CERIAS Tech Report 2011-19

Specification and Verification of a Context-Based Access Control Framework for Cyber Physical Systems

by Arjmand Samuel, Hammad Haseeb, Arif Ghafoor and Elisa Bertino
Center for Education and Research
Information Assurance and Security
Purdue University, West Lafayette, IN 47907-2086

Specification and Verification of a Context-Based Access Control Framework for Cyber Physical Systems

Arjmand Samuel, Hammad Haseeb, Arif Ghafoor and Elisa Bertino

Abstract

Cyber Physical Systems (CPS) are complex systems that operate in a dynamic environment where security characteristics of contexts are unique, and uniform access to secure resources anywhere anytime to mobile entities poses daunting challenges. To capture context parameters such as location and time in an access control policy for CPS, we propose a Generalized Spatio- Temporal RBAC (GST-RBAC) model. In this model spatial and temporal constraints are defined for role enabling, user-role assignment, role-permission assignment, role activation, separation of duty and role hierarchy. The inclusion of multiple types of constraints exposes the need of composing a policy which is verifiable for consistency. The second contribution in this paper is GST-RBAC policy specification and verification framework using light weight formal modeling language, Alloy. The analysis assists in consistency verification leading to conflict free composition of the actual policy for implementation for CPS.

Index terms*:* context-aware access control, Cyber Physical Systems, spatio-temporal constraints, role hierarchy, policy verification

1 Introduction

Cyber-Physical Systems (CPS) are complex dynamic systems characterized by tight coordination and linkage among computational, physical and virtual resources [Poo10,Wol09]. These systems have wide range applications ranging from nano-world to large scale distributed wide area systems such as critical infrastructure monitoring, health care [Lee10, Nit09], intelligent transportation system, smart structures, and cyber-enabled manufacturing process control [Lee08]. The positive economic impact of CPS is envisioned to be enormous. However, the security notion related to CPS is still in infancy and needs to be addressed before its wide spread applicability. In this paper, we focus on the access control challenge which is equally critical among the numerous other security challenges facing CPS, including authentication, reliability, malicious and denial of service attacks [Bar10,Mcd09,Mit11,Ten08,Zim10].

Protecting critical resources from unauthorized entities (users, devices, sub-systems) represents one of the most important security concerns for CPS. Secure access to such resources is critical for CPS since it allows interaction among numerous cyber and physical entities. Access control challenge is further exacerbated in CPS since physical and cyber entities are dispersed over a wide geographical area and across different sub-systems. Failure in providing

*This research is in part supported by an NSF grant number: IIS - 09646391

secure access can lead to detrimental loss to both the controlled physical system and users. Security policies and mechanisms developed for traditional information systems for secure access are inadequate for CPS for several reasons. First, CPS operate in an unpredictable and dynamic environment where security characteristics are generally context dependent, and therefore uniform access to secure and privileged information/physical resources anywhere and anytime by mobile entities may not be realistic. Secondly, CPS consist of numerous collaborating geographically distributed entities in which traditional access control mechanisms can result in substantial administration overhead. Thirdly, a pressing requirement for enforcing robust access control is that access control policies should be conflict free and conform to predefined safety and liveliness properties [Alp84,Bur90]. The challenge here is to identify conflicts within the overall policy to avoid security risks associated with unauthorized access.

A motivating example in this regard is a smart home-based health environment for remote health monitoring and providing assisted living for senior citizens. A smart home consists of miniature medical sensors monitoring and reporting health data to remote centers, such as hospitals. These homes can be dispersed over a vast geographical area. Patient information received from smart homes is analyzed and medical devices such as drug infusion pump are controlled by doctors or nurses remotely. Unauthorized access to such devices or underlying resources may result in physical harm to patients. In addition, medical devices, users within a hospital environment along with hospital's mobile paramedic units are part of CPS [Nit09].

In such scenarios, contexts of users/devices pose multifaceted security challenges which include heterogeneity, scalability, mobility and overall verifiability of underlying security framework. Heterogeneity implies specialized functionality of devices while, scalability deals with the provisioning of security for large number of users and devices dispersed over the region covered by the CPS. In addition, the security framework should be verifiable. Malicious codes in autonomous devices could be used to get unauthorized access to underlying CPS resources. The challenge in access control thus has a broad scope encompassing all types of CPS.

The aforementioned examples highlight the need to take contextual parameters such as location or time into account and accordingly assign privileges to devices or users in access control policy. In this paper, a new access control model, namely, Generalized Spatio-Temporal Role Based Access Control (GST-RBAC) is proposed, especially to address the aforementioned

access control challenges in CPS. GST-RBAC takes into account the environmental contexts, such as location and time to provide a comprehensive and generalized approach to security management in CPS. The main contributions of this paper are summarized as follows:

1. CPS involves interactions among a large number of entities that are geographically distributed across different organizational boundaries and hence require environmental contexts, such as location and time to play a crucial role in access decisions. In this regard, the proposed GST-RBAC model provides a formal framework for composition of complex spatio-temporal constraints exploiting topological relationship among physical locations. We also present access control ramification of combining spatial and temporal constraints for role enabling, user-role assignment, role-permission assignment and activation of roles. The model is generic and is applicable to develop an access control policy for any CPS.

2. Inclusion of multiple constraints in the access control policy exposes the need of verifying the policy for consistency. The challenge in this regard is to develop a consistent access control policy by identifying conflicts which can cause unauthorized access. We develop a specification model for GST-RBAC policy and outline methodology for inconsistency identification. Our approach is two tiered. Firstly, we capture the GST-RBAC policy requirements using a light-weight formal model. For this purpose we use the Alloy [Jac03] specification language and the accompanying constraint analyzer to verify the specification model. Subsequently, the GST-RBAC is analyzed for identifying conflicts and inconsistencies. The objective is to provide a formal framework to the policy administrator to formally compose GST-RBAC policy and verify policy composition prior to its implementation.

The paper is organized as follows. In Section 2, we present the related work. In Section 3, we present the GST-RBAC model, and discuss various spatio-temporal constraints and their execution semantics. Section 4 provides a detailed example of GST-RBAC model for a CPS environment. We present a specification model for GST-RBAC policy in Section 5 for analysis of policy for identifying conflicts using a lightweight formalism. Finally, in Section 6, we provide our conclusions and future work.

2 Related Work:

RBAC a de-facto model for specifying security requirements of large organizations, can be directly applicable to CPS by extending it with location and time. RBAC model consists of

four basic components [Jos05]: a set of users/devices, a set of roles, a set of permissions, and a set of sessions. A user can be a human being, an autonomous agent, a task or a cyber-physical device. A role is a collection of permissions needed to perform a certain job function within an organization. Permission is an access mode that can be exercised on objects in the system, and a session relates a user to possibly many roles. The RBAC model differentiates itself from traditional access control models in that the permissions in RBAC are not directly associated with users, but with roles. Roles are created by the security administrators to reflect the various functional categories of users within an enterprise. Users are then assigned membership to roles, and these roles are in turn assigned permissions. Such grouping provides a scalable mechanism, which is major advantage of RBAC over other authorization approaches [Jos05]. Also, RBAC is distinguished by its inherent support for principle of least privilege which requires a user to be given no more privileges than necessary to perform a task.

RBAC model has been widely investigated and its several extensions have been proposed due to its relevance and the abovementioned benefits. An initial temporal extension to RBAC has been proposed in the generalized temporal RBAC (GTRBAC) model [Jos05]. This has been motivated by the fact that in many organizations, functions may have limited or periodic temporal duration. Spatial extensions have also been proposed to RBAC. One such extension, GEO-RBAC [Ber05] defines spatial roles, which can be assumed within a defined spatial extent. Spatial information with regard to a role is represented as a role schema. GEO-RBAC defines one spatial extent for each role implying that each spatial location should have its own role. Therefore, this approach is not scalable. On the other hand, our proposed GST-RBAC model allows definition of semantic relationship between locations in the form of spatial constraints. This approach results in de-linking of the number of roles from the number of locations in CPS, thus provides a scalable solution. Further, GST-RBAC allows spatial information to express as spatial constraints which can be attached to any role already existing in an access control policy. Our approach is close to the one given in [Bha06], in which constraints on role activation are specified using an XML based grammar. However, the approach in [Bha06] does not exploit the relationship between locations and roles based on spatial relations.

A pressing requirement for enforcing robust access control is that access control policies be conflict free and conform to predefined safety and liveness properties [Alp84,Bur90]. The challenge in this regard is to be able to model an access control policy and to pinpoint conflicts which can cause

unauthorized access. Since CPS can be a system of systems, an unauthorized access resulting from conflicts can cause harmful cascading consequences on underlying resources and users. This challenge is further exacerbated by context-aware access control policies resulting in possible rule explosion [Ahm03]. Conflicts might occur due to the inclusion of multiple constraints such as spatial and time in access control policies. Therefore, a framework to identify conflicts in a policy is needed. Such a framework is proposed in Section 5.

3. Proposed GST-RBAC Access control Model

In this section, we present the GST-RBAC model. The model uses a formal representation theory of topological relations among the locations constituting CPS.

3.1 Characterizing Location in CPS

Location is traditionally defined as a place or site in physical space expressed relative to the position of another location. Location can be represented either using symbols such as name of places (e.g. City Hall, Times Square) or function of places (e.g. public parking, school, university etc.) or through geometric representations such as latitude, and longitude.

