks) && (d in r.timeCon) &&
(l in r.location))}
```

We chose a value of 8 for the scope of analysis, and the assertion was checked for this scope. A scope of 8 means that the Alloy Analyzer will attempt find an instance that violates the assertion, using up to 8 instances for each of the entities defined in the class diagram of Figure 2. The assertion produced no counterexample, meaning that it is valid for the given scope.

Next, we will check whether the SoD for role permission assignment is maintained. To do this, we create the following assertion:

```
assert NoConflictPermsSTVC{
all r: StateVC, d: Time, l: Location|
((ELEVEN in r.tasks) && (d in r.timeCon) &&
(l in r.location)) =>
((FIFTEEN !in r.tasks) && (d in r.timeCon) &&
(l in r.location))}
```

We chose a value of 8 for the scope of this analysis as well. However, this time the analyzer showed the counterexample, which means these conflict permissions can be assigned to the same role. The counterexample is shown in Figure 3.

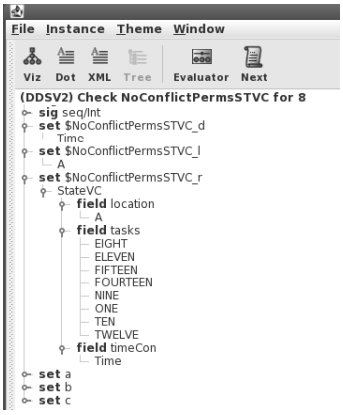

Figure 3: Counterexample for Assertion NoConflictPermsSTVC

7. CONCLUSION AND FUTURE WORK

Traditional access control models do not take into account environmental factors before making access decisions and may not be suitable for pervasive computing applications such as the Dengue

Decision Support system discussed in this paper. New access control models that allow access decisions to be made based on contextual information must be used for securing such applications. However, due to the complexity of the application and the access control model, we need assurance that the application is indeed adequately protected. Towards this end, we propose a methodology for protecting complex applications from access control breaches.

Since UML is the de facto standard used in the software industry, we use it for specifying the application and its access control requirements. The constraints are specified using OCL. However, not much tool support exists for automatically analyzing UML and OCL specifications. Towards this end, we show how the UML model can be converted automatically into an Alloy specification. The Alloy specification can be automatically analyzed using the Alloy Analyzer which has embedded SAT-solvers. The results of the analysis indicate how well the application is protected.

A lot of work remains to be done. One limitation of using SAT-solvers for the purpose of analysis is the size of the model that can be verified. Consequently, we are investigating how to further abstract the model resulting in the construction of smaller SAT formulas that can be efficiently verified. This, together with new research for improving SAT-solver technology, will alleviate the limitation mentioned above.

Since complex applications are typically modeled as workflows consisting of tasks that have dependencies among them, we need new spatio-temporal access control models for workflows. We will also need sophisticated analysis techniques to verify the complex spatio-temporal constraints that exist among tasks in workflows. Since Alloy is not suitable for expressing sophisticated temporal properties, we need to investigate other specification languages and tools for performing such analysis. It will also be interesting to investigate the interactions of workflow application constraints and authorization constraints.

Acknowledgement

This work was supported in part by AFOSR under contract number FA9550-07-1-0042. The authors thank Dr. Lars Eisen and Dr. Saul Lorenzo in the Department of Microbiology, Immunology and Pathology at Colorado State University for their help in formulating the access control requirements for the Dengue Decision Support system.

8. REFERENCES

[1] K. Anastasakis, B. Bordbar, G. Georg, and I. Ray. On Challenges of Model Transformation from UML to Alloy. *Journal on Software & System Modeling*, 2009. To appear.

[2] Kyriakos Anastasakis, Behzad Bordbar, Geri Georg, and Indrakshi Ray. UML2Alloy: A Challenging Model Transformation. In *Proceedings of the ACM/IEEE 10th International Conference on Model Driven Engineering Languages and Systems*, pages 436–450, Nashville, TN, USA, October 2007.

[3] Claudio A. Ardagna, Marco Cremonini, Ernesto Damiani, Sabrina De Capitani di Vimercati, and Pierangela Samarati. Supporting location-based conditions in access control policies. In *Proceedings of the ACM Symposium on Information, Computer and Communications Security*, pages 212–222, Taipei, Taiwan, March 2006.

[4] Elisa Bertino, Piero Andrea Bonatti, and Elena Ferrari. TRBAC: a temporal role-based access control model. In *Proceedings of the 5th ACM Workshop on Role-Based Access Control*, pages 21–30, Berlin, Germany, July 2000.

[5] Elisa Bertino, Barbara Catania, Maria Luisa Damiani, and Paolo Perlasca. GEO-RBAC: a spatially aware RBAC. In *Proceedings of the 10th ACM Symposium on Access Control Models and Technologies*, pages 29–37, Stockholm, Sweden, June 2005.

[6] Behzad Bordbar and Kyriakos Anastasakis. MDA and Analysis of Web Applications. In *Trends in Enterprise Application Architecture*, volume 3888 of *Lecture notes in Computer Science*, pages 44–55, Trondheim, Norway, August 2005.

[7] Behzad Bordbar and Kyriakos Anastasakis. UML2ALLOY: A tool for lightweight modelling of discrete event systems. In *Proceedings of the IADIS International Conference on Applied Computing*, pages 209–216, Algarve, Portugal, February 2005.

[8] Suroop Mohan Chandran and James B. D. Joshi. *LoT-RBAC*: A Location and Time-Based RBAC Model. In *Proceedings of the 6th International Conference on Web Information Systems Engineering*, pages 361–375, New York, NY, USA, November 2005.

[9] Michael J. Covington, Prahlad Fogla, Zhiyuan Zhan, and Mustaque Ahamad. A Context-Aware Security Architecture for Emerging Applications. In *Proceedings of the Annual Computer Security Applications Conference* , pages 249–260, Las Vegas, NV, USA, December 2002.

[10] Michael J. Covington, Wende Long, Srividhya Srinivasan, Anind Dey, Mustaque Ahamad, and Gregory Abowd. Securing Context-Aware Applications Using Environment Roles. In *Proceedings of the 6th ACM Symposium on Access Control Models and Technologies*, pages 10–20, Chantilly, VA, USA, May 2001.

[11] David F. Ferraiolo, Ravi S. Sandhu, Serban I. Gavrila, D. Richard Kuhn, and Ramaswamy Chandramouli. Proposed NIST Standard for Role-Based Access Control. *ACM Transactions on Information and Systems Security*, 4(3):224 – 274, August 2001.

[12] G. Georg, I. Ray, K. Anastasakis, B. Bordbar, M. Toahchoodee, and S. H. Houmb. An Aspect-oriented Methodology for Designing Secure Applications. *Information and Software Technology, Special Issue on Model-Driven Development for Secure Information Systems*, 51(5):846–864, May 2009.

[13] Geri Georg, James Bieman, and Robert B. France. Using Alloy and UML/OCL to Specify Run-Time Configuration Management: A Case Study. In Andy Evans, Robert France, Ana Moreira, and Bernhard Rumpe, editors, *Practical UML-Based Rigorous Development Methods - Countering or Integrating the eXtremists.*, volume P-7 of *LNI*, pages 128–141, 2001.

[14] Urs Hengartner and Peter Steenkiste. Implementing Access Control to People Location Information. In *Proceedings of the 9th ACM Symposium on Access Control Models and Technologies*, pages 11–20, Yorktown Heights, NY, USA, June 2004.

[15] Hongxin Hu and GailJoon Ahn. Enabling verification and conformance testing for access control model. In *Proceedings of the 13th ACM Symposium on Access control Models and Technologies*, pages 195–204, Estes Park, CO, USA, June 2008.

[16] Daniel Jackson. Automating first-order relational logic. In *Proceedings of the 8th ACM SIGSOFT international symposium on Foundations of Software Engineering*, pages

130–139, San Diego, CA, USA, November 2000.

[17] Daniel Jackson. *Micromodels of Software: Lightweight Modelling and Analysis with Alloy*. At http://alloy.mit.edu/alloy2website/reference-manual.pdf , 2002.

[18] Daniel Jackson. *Alloy 3.0 reference manual*. At http://alloy.mit.edu/reference-manual.pdf, 2004.

[19] Daniel Jackson. *Software Abstractions: Logic, Language, and Analysis*. MIT Press, 2006.

[20] Daniel Jackson. *Software Abstractions: Logic, Language, and Analysis*. The MIT Press, London, England, 2006.

[21] James B.D. Joshi, Elisa Bertino, Usman Latif, and Arif Ghafoor. A Generalized Temporal Role-Based Access Control Model. *IEEE Transactions on Knowledge and Data Engineering*, 17(1):4–23, January 2005.

[22] Anneke G. Kleppe, Jos Warmer, and Wim Bast. *MDA Explained: The Model Driven Architecture: Practice and Promise*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2003.

[23] Ulf Leonhardt and Jeff Magee. Security Consideration for a Distributed Location Service. *Imperial College of Science, Technology and Medicine, London, UK*, 1997.