3.1.1 Location Sensing

An overview of context discovery, especially location sensing is given in [Leo98]. Accurately and reliably sensing location context of a user or a physical device is at the core of developing the location aware access control mechanisms. No single location-sensing technology is expected to become dominant due to the numerous dimensions along which location-sensing mechanisms can vary [Hig01]. Examples include indoor vs. outdoor use, accuracy, precision, energy usage, and the extent to which there is potential loss of privacy for users of the technology. Thus, the choice of location-sensing technology is influenced by the usage context, and various technologies can coexist.

Known location techniques can be divided into two broad categories, namely: outdoors and indoors. Well-known GPS techniques provide an inexpensive but accurate means of acquiring longitude, latitude and altitude. Generally, GPS signals cannot be received indoors, several indoor location sensing techniques have been proposed including Olivetti Active Badge System [Wan93], Xerox ParcTab [Sch93] and the Cyberguide project [Abo97].

Multiple location sensing techniques using technologies such as GPS, smart phones, and wifi. can be used to accurately capture the location of accessing entities in a CPS. Car Tel project

[Hull06], a massive vehicular cyber system uses in-car devices such as GPS phone and other custom built on board telematics for location sensing. Today, location aware services are possible due to the widespread availability of smart phones equipped with GPS.

Our current work builds on the location sensing research conducted by the community and assumes the availability of location parameter with the desired resolution and representation. In this regard, the location acquisition and integration framework proposed by [Leo98] provides a robust mechanism for our model.

3.1.2 Location Based Representation of Access Control Constraints in CPS

Formally, the overall topological configuration of locations in CPS can be represented as a directed graph, which we term as *topological profile* of the CPS. For this purpose, we define locations in a CPS as either a set of symbols or coordinates of physical space. Given *n* locations in a CPS, we can represent these locations symbolically by the set $L_s = \{l_1, l_2,..l_n\}$. For example, for a healthcare CPS, symbolic locations can represent various hospital wards, its floors, names of places by their functions (e.g. emergency rooms, trauma center, surgical rooms etc.), and other remote locations which are part of CPS such as remotely connected smart homes for assisted-living patients. We present the following definitions to describe *topological profile.*

Definition 3.1: Distinct topological relations can exist between two locations, which can be represented by a set T= {*d, m, o, e, ct, i, cv, cB*}, where each member represents relation: disjoint, meet, overlap, equal, contains, inside, covers and coveredBy, respectively [Ege91]. We define $\subseteq p$ as a *topologic operator* describing semantic relationship between symbolic locations where *p* is a topological relationship between symbolic locations. Note, $p \in T$.

Definition 3.2: Based on topological relations among members of L_s, the *topological profile* of a CPS can be represented by an edge labeled directed graph G = (V, E), which is called Spatial Symbolic Graph (SSG). Here V is a set of symbolic locations representing set L_s and E is the edge set defined as $E = \{(v_i,v_j,l_k) \mid v_i, v_j \in V, l_k \in L_s\}$. Note, $\subseteq p$ represents an edge of the SSG graph with the end nodes related through the topological relation *p*.

Definition 3.3: We define *locale* in a CPS as a region of interest (especially for the purpose of providing access control privileges) that has a semantic connotation. Examples of *locales* can be “Surgical Quadrangle” which may consist of all the rooms connected to the surgical ward in hospital, or “all the remote parking lots, i.e. the ones not adjacent to the main

office building". We assume that each *locale* has a root location that serves as the reference point for extending its access control privileges to the other topological members that fit within the semantic connotation of the *locale*. For example, the definition of "Surgical Quadrangle" *locale* can imply all the rooms adjacent to the Surgery Room (SR) in a hospital. Assuming, SR is the root node of this *locale*, its access control privileges can be extended (fully or partially) to all the rooms adjacent to it.

Definition 3.3 and its significance to access privileges raise two issues that need to be addressed. First, how *locales* can be formally described, and second, how this definition can be used to identify the locations which get the extended access privileges? For this purpose, we use description logic [Haa94,Rou10] and formally define a *locale* as: $\{\underline{x} \mid P(\underline{x}) \cap \underline{x} \in L_s\}$ where $\underline{x}$ is a vector of variables and $P(\underline{x})$ is a predicate that $\underline{x}$ needs to satisfy. In general, multiple variables may be required to define a *locale*. It can be noted that the *locale* (SQ) for "Surgical Quadrangle" can be defined as SQ: $\{\forall x \mid (x \subseteq m \text{ SR}) \cap (x \in L_s)\} \cup \{\text{SR}\}$

Definition 3.4: Predicate $P(\underline{x})$ in Definition 3.3 is termed as the spatial constraint (SpC).

Additional examples of *locales* and corresponding SpCs are given in Section 4.

The root node in each *locale* is assigned role(s) and privileges by the authorization administrator of the CPS. SpC plays a crucial role in granting an access request. At the time an access request is received from a location in CPS, if the requesting entity (user/device) is from a location which is within the extent of a *locale*, the related root node is identified in order to grant appropriate privileges (all or some, depending on the role of the requester) to the requester. To check the extent of a *locale* and determining the inclusion of the requesting location in this extent, *locale* 's SpC is verified by consulting SSG of the CPS and by assigning values to the variable(s) x from set L_s.

Note, there is many-to-many relation between access control roles and *locales*. Also, SpC can be applied in situations where some restrictions to roles and permissions to users in some *locales* is desirable.

Advantages of *locales* and SSG for Access Control: Arbitrary *locale* can be defined by formulation of complex SpC by compounding existing SpC expressions thus providing a scalable solution to address the location based access control challenges in CPS. In essence, defining

locales and capturing topological profile of a CPS via SSG for making access control decisions offer the following advantages:

- Scalability: The concept of *locale* allows grouping of spatial locations in a meaningful way so that access control decisions do not have to be defined for each constituent location, but rather can be specified for regions of interest of several symbolic locations. This addresses a serious shortcoming in GEO-RBAC [Ber05] which implies that each spatial location in an organization needs to have its own role.
- Uniqueness: SSG can effectively be used to compose unique access control *locales* using symbolic locations. Generally, symbolic locations tend not to be unique since multiple locations can be represented using the same symbolic name. For example, the label *school* may represent many locations in a city. However, SSG uniquely identifies each location by specifying the topological relation among locations resulting in precisely identifying the requisite location.

3.2 The Generalized Spatio-Temporal Access Control Model

The proposed GST-RBAC model allows specification of the following type of constraints:

- Spatial constraint SpC on role enabling, user-role assignment, role-permission assignment and activation. Note, here users imply human beings, devices and other sub-systems of CPS.
- Temporal constraints on role enabling, user-role assignment, and role-permission assignments, activation, runtime events, constraint enabling expressions and triggers as mentioned in detail in [Jos05]. We denote these constraints as TempC.

Given a set of users *U*, a set of roles *R*, and a set of permissions *P*, the general form of the temporal and spatial constraint in GST-RBAC expression is given as follows:

$$(([\neg][\text{TempC}][\alpha][\neg][\text{SpC}]), [pr], \mathit{Function}, [u], r)$$

where, TempC is the temporal constraint [Jos05], α is an Boolean operator $\in$ ($\cap$, U), SpC is the spatial constraint and $\mathit{Function} \in \{\text{enable, disable, assign}_u, \text{deassign}_u, \text{assign}_p, \text{deassign}_p, \text{activate}_r, \text{deactivate}_r\}$. Details of each function is depicted Table3.1. In addition, $u \in U$, $r \in R$ and $pr \in P$. Note, use of permission *pr* and user *u* in the expression depends on the type of function. It can be noticed that both TempC and SpC constraints are optional to allow provision for those requests which are independent of time and/or location. An example is mobile paramedic units in a CPS

environment for which SpC may not be specified. In such cases a NULL value can be used in the aforementioned expression. The *Function* in the GST-RBAC expression is executed if ([TempC]α[SpC]) is evaluated to true for a given request. An optional negation operator (¬) can also be used to signify that function *Function* be executed if the TempC and SpC are not true.

Table 3.1 Summary of Constraint Expressions

Constraint Categories	Constraints	Expression
Spatial and Temporal constraints on role enabling, user-role , role-permission assignments and role activation	*User-role assignment*	*(TempC α SpC, assign$_U$/deassign$_U$ r to u)*
	Role enabling	*(TempC α SpC), enable/disable r)*
	Role-permission assignment	*(TempC α SpC, assign$_P$/deassign$_P$ p to r)*
	Role activation	*(TempC α,SpC,, activate$_r$/deactivate$_r$ r)*
Temporal Constraint	*TempC*	*Temporal constraint defined in GTRBAC [Jos05]*
Spatial Constraint	*SpC*	*$((L_1 \sqsubseteq (p)\ L_2) \cup \ldots (L_{n-1} \sqsubseteq (p)\ L_n)$ where $L_1 \ldots L_n$ are symbolic locations $\in L$. $\sqsubseteq$ is the spatial hierarchical relationship between locations and $p \in$ (d,m,o,e,ct,I,cv,cB)*
Constraint Enabling	*enable/disable temporal constraint c where c ∈{TempC}* *enable/disable spatial constraint k where k ∈{SpC}*	

Table 3.1 summarizes the constraint types and expressions of the GST-RBAC model. Basic event expressions used by the GST-RBAC constraints are depicted in Table 3.2. Status predicates, listed in Table 3.2, are used to capture the state information associated with roles. The expressions defined for temporal and spatial constraints for role enabling, user-role and role-permission assignments and activation, are denoted by TempC and SpC, respectively.

Table 3.2 Events and Status Predicates

Simple Event (r ∈ Roles, u ∈ Users, and p ∈ Permissions)	Status Predicate (C)	Status Predicate with time and spatial constraint	Semantics [for time]
enable r or disable r	*enable(r)*	*enable(r, t,SpC)*	*r is enabled [at time t] satisfying SpC*
assign$_U$ r to u or de-assign$_U$ r to u	*u_assigned(u, r)*	*u_assigned(u, r, t,SpC)*	*u is assigned to r [at time t] satisfying SpC*
Assign$_P$ p to r or de-assign$_P$ p to r	*p_assigned(p, r)*	*p_assigned(p, r, t,SpC)*	*p is assigned to r [at time t] satisfying SpC*
enable c or disable c, (where c is a Temporal Constraint TempC)	*active(r)*	*active(r, t,SpC)*	*r is active [at time t] in at least one session satisfying SpC*
enable k or disable k, (where k is a Spatial Constraint SpC)	*u_active(u, r)*	*u_active(u, r, t,SpC)*	*r is active in a u's session [at time t] satisfying SpC*
	s_active(u, r, s)	*s_active(u, r, s, t,SpC)*	*r is active in a u's session s [at time t] satisfying SpC*
	acquires(u, p)	*acquires(u, p, t,SpC)*	*u acquires p [at time t] satisfying SpC*

3.2.2 Temporal Constraints in GST-RBAC

Temporal and spatial constraints for fine grained access control exploiting both time and location, respectively. The following TempCs are provided in GTRBAC model [Jos05]:

1. *Temporal constraints on role enabling/disabling*: These constraints allow the specification of intervals and durations in which a role is enabled. When a role is enabled, the permissions assigned to it can be acquired by a user by activating it. When a duration constraint is specified, the enabling/disabling of a role is initiated by a constraint-enabling event that results from the firing of a trigger or through an administrator-initiated run-time event.

2. *Temporal constraints on user-role and role-permission assignments*: These constraints allow specifying intervals and durations in which a user or permission is assigned to a role.

3. *Activation constraints*: These constraints allow specification of restrictions on the activation of a role. These include, for example, specifying the total duration for which a user may activate a role or the number of concurrent activations of a role at a particular time.

4. *Run-time events*: A set of run-time events allows an administrator to dynamically initiate GTRBAC events or enable duration or activation constraints. Another set of run-time events allow users to request activation or deactivation of a role.

5. *Constraint-enabling expressions*: The GTRBAC model includes events that enable or disable duration and role-activation constraints mentioned earlier.

5. *Triggers*: The GTRBAC triggers allow expressing dependencies among events.

3.2.3 Role Enabling in GST-RBAC

The proposed GST-RBAC model distinguishes between the notions of role enabling and role activation as in [Jos05]. A role can assume one of the three states: *disabled*, *enabled* and *active*. The *disabled* state indicates that the role cannot be used in any user session, i.e., a user cannot acquire the permissions associated with the role. A role in the *disabled* state can be enabled. The *enabled* state indicates that users who are authorized to use the role at the time and location may activate the role. Subsequently, if a user activates the role, the state of the role becomes *active*. A role in the *active* state implies that there is at least one user who has activated the role. Once in the *active* state, re-activation of the role does not change its state. When a role is in the *active* state, upon deactivation, the role transitions to the *enabled* state provided there is

only one session in which it is *active*; otherwise the role remains in the *active* state. A role in the *enabled* or *active* state transitions to the *disabled* state if a disabling event occurs.

Role enabling constraint in GST-RBAC defines the spatial relationship between locations and time for which the role can be enabled. A role is enabled at a certain location and time, while it is not enabled at other locations or other times

3.2.4 User-role and Role-permission Assignment in GST-RBAC

The *user role assignment* (UA) and the *role permission assignment* (PA) functions model the assignment of users to roles and the assignment of permissions to roles, respectively. When spatial and temporal constraints are evaluated, user to roles and permissions to roles assignments take place depending on location and time.

Note, spatial and temporal constraints may be applied in some situations where restriction to roles and permissions to users in some *locales* is desirable.

3.2.5 Role Activation in GST-RBAC

Role activation requests are made at the discretion of a user at arbitrary times and locations thus requiring the need to restrict activation at certain *locales* and time inorder to protect against unauthorized access.

In GST-RBAC, a user activates a role by sending an activation request. The request, along with other components such as the user/device id, also contains the location parameter l_1 of the user. If the context of the end user (TempC and SpC) is valid, the user can activate/deactivate role *r* at location l_1.

3.2.6 Separation of Duty Constraints in GST-RBAC

In order to curtail conflicts of interest in a role based system, RBAC offers Separation of Duty (SoD) constraint which does not allow a user to be activated to conflicting roles. The roles involved are mutually exclusive to the user. SoDs contain desirable restrictions for avoiding possible fraud that users may commit by carrying out conflicting activities [Jos05]. We extend this concept to include spatial constraints on SoD resulting in mutual exclusiveness of roles being dictated by the location of the user. We define 3 types of SoD constraints for GST-RBAC with their types and details given below:

1. Static SoD: SoD defined with only temporal constraints in GTRBAC model [Jos05].
2. Simple Spatial SoD (SSoD): SoD defined for only spatial constraints.

3. Spatio-temporal SoD (SpTSSoD): SoD defined taking spatial and/or temporal constraints into consideration while enabling or activating roles.

SoD in GST-RBAC are defined in Table 3.3 for some arbitrary *SpC, and* $TempC_1$.

Table 3.3 Separation of duty constraints in GST-RBAC

Separation of duty type	**Notation**	**Condition**
Static SoD	$SSoD(u, r_1, r_2)$, $u \in U$, $r_1, r_2 \in R$	$(assign_u, u, r_1) \rightarrow \neg(assign_u, u, r_2)$, $u \in U$, $r_1, r_2 \in R$, $assign_u \in Function$
Simple Spatial SSoD	$SpSSoD(SpC_1, u, r_1, r_2)$, $u \in U$, $r_1, r_2 \in R$,	$(SpC_1, assign_u, u, r_1) \rightarrow \neg(SpC_1, assign_u, u, r_2)$, $u \in U$, $r_1, r_2 \in R$, $assign_u \in Function$,
Spatio-temporal SSoD (Type I)	$SpTSSoD_I(SpC_1, TempC_1, u, r_1, r_2)$, $u \in U$, $r_1, r_2 \in R$,	$(SpC_1 \cap TempC_1, assign_u, u, r_1) \rightarrow \neg(SpC_1 \cap TempC_1, assign_u, u, r_2)$, $u \in U$, $r_1, r_2 \in R$, $assign_u \in Function$
Spatio-temporal SSoD (Type II)	$SpTSSoD_{II}(SpC_1, TempC_1, u, r_1, r_2)$, $u \in U$, $r_1, r_2 \in R$,	$(SpC_1 \cap TempC_1, assign_u, u, r_1) \rightarrow \neg(SpC_1 \cap TempC_1, assign_u, u, r_2)$, $u \in U$, $r_1, r_2 \in R$, $assign_u \in Function$

3.2.7 Role Hierarchy in GST-RBAC

Critical events may arise in CPS due to natural or man-made disasters requiring policies governing access under normal situations be quickly adapted to allow timely access to resources for crisis management. Protection requirements for critical sources shift in favor of more readily accessible resources to authenticated users at possibly different times and locations. For example a physician authorized to access hospital information from 9 AM to 5 PM in normal circumstances may now be authorized for 24 hours access, without any interruption. Similar is the case of location, where a nurse may be authorized to access records of a patient while at the nursing station only; however, in case of an emergency, the nurse may be allowed to access the same records/ resources throughout the out-patient wing of the hospital.

Role hierarchy plays a crucial role adaptation of policy in GST-RBAC model to deal with crisis situation for CPS. A role hierarchy defines permission acquisition and role-activation semantics through role–role relationships. It can be utilized for efficiently and effectively structuring functional roles of an organization having related access-control needs. RBAC offers role hierarchy as a means for roles to inherit permissions and users from each other [Jos05]. Role r_1 inherits role r_2 (expressed as $r_1 \geq r_2$) if all permissions of r_2 are also permissions of r_1. In this case r_1 acquires all users of r_2. Spatial constraints allow us to define spatial role hierarchies

between roles activated from disparate physical locations. We define the following three categories of hierarchies.

Spatially unrestricted role hierarchy: Spatial constraints do not have any bearing on the semantics of role hierarchy and is similar to the one defined in [San00].

Spatially restricted role hierarchy manifests itself in the form of a role hierarchy when a user activates a role from one location; while on the other hand, no hierarchy exists when the same role is activated from another location. Formally, $r_1, r_2 \in R$ $r_1 \geq_{SpC1} r_2$ which implies that when a user activates role r_1 while in the *locale* associated with SpC_1, the permissions of role r_2 are available to him. However, when the same user activates role r_1 from a *locale* not associated with SpC_1, the hierarchical relationship does not hold.