[24] OMG. MOF Core v. 2.0. Document Id: formal/06-01-01. `http://www.omg.org`.

[25] OMG. OCL Version 2.0. Document id: formal/06-05-01. `http://www.omg.org`.

[26] OMG. UML: Superstructure. Version 2.0. Document id: formal/05-07-04. `http://www.omg.org`.

[27] OMG. *Unified Modeling Language: Superstructure Version 2.1.2 Formal/07/11/02*. At http://www.omg.org/docs/formal/07-11-02.pdf., 2002.

[28] Indrakshi Ray and Mahendra Kumar. Towards a Location-Based Mandatory Access Control Model. *Computers & Security*, 25(1), February 2006.

[29] Indrakshi Ray, Mahendra Kumar, and Lijun Yu. LRBAC: A Location-Aware Role-Based Access Control Model. In *Proceedings of the 2nd International Conference on Information Systems Security*, pages 147–161, Kolkata, India, December 2006.

[30] Indrakshi Ray, Na Li, Robert France, and Dae-Kyoo Kim. Using UML to Visualize Role-Based Access Control Constraints. In *Proceedings of the 9th ACM symposium on Access Control Models and Technologies*, pages 115–124, Yorktown Heights, NY, USA, June 2004.

[31] Indrakshi Ray and Manachai Toahchoodee. A Spatio-temporal Role-Based Access Control Model. In *Proceedings of the 21st Annual IFIP WG 11.3 Working Conference on Data and Applications Security*, pages 211–226, Redondo Beach, CA, July 2007.

[32] Indrakshi Ray and Manachai Toahchoodee. A Spatio-Temporal Access Control Model Supporting Delegation for Pervasive Computing Applications. In *Proceedings of the 5th International Conference on Trust, Privacy & Security in Digital Business*, pages 48–58, Turin, Italy, September 2008.

[33] Mark Richters. *A Precise Approach to Validating UML Models and OCL Constraints*. PhD thesis, Universitaet Bremen, 2002. Logos Verlag, Berlin, BISS Monographs, No. 14.

[34] Geetanjali Sampemane, Prasad Naldurg, and Roy H. Campbell. Access Control for Active Spaces. In *Proceedings of the Annual Computer Security Applications Conference* , pages 343–352, Las Vegas, NV, USA, December 2002.

[35] Arjmand Samuel, Arif Ghafoor, and Elisa Bertino. A Framework for Specification and Verification of Generalized Spatio-Temporal Role Based Access Control Model. Technical report, Purdue University, February 2007. CERIAS TR 2007-08.

[36] Ravi S. Sandhu, Edward J. Coyne, Hal L. Feinstein, and Charles E. Youman. Role-based access control models. *IEEE Computer*, 29(2):38–47, February 1996.

[37] Andreas Schaad and Jonathan D. Moffett. A Lightweight Approach to Specification and Analysis of Role-Based Access Control Extensions. In *Proceedings of the 7th ACM Symposium on Access Control Models and Technologies*, pages 13–22, Monterey, CA, USA, June 2002.

[38] Richard Simon and Mary Ellen Zurko. Separation of Duty in Role-based Environments. In *Proceedings of the 10th Computer Security Foundations Workshop*, pages 183–194, Rockport, MA, USA, June 1997.

[39] Mana Taghdiri and Daniel Jackson. A lightweight formal analysis of a multicast key management scheme. In *Formal Techniques for Networked and Distributed Systems - FORTE 2 003*, volume 2767 of *Lecture Notes in Computer Science*, pages 240–256, 2003.

[40] Manachai Toahchoodee and Indrakshi Ray. On the Formal Analysis of a Spatio-Temporal Role-Based Access Control Model. In *Proceedings of the 22nd Annual IFIP WG 11.3 Working Conference on Data and Applications Security*, pages 17–32, London, U.K., July 2008.

[41] Manachai Toahchoodee and Indrakshi Ray. Using Alloy to Analyze a Spatio-Temporal Access Control Model Supporting Delegation. *IET Information Security*, 2009. To appear.

[42] Chunyang Yuan, Yeping He, Jianbo He, and Zhouyi Zhou. A Verifiable Formal Specification for RBAC Model with Constraints of Separation of Duty. In *Proceedings of the 2nd SKLOIS Conference on Information Security and Cryptology*, pages 196–210, Beijing, China, November 2006.

[43] John Zao, Hoetech Wee, Jonathan Chu, and Daniel Jackson. *RBAC Schema Verification Using Lightweight Formal Model and Constraint Analysis*. At http://alloy.mit.edu/publications.php, 2002.