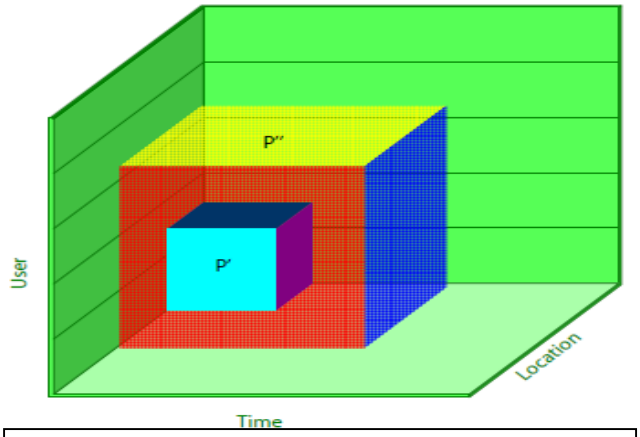


Figure 3.1 Privilege envelope expands from P` to P`` during crisis management by GST-RBAC role hierarchy

Figure 3.1 depicts the impact of role hierarchies in GST-RBAC that expand the *privilege envelope* in case of an emergency, compared to the smaller one before the emergency. Role hierarchies in GST-RBAC address crisis management viz-a-viz access control in CPS by:

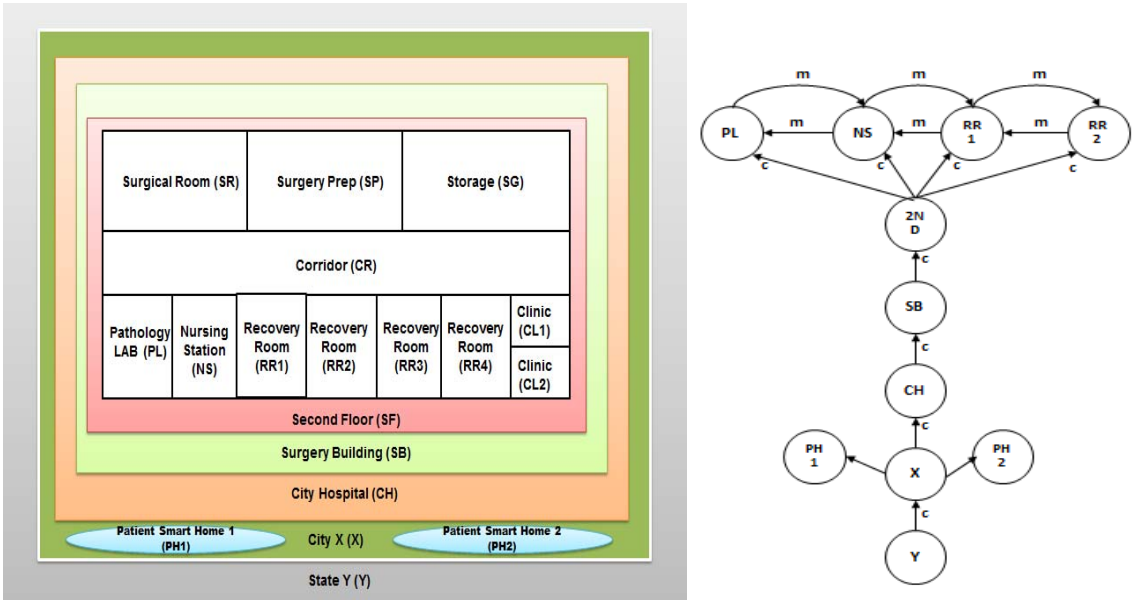
1. Enabling or disabling a role hierarchy in a crisis, results in all subordinate roles (in the hierarchy) to be enabled or disabled, respectively. This mechanism allows escalation or revocation of privileges in a crisis
2. Adaptation of time based role hierarchy allows roles to inherit privileges at a different time of the day, not allowed previously, and for longer (or shorter) duration.
3. Adaptation of spatial role hierarchy allows roles to inherit privileges at *locales* not previously allowed.

We observe that definition of spatially restricted role hierarchy may entail additional administrative overhead in terms of defining different role hierarchies for different *locales*. Therefore, this overhead is may not be significant as the number of *locales* can be far less than the number locations, providing a scalability advantage over GEO-RBAC model [Ber05]. In

addition, adaptation of role hierarchy may only be desirable for some *locales* depending upon the significance of their associated roles during crisis management.

4. An Example of GST-RBAC Access Control Policy in CPS

In this section, we present an example to illustrate the application and the relevance of spatial temporal constraints for the proposed GST-RBAC in a CPS environment. In the subsequent section, we use this example to illustrate spatio-temporal policy modeling for verification. We consider an access control policy for a CPS environment consisting of a city hospital connected to several patient smart homes. The smart homes consist of medical sensors and devices that monitor the patients' health and send the pertinent information to the hospital. The information is analyzed at the hospital and update regarding treatment is relayed back to the devices in smart homes. The Electronic Health Record policy of the hospital consists of both temporal and spatial constraints. In order to illustrate composition of spatial constraints for the access control policy, we consider a simplified layout of the CPS depicted in Figure 4.1 (a).



(a) (b)

Figure 4.1 (a) Spatial map of CPS (b) SSG of Pathology Lab (PL), Nursing Station (NS), Recovery Room 1 (RR1) and 2 (RR2), Patient Smart Home 1 (PH1) and Patient Smart Home 2 (PH2).

The layout includes the hospital and its geographical location within a city and state. Figure 4.2(b) depicts SSG of the Pathology Lab (PL), Nursing Station (NS), Recovery Room 1(RR1), Room 2(RR2) and two patient smart homes formally represented as:

$V = \{PL, NS, RR1, RR2, 2ND, SB, CH, PH1, PH2, X, Y\}$

$$E_1 = \{(PL \subseteq_m NS), (NS \subseteq_m PL), (NS \subseteq_m RR1), (RR1 \subseteq_m NS), (RR1 \subseteq_m RR2),$$
$$(RR2 \subseteq_m RR1), (2ND \subseteq_c PL), (2ND \subseteq_c NS), (2ND \subseteq_c RR1), (2ND \subseteq_c RR2),$$
$$(SB \subseteq_c 2ND), (CH \subseteq_c SB), (X \subseteq_c CH), (X \subseteq_c PH1), (X \subseteq_c PH2)(Y \subseteq_c C)\}$$

Note, the meet (m) relationship exists between locations *PL* and *NS* which is the label of the edge connecting the two nodes. Also, these locations are contained (c) within the Second Floor (*2ND*), which in turn is contained (c) in the Surgery Building (*SB*). The smart homes *PH1* and *PH2* are located inside the city *X*. The SSG can be composed so as to include the city as well as the state.

Table 4.1 SpC in CPS

Spatial Constraint	Notion	Semantic Definition of Locale	Root Node (partial or all privileges)
SpC1	*CH ⊆ (c)SB ⊆ (c)2ND ⊆ (c) SR ⊆ (m)SP*	*Surgical Room and surgeon prep area located on the second floor*	*SR on the second floor*
SpC2	*2ND ⊆(c) NS1 ⊆ (m)RR1 ⊆ (m) RR2 ⊆ (m) RR3 ⊆(m)RR4*	*Nursing station and the recovery rooms on the second floor of the building*	*NS1 or any RR room*
SpC3	*SB ⊆ (c)2ND ⊆ (c) SR ⊆ (m)SP)∪(2ND ⊆(c)RR1 ⊆(m) RR2 ⊆ (m) RR3 ⊆(m) RR4)∪(2ND ⊆(c) CL1 ⊆(m) CL2)*	*Spatial location spanning the surgeon prep area, the recovery rooms and the two clinics*	*SB*
SpC4	*CH ⊆ (c)SB ⊆ (c)2ND ⊆ (c) (SR) ∪ (NS ⊆ (m)PL))*	*Spatial location spanning the operation theater and nursing station which meets the pathology lab*	*Any location can be selected*
SpC5	*X⊆(c)PH1 ∪ X⊆(c) PH2 ∪ X⊆(c) CH*	*Smart homes PH1,PH2 and Hospital CH in City X*	*CH*

In Table 4.1, we list five hypothetical *locales*, along with their associated SpC's and root nodes. Note, that constraint composition in this example is a semantic relationship of symbolic locations up to the city hospital and can be extended to the whole city and the state. This allows a succinct definition of spatial context in the presence of heterogeneous symbolic locations (e.g more than one surgical wards in a hospital). The spatial constraint SpC3, describes the relationship between the surgery building and the second floor of the building. In addition, second floor of the building is further related to the surgeon prep room, the recovery rooms and clinics. The relationships among these locations are also represented as part of the constraint. At the time an access control decisions is made based on SpC3 constraint, a role defined for any one

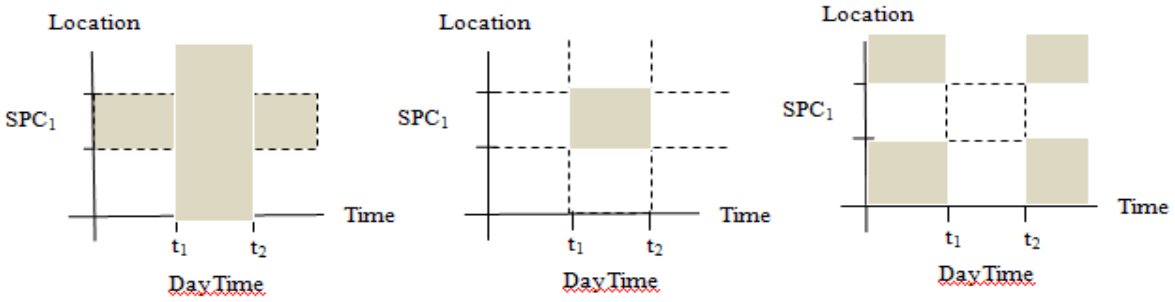
constituent location can be activated/de-activated or assigned/de-assigned from all other locations part of SpC3 constraint. Note, the privileges (all or partial) of root node are extended to all the components of its *locale.*

A hypothetical policy describing roles in the access control and the relevant semantics is described in Table 4.2. The policy consists of constraints on role enabling, activation, separation of duy and role hierarchy. NightTime and DayTime are temporal constraints [Jos05] of Type TempC that specify constraints on Time. Roles in GST-RBAC are enabled or activated based on the GST-RBAC expression. For example, consider the following GST-RBAC expression for role Day Surgeon, (DayTime ∨ SpC1 , enable, DaySurgeon)

In this expression, DayTime is a duration expression [Jos05] and SpC1 is the spatial constraint defined by an SSG that consists of Surgical Ward and surgeon prep area located on the second floor. The logical ∨ operator signifies that role DaySurgeon is enabled if DayTime or SpC1 are satisfied for a request. Figure 4.2 (a) depicts the graphical representation of the temporal and spatial constraints. The horizontal axis represents time dimension and the vertical axis represents space dimension. The shaded area represents time and *locale* where role DaySurgeon is enabled. Note that role DaySurgeon is enabled at all *locales* associated with SpC1 and between time interval t_1 and t_2 which is associated with temporal constraint DayTime.

Table 4.2 Access Control Policy Roles, Semantics and Expressions GST-RBAC

Role Name	***Semantics***	***Expression***
DaySurgeon	*The role is enabled and activated at DayTime , every day of the week and for locations inside Surgical Room and the Surgeon Prep (represented by SpC_1 constraint)*	*1. (DayTime ^ SpC1 , enable, DaySurgeon)* *2. (NightTime ^ SpC1, enable, NightSurgeon)* *3. (DayTime^ SpC1, $assign_u$, Beth, DaySurgeon);* *(NightT ime^ SpC1,$assign_u$, Mark, NightSurgeon);* *(null,SpC2, $assign_u$, Adam, SeniorNurse);* *(NightTime^ NS, $assign_u$, Ami, NightNurse);* *(NightTime ^SpC2, $assign_u$, Meg, NightNurse);* *($assign_u$, Kevin,PrepSurgery);* *($assign_u$, Bill, SurgeryLab)* *5. (Daytime ^ SpC1, $activate_r$, Beth, DaySurgeon)* *→(Nighttime^ SpC1, $deactivate_r$, Mark, NightSurgeon)* *6. (Nighttime^ OR, $activate_r$, Mark,NightSurgeon)* *→(Daytime^ SpC1, $deactivate_r$, Beth,DaySurgeon)* *7. spSSoD(SeniorNurse, NightNurse, NS1)* *8. (null,NS1, $activate_r$, Adam, SeniorNurse)* *9. (NightTime^NS1, $activate_r$, Ami, NightNurse);* *(NightTime^NS1, activater, Meg, NightNurse)* *10. SeniorNurse≥NS1NightNurse*
NightSurgeon	*The role is enabled and activated at NightTime , every day of the week and for locations inside Operation Room and the Surgeon Prep (represented by SpC_1 constraint)*	
SeniorNurse	*Role enabled and activated all day in locations represented by spatial constraint SpC_2*	
PrepSurgery	*Role enabled and activated all day at locations specified by SpC_3 and only if roles DaySurgeon and NightSurgeon are enabled*	



(a) **(b)** **(c)**

Figure 4.2 Examples of GST-RBAC expressions using (a) using or (∨) operator. (b) using the and (^) operator. (c) using the not (¬) operator with temporal and spatial constraints

Similarly according to GST-RBAC expression: (DayTime ^ SpC1,Enable, DaySurgeon), the Role DaySurgeon is enabled when both DayTime and associated SpC1 constraints are satisfied. The shaded area in Figure 3.4(b) depicts the time and location for enabling of DaySurgeon role. As per GST-RBAC expression (¬DayTime ^ ¬SpC1,Enable, DaySurgeon) Role DaySurgeon is enabled when both DayTime and SpC1 are not satisfied. The shaded area in Figure 3.4(c) depicts the time and location for DaySurgeon role to be enabled.

SoD constraints demonstrate how permissions can be made available to a certain role in a certain location while avoiding the separation of duty conflicts. A motivating example for a situation where SoD is useful can be considered when two roles, say, NightSurgeon and DaySurgeon should not be activated by the same Surgeon in the *locale* associated with SpC1during DayTime. Accordingly, this spatial SoD can be expressed formally as: spTSoD(DayTime,SpC1,DaySurgeon,NightSurgeon) .

Role hierarchy in GST-RBAC allows role to inherit permissions from users and devices. A spatially restricted role hierarchy can be defined between the role JuniorSurgeon and SeniorSurgeon by the expression, JuniorSurgeon$\geq_{SpC1}$ SSGSeniorSurgeon. This results in all the permissions associated with role SeniorSurgeon to be available to role JuniorSurgeon at the *locale* associated with SpC1. The rationale behind this hierarchy is that a user activating role JuniorSurgeon in an Surgical Room inherits permissions from the role SeniorSurgeon so that life saving procedures can be based on all available data rather than a restricted data set available to the JuniorSurgeon.

Expression 1 in Table 4.2 shows that the role NightSurgeon is enabled during temporal constraint NightTime and at *locale* described by SpC1. Expression 3 indicates that User Beth is assigned to role DaySurgeon within *locale* of SpC1 during DayTime. User Adam is assigned the role SeniorNurse while at *locale* associated with SpC2 any time of the day. User Ami is assigned the role NightNurse only when she is in Nursing Station (*NS*) and at time NightTime. However, Meg is assigned role NightNurse at all *locales* associated with SpC2 and at NightTime. It may be noted that although Meg and Ami are assigned the same role (i.e. NightNurse) and during the same time of the day, Ami can only access permissions of this role from the nursing station, while Meg can access the same permissions from *NS* as well as from the recovery rooms. Expressions 5 and 6 of Table 3.4, indicate activation constraints of activation/deactivation for users Adam and Mark. Both the users are activated when they are at the *locale* associated with SpC1 and only at the specified times of the day. Expression 7 defines a spatial SoD between roles SeniorNurse and NightNurse for location *NS1*. The rational for this constraint is that in the presence of a senior nurse the junior nurse should not be able to acquire permissions while at location *NS1*. However, the two roles can be activated by the same user in locations other than *NS1* but at the *locale* associated with SpC2. Expression 10 depicts the spatial hierarchical relationship between roles SeniorNurse and NightNurse at location *NS1*. The permissions of role NightNurse are inherited by the SeniorNurse only at location *NS1*.

5. Verification Framework for GST-RBAC

In this section, we develop a specification model for GST-RBAC policy and outline a methodology for its verification. Our modeling approach is three tiered. Firstly, we specify various conflicts that may arise in GST-RBAC policy. Secondly, we develop a GST-RBAC policy specification model by capturing the desired GST-RBAC features in a light-weight formal model, with the aim to identify potential conflicts in GST-RBAC specifications. We use the Alloy specification language and the accompanying constraint analyzer to verify the specification model [Jac03]. Subsequently, we utilize the specification model to develop a model of GST-RBAC policy for conflict analysis. The objective is to provide a formal framework to the policy administrator to compose GST-RBAC policies and verify policy composition prior to its implementation. This methodology also allows a conflict free evolution of both the access control model being used to implement the desired authorization framework (in this case GST-RBAC) and the actual policy instance being implemented in CPS environment.

5.1.1 Conflicts in GST-RBAC

During the life cycle of an access control policy, constraints can be added and removed from the policy. The evolutionary nature of access control policies can potentially lead to situations in which constraints may conflict with each other. Our aim is to detect such situations in which spatial constraints can lead to harmful and detrimental consequences.

Various types of conflicts may arise in a GST-RBAC system. GTRBAC conflicts [Jos05] are included in the conflicts that may arise in GST-RBAC only due to temporal constraints. We incorporate spatial conflicts and analyze temporal conflicts of GST-RBAC and categorize them into categories of conflicts explained below:

1. *Type 3a(i),* are the role enabling or assignment conflicts defined in [Jos05], in which the conflicting constraints are temporal in nature only.
2. *Type 3a(ii)* conflict constraints are the role enabling and assignments conflicts in which both the conflicting constraints are spatial in nature. Consider (SpC_1, enable, r_1) where $r_1 \in R$ and (SpC_2, disable, r_1) where $r_1 \in R$ where $SpC_1 = SpC_2$. This implies that the same role r1 will be enabled as well as disabled at the same location.
3. *Type 3a(iii)* conflict constraints are the role enabling and assignments conflicts in which both the conflicting constraints are spatial and temporal in nature. Consider ($TempC_1$, SpC_1, enable, r_1) where $r_1 \in R$ and ($TempC_2$, SpC_2, disable, r_1) where $r_1 \in R$ where $TempC_1 = TempC_2$ and $SpC_1 = SpC_2$. This implies that the same role r_1 is enabled as well as disabled at the same *locale* and time.
4. Conflicting SSoD is a conflicting spatial SoD constraint and is designated as Type 4. The Type 4 conflict occurs when a spSoD relationship is defined for the same *locale*. Formally, this conflict is of the form, $spSSoD(SpC_1, r_1, r_2) \wedge \neg spSSoD(SpC_2, r_1, r_2)$ where $SpC_1 = SpC_2$, and $r_1, r_2 \in R$ and $r_1 \neq r_2$.
5. Conflicting hierarchies are the role hierarchies with conflicts by definition of spatial and temporal constraints which lead to situations where role hierarchy may not work as desired. We designate these conflicts as type 5 conflicts. Consider the example of two roles r_1 and r_2 that may have inheritance property at SpC_1 as defined by $r_1 \geq_{SpC1} r_2$ .Additionally, the two roles r_1 and r_2 may also be enabled for the location space defined

by SpC_1. However the enabling times of the two roles may not overlap, consequently, not allowing this inheritance to be applied.

5.2 Specification Modeling and Conflict Identification

In order to capture the functional structure of the GST-RBAC policy, we use Alloy [Jac02,Jac03] that has a number of features which make it convenient for formal specification of access control policies. First, Alloy is based on predicate logic and its set theoretic based First-order modeling notation is used to model software components. Second, Alloy model is declarative, implying it lists the properties and constraints of the system being analyzed. Thirdly, the Alloy formalism captures the structure of the original software rather than events and allows specifying conditions and constraints which causes the software to go through state transition For an exhaustive detail of Alloy, refer to [Jac02].

Once the software is specified using Alloy formalism it can be analyzed using the accompanying constraint analyzer which allows simulation of the model to generate structure and behavior in the form of examples of the system. The constraint analyzer also allows checking of models using a counter example approach which identifies model properties which do not hold under the specified conditions.

5.2.1 GST-RBAC Policy Specification Modeling using Alloy

Access control policies are inherently declarative in nature and exhibit a structure comprising of a set of constraints and assertions. Policy assertion cause security systems to go from one state to the next which can be achieved only if the constraints allow. Since access control policies may evolve over time in CPS, the need for conflict identification cannot be overstated. Creating a policy specification model using Alloy and employing the Alloy constraint analyzer can assist in composition and evolution of consistent access control policies.

The complete Alloy model is available in [Jac02,Jac03]. The model is divided into four parts, namely: declarations, invariants, functions/predicates and assertions. In order to capture the functional structure of the GST-RBAC policy, we define objects as the basic pillars of the policy, together with constraints defining rules which govern the interplay between these objects.

Declarations of objects are in the form of *sig* structure and include elements such as fields (Table 5.1). The sig *User,* sig *Role,* sig *Permission,* sig *Location* and sig *Time* are the five basic signatures of objects of the model. It may be noted here that User, Role and Permission signify the usual components of the RBAC model.

The *RoleEnable* signature in Table 5.1 has a field *re_member* that maps Roles to Location to Time. In fact, *re_member* is a four-way mapping associating *RoleEnable, Role, Location* and *Time*. re_member can be thought of as a relation which represents the *roles* which have been enabled for a particular location and time. Signature *RoleDisable* has similar dynamics as the *RoleEnable*, only difference being that it refers to the set of roles which have been disabled for the respective times and locations, referred by the sole attribute *rd_member*. In order to constrain the structure of the access control policy, facts have been defined. Most facts in the Alloy model of GST-RBAC correspond to the conflicts defined in [Jos05] and Section 5.1.1

Table 5.1 Signature Declarations of GST-RBAC Policy

Alloy Signature Declaration
(a) sig User {}
(b) sig Role{}
(c) sig Permission{}
(d) sig Location {operator: SpatialOperators -> Location}
(e) sig SpatialOperators{}
(f) sig Time {}
(g) sig RoleEnable {re_member :some Role->some Location ->some Time}
(h) sig RoleDisable {rd_member : some Role->some Location ->some Time}
(i) sig UserRoleAssignment {URA_member : some User ->some Role -> some Location ->some Time}
(j) sig UserRoleDeAssignment {URDA_member :some User ->some Role ->some Location ->some Time}
(k) sig RolePermissionAssignment {RPA_member :some Role->some Permission ->some Location ->some Time}
(l) sig RolePermissionDeAssignment {RPDA_member :some Role->some Permission ->some Location ->some Time}
(m) sig UserRoleActivation {URAct_member :some User-> some Role->some Location->some Time}
(n) sig UserRoleDeActivation {URDAct_member :some User->some Role-> some Location->some Time }
(o) sig RoleHierarchy {rh_member: some Role -> some Role -> some Location -> some Time}

The fact *UsersEnableNotDisable* in Table 5.2 (a) ensures that a role enabled for a location *l* and time *t* will not be disabled. Conversely, if a role has been disabled at a location *l* and time *t*, then it will not be enabled. The two signature structures, *RoleEnable* and *RoleDisable* are utilized for maintaining the two disjoint sets. This fact is semantically equivalent to conflict Type 1a defined in [Jos05] and the spatial role enabling conflict defined in Section 5.1.1

EnableRole in Table 5.2 (b) enables a role in the model where *re* and *re'* are the role enabled sets before and after the role enabling action. The difference between the two sets is the addition of the current tuple of role, location, time. Additionally, *rd* and *rd'* are the two sets which represent the disabled roles before and after the role enabling action. Similarly, the *DisableRole* predicate [Sam07] works opposite to the enable role predicate. Executing the predicate by the command *EnableRole for 3 but 2 RoleEnable, 2 RoleDisable* causes Alloy to look for examples where the facts and the predicate are true. In case it cannot find such an example, alloy returns with a report of inconsistency. In the current case ,it comes up with an example where the fact *UsersEnableNotDisable* and predicate *EnableRole* are true. The visual representation of role

enabling is depicted in Figure 5.1. Note, that role Role0 has been enabled for location Location0 and at two times Time0 and Time2. On the other hand the same role, Role0 is disabled at the same location Location0 but at time Time1, which conforms to the defined fact.

In order to evaluate the complete working of the *EnableRole* and *DisableRole* pair, assertion, *CheckRoleEnable (Table 5.2 c)* is excecuted by the command *check CheckRoleEnable for 2*. The assertion *CheckRoleEnable* enables a role and then disables it.. This assertion looks for counter-examples whereby facts and assertion are at odds with each other. In our case, no counter example is found, implying that no violations of the conflict *3a(i)* (Section 5.1.1) were found for two instances of role enabling.

Table 5.2 Facts, Assertions and Predicates of Alloy Model

<table>
<tr>
<td>(a) fact UsersEnableNotDisable{//conflict type 1a
some rd: RoleDisable, re: RoleEnable,
r: Role, t: Time, l: Location |
((r -> l-> t) in re.re_member =>
(r -> l-> t) not in rd.rd_member &&
(r -> l-> t) not in re.re_member =>
(r -> l-> t) in rd.rd_member) }</td>
<td>(b)pred EnableRole (rd, rd': RoleDisable,
re, re': RoleEnable, r: Role, t: Time, l: Location) {
re'.re_member=re.re_member+ (r-> l -> t) &&
rd'.rd_member=rd.rd_member- (r-> l -> t) }</td>
</tr>
<tr>
<td rowspan="2">(c)assert CheckRoleEnable {
all rd, rd', rd'': RoleDisable, re, re', re'': RoleEnable,
r: Role, t: Time, l: Location |
no r.(re.re_member) and
EnableRole (rd, rd', re, re', r, t, l)
and DisableRole (rd', rd'', re', re'', r, t, l)
implies re.re_member=re''.re_member }</td>
<td>(d) sig RoleHierarchy
{rh_member :textit{Role} -> Role -> Location -> Time}</td>
</tr>
<tr>
<td>(f) fact OneWayHierarchy{
some rh: RoleHierarchy,
u: User, r1, r2: Role, t: Time, l: Location|
((r1->r2 -> l-> t) in rh.rh_member =>
(r2->r1 -> l-> t) not in rh.rh_member) }</td>
</tr>
<tr>
<td>(e)((u->r -> l-> t) in ura.URA_member =>
(u->r -> l-> t) not in urda.URDA_member &&
(u->r -> l-> t) not in ura.URA_member =>
(u->r -> l-> t) in urda.URDA_member)</td>
<td>(g)pred SoD (r1, r2:Role, u:User, uract: UserRoleActivation,
l:Location, t: Time){
(u->r1-> l-> t) in (uract.URAct_member) =>
(u->r2-> l-> t) not in (uract.URAct_member) }</td>
</tr>
</table>

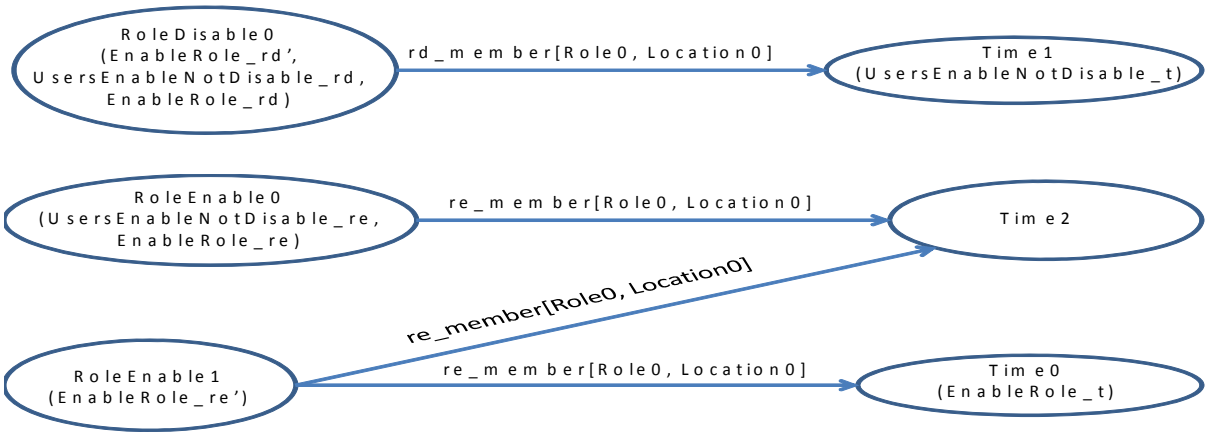


Figure 5.1 Role enable/disable for one role at one location and at three times

The signature, *UserRoleAssignment* [Sam07] refers to the set of users which have been assigned roles for certain locations and times. The *UserRoleDeAssignment* is the set of users, roles, locations and times which have been deassigned. Conflicts Types 3a(ii) and 3a(iii) (as defined in Section 5.1.1) are represented as facts *UsersAssignedNotDeassigned* and *RoleEnabledThenAssigned* in [Sam07]. The fact *UsersAssignedNotDeassigned* ensures that if a user is assigned to a role at a specific location and time, then the same user cannot also exist as a deassigned user of a role for the same location and time. This is made possible by the alloy formula mentioned in table 5.2 (e) where *ura* and *urda* are instances of the signatures *UserRoleAssignment* and *UserRoleDeAssignment*. Conflict Type 2 has been incorporated in the model as fact *UsersAssignedNotDeassigned* which ensures that a user can only be assigned to an enabled role. We assign users with the help of predicate *UserRoleAssignPred*. Relevant portion of the output generated by Alloy is shown in Figure 5.2.

UserRoleAssignment

policy/UserRoleAssignment_0,

URA_member : (policy/User) ->some ((policy/Role) ->some ((policy/Location) ->some (policy/Time)))

policy/UserRoleAssignment_0	policy/User_0	policy/Role_0	policy/Location_0	policy/Time_0
policy/UserRoleAssignment_0	policy/User_0	policy/Role_0	policy/Location_1	policy/Time_0
policy/UserRoleAssignment_0	policy/User_0	policy/Role_1	policy/Location_0	policy/Time_0
policy/UserRoleAssignment_0	policy/User_0	policy/Role_1	policy/Location_1	policy/Time_0

UserRoleDeAssignment

policy/UserRoleDeAssignment_0,

URDA_member : (policy/User) ->some ((policy/Role) ->some ((policy/Location) ->some (policy/Time)))

policy/UserRoleDeAssignment_0	policy/User_0	policy/Role_0	policy/Location_0	policy/Time_1
policy/UserRoleDeAssignment_0	policy/User_0	policy/Role_0	policy/Location_1	policy/Time_1
policy/UserRoleDeAssignment_0	policy/User_0	policy/Role_1	policy/Location_0	policy/Time_1
policy/UserRoleDeAssignment_0	policy/User_0	policy/Role_1	policy/Location_1	policy/Time_1

Figure 5.2 Partial Alloy output after running predicate UserRoleAssignPred

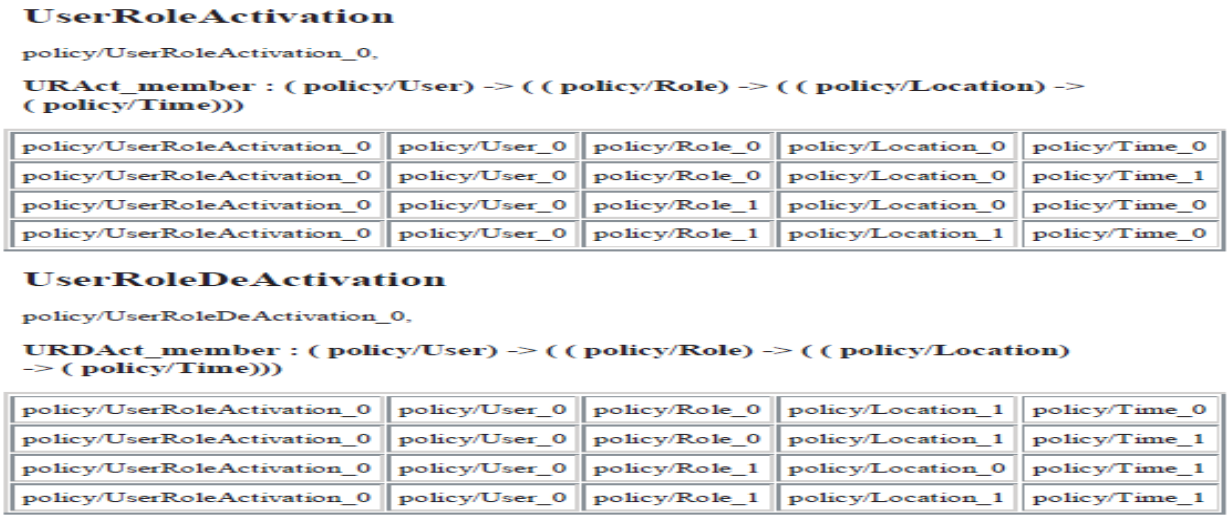
Note that user User_0 has been assigned to roles Role_0 and Role_1 at two locations (Location_0, Location_1) and times (Time_0, Time_1). The tables depicting *UserRoleAssignment* and *UserRoleDeAssignment* are both disjoint because of fact *UsersAssignedNotDeassigned*.

Similar analysis is done for modeling conflicts between events of role activation and role deactivation (Conflict 1d in [Jos05] and spatial role activation conflict, Type 3a(ii), defined in Section 5.1.1) . We use the signatures *UserRoleActivation* and *UserRoleDeActivation* which

correspond to the set of activated users and the set of deactivated users. The conflict has been represented by the fact *UsersActivatedNotDeActivated*. The predicate *UserRoleActivationPred* is used to test such conflicts. Part of the example generated by Alloy is depicted in Figure 5.3 which shows user User_0 has been assigned to roles Role_0 and Role_1 at two locations (Location_0, Location_1) and times (Time_0, Time_1). The two sets *UserRoleActivation* and *UserRoleDeActivation* have disjoint tuples.

Facts *RoleEnabledThenAssigned* and *RoleAssignedThenActivated* are conflicts related to activation listed in [Sam07]. *RoleAssignedThenActivated* ensures that the user is only assigned to a role which has been enabled. Similarly, *RoleAssignedThenActivated* ensures that a user can only activate a role from a certain location and time which has been assigned for the specified location and time.

The definition of *spSoD* (Section 3.2.6) in the alloy specification model of GST-RBAC model is achieved by the Separation of Duty predicate mentioned in Table 5.2 (g). This predicate ensures that *spSoD* can be defined between roles *r1* and *r2* at location *l* and time *t*. The signature object used in this case is *UserRoleActivation*. A user *u* can activate role *r1* at location *l* and at time *t* but cannot activate role *r2* at the same time and location. *spSoD* has been defined as a predicate so that it can be called during composition of the policy. Note that this predicate can also be used to define time-based separation of duty constraints as defined in [Jos05].



UserRoleActivation

policy/UserRoleActivation_0,

URAct_member : (policy/User) -> ((policy/Role) -> ((policy/Location) -> (policy/Time)))

policy/UserRoleActivation_0	policy/User_0	policy/Role_0	policy/Location_0	policy/Time_0
policy/UserRoleActivation_0	policy/User_0	policy/Role_0	policy/Location_0	policy/Time_1
policy/UserRoleActivation_0	policy/User_0	policy/Role_1	policy/Location_0	policy/Time_0
policy/UserRoleActivation_0	policy/User_0	policy/Role_1	policy/Location_1	policy/Time_0

UserRoleDeActivation

policy/UserRoleDeActivation_0,

URDAct_member : (policy/User) -> ((policy/Role) -> ((policy/Location) -> (policy/Time)))

policy/UserRoleActivation_0	policy/User_0	policy/Role_0	policy/Location_1	policy/Time_0
policy/UserRoleActivation_0	policy/User_0	policy/Role_0	policy/Location_1	policy/Time_1
policy/UserRoleActivation_0	policy/User_0	policy/Role_1	policy/Location_0	policy/Time_1
policy/UserRoleActivation_0	policy/User_0	policy/Role_1	policy/Location_1	policy/Time_1

Figure 5.3 Relevant Alloy output after running predicate UserRoleActivationPred

Spatial role hierarchy, defined in Section 3.2.7 is captured in the GST-RBAC Alloy specification model by the signature *RoleHierarchy* in Table 5.2 (d). The attribute *rh_member* is

a five-way mapping between role hierarchy, senior role, junior role, location and time. The addition of time parameter in this mapping ensures that this structure can be utilized to define temporal role hierarchies proposed in [Jos05]. Fact *OneWayHierarchy,Table 5.2 (f)* ensures that if there is a role hierarchy defined between two roles *r1* and *r2*, then it cannot be defined in the reverse sense (*r2* to *r1*). Note that this relationship is constrained in the temporal and spatial dimension, implying that the direction of a hierarchy is preserved for a given time and location.

In the above discussion, we have introduced a methodology for the creation of the GST-RBAC specification model using Alloy. This is achieved by representing conflicts as facts and ensuring that each fact can be validated for the signatures depicted in Figure 5.1. Complete listing of the GST-RBAC specification model is available at [Sam07].

5.2.2 GST-RBAC Access Control Policy Modeling using Alloy

In the previous subsection, we presented the GST-RBAC specification model using the Alloy framework. In this subsection we use the developed policy framework to model an access control policy and illustrate the conflict resolution mechanism afforded by Alloy to create conflict free policy artifact for access control in CPS which can be composed and analyzed piece-meal. In the following discussion we demonstrate the access control policy using example from Section 4. To demonstrate our approach, we describe in detail role-enabling, user-role-assignment, user-role-activation and spatial SoD functions of the GST-RBAC policy.

The complete listing of the policy specification for role enabling is available at [Sam07]. Role signature is extended by *DaySurgeon*, Time signature is extended by *DayTime* and *NightTime*, and Location signature is extended by SpC_1. The policy assertions of Table 4.2 are represented in the Alloy model as facts shown in Table 5.3

The facts *DaySurgeonEnableAtDayTime* and *NightSurgeonEnableAtNightTime* presented in Table 5.3 *e* and *a* enable the role *DaySurgeon* and *NightSurgeon* at *DayTime* and *NightTime*, respectively. Partial view of output of executing predicate *EnableRole* is depicted in Figure 5.4. Since the predicate evaluates to true and no fact is violated, Alloy outputs an example. Figure shows the role *DaySurgeon_0* is enabled at *Daytime* and at location *SpC1*. On the other hand the role, *DaySurgeon_0* is disabled at *NightTime* at location *SpC1*. The fact *NightSurgeonDisableAtNightTime in Table 5.3 b* violates the previously defined *NightSurgeonEnableAtNightTime*. Alloy does not return an example for the role enable /disable signatures and states that the model is inconsistent.

Table 5.3 GSTRBAC Access Control Policy in Alloy

Facts and Predicates	
(a)NightSurgeonEnableAtNightTime{ *some rd: RoleDisable, re: RoleEnable,* *r: NightSurgeon , t: NightTime, l: SpC1 |* *(r -> l-> t) in re.re_member }*	*(g)AdamNotAssignedToDaySurgeonAtNightTimeAtSpC1{* *some ura: UserRoleAssignment,* *urda: UserRoleDeAssignment, u: Adam,* *r: DaySurgeon, t: NightTime, l: SpC1 |* *(u->r -> l-> t) not in ura.URA_member and* *(u->r -> l-> t) in urda.URDA_member }*
(b)NightSurgeonDisableAtNightTime{ //conflict *some rd: RoleDisable, re: RoleEnable,* *r: NightSurgeon, t: NightTime, l: SpC1 |* *(r -> l-> t) in rd.rd_member}*	*(h)MarkAssignedToNightSurgeonAtNightTimeAtSpC1{* *some ura: UserRoleAssignment,* *urda: UserRoleDeAssignment, u: Mark,* *r: NightSurgeon, t: NightTime, l: SpC1 |* *(u->r -> l-> t) in ura.URA_member and* *(u->r -> l-> t) not in urda.URDA_member}*
(c)AdamActivatesDaySurgeonAtDayTimeAtSpC1{ *some uract: UserRoleActivation, urdact:* *UserRoleDeActivation, u: Adam, r: DaySurgeon, t: DayTime, l: SpC1 |* *(u->r -> l-> t) in uract.URAct_member and* *(u->r -> l-> t) not in urdact.URDAct_member}*	*(i)MarkNotAssignedToNightSurgeonAtDayTimeAtSpC1{* *some ura: UserRoleAssignment,* *urda: UserRoleDeAssignment, u: Mark,* *r: NightSurgeon, t: DayTime, l: SpC1 |* *(u->r -> l-> t) not in ura.URA_member and* *(u->r -> l-> t) in urda.URDA_member}*
(d)AdamSoDForDaySurgeonAndNightSurgeon { *some r1: DaySurgeon, r2:NightSurgeon,* *u: Adam, uract: UserRoleActivation, l:SpC1, t: DayTime|* *(u->r1-> l-> t) in (uract.URAct_member) =>* *(u->r2-> l-> t) not in (uract.URAct_member)}*	*(j)fact AdamAssignedToDaySurgeonAtDayTimeAtSpC1{* *some ura: UserRoleAssignment,* *urda: UserRoleDeAssignment, u: Adam,* *r: DaySurgeon, t: DayTime, l: SpC1 |* *(u->r -> l-> t) in ura.URA_member and* *(u->r -> l-> t) not in urda.URDA_member}*
(e)DaySurgeonEnableAtDayTime{ *some rd: RoleDisable, re: RoleEnable,* *r: DaySurgeon, t: DayTime, l: SpC1 |* *(r -> l-> t) in re.re_member }*	*(k)pred UserRoleActivationPred (ura, ura':* *UserRoleActivation, urda, urda': UserRoleDeActivation,* *u: User, r: Role, l: Location, t: Time){* *ura'.URAct_member=ura.URAct_member+ (u->r->l->t) &&* *urda'.URDAct_member=urda.URDAct_member- (u->r-> l -> t)}*
(f)DaySurgeonDisableAtNightTime{ *some rd: RoleDisable, re: RoleEnable,* *r: DaySurgeon, t: NightTime, l: SpC1 |* *(r -> l-> t) in rd.rd_member}*	*(l)MarkDeActivatesNightSurgeonAtDayTimeAtSpC1{* *some uract: UserRoleActivation, urdact:* *UserRoleDeActivation, u: Adam, r: NightSurgeon, t: DayTime, l: SpC1 |* *(u->r -> l-> t) not in uract.URAct_member and* *(u->r -> l-> t) in urdact.URDAct_member}*

Figure 5.4 Alloy output after running predicate EnableRole for the example in Section 4

In Table 4.2, two users, Adam and Mark are assigned to roles *DaySurgeon* and *NightSurgeon*, respectively. The constraints on the two assignments are both temporal and

spatial. We extend the User signature for Adam and Mark and define four new facts to represent the assignment function in Table 5.3. The fact *AdamAssignedToDaySurgeonAtDayTimeAtSpC1* ensure that Adam is assigned to the role *DaySurgeon* at time *DayTime* and location *SpC1*. similarly, the fact and constraint *MarkAssignedToNightSurgeonAtNightTimeAtSpC1* perform a similar assignment for Mark at a different time, but the same location. Facts 5.3 g, i ensure that Adam and mark are not assigned to the wrong roles at the wrong times. On executing Predicate, *UserRoleAssignPred* Alloy returns a policy depicted in Figure 5.5. Note, users Adam and Mark are made part of the *UserRoleAssignment* signature structure for role assignment at designated times and locations. On the other hand they have been made part of the *UserRoleDeAssignment* signature structure for the specified roles, time and locations. Next, we activate user *Adam* in roles *DaySurgeon* at time *DayTime* and at location *SpC1*. We also deactivate user *Adam* (if he is active) from role *NightSurgeon.* (Table 4.3).

UserRoleAssignment

policy/UserRoleAssignment_0, policy/UserRoleAssignment_1,

URA_member : (policy/User) -> ((policy/Role) -> ((policy/Location) -> (policy/Time)))

policy/UserRoleAssignment_0	policy/Adam_0	policy/DaySurgeon_0	policy/SpC1_0	policy/DayTime_0
policy/UserRoleAssignment_0	policy/Mark_1	policy/NightSurgeon_0	policy/SpC1_0	policy/NightTime_1
policy/UserRoleAssignment_1	policy/Adam_0	policy/DaySurgeon_0	policy/SpC1_0	policy/DayTime_0
policy/UserRoleAssignment_1	policy/Mark_1	policy/NightSurgeon_0	policy/SpC1_0	policy/NightTime_1

UserRoleDeAssignment

policy/UserRoleDeAssignment_0, policy/UserRoleDeAssignment_1,

URDA_member : (policy/User) -> ((policy/Role) -> ((policy/Location) -> (policy/Time)))

policy/UserRoleDeAssignment_1	policy/Adam_0	policy/DaySurgeon_0	policy/SpC1_0	policy/NightTime_0
policy/UserRoleDeAssignment_1	policy/Mark_1	policy/NightSurgeon_1	policy/SpC1_1	policy/DayTime_0

Figure 5.5 Alloy output after the predicate UserRoleAssignPred for Example in Section 4

We test the model by running *UserRoleActivationPred* predicate (Table 5.3). Alloy presents an example where the above facts are true. In order to illustrate the composition of spatial SoD we define *spSoD* for roles *DaySurgeon* and *NightSurgeon* which are conflicting for user *Adam* in Table 5.3. Note that this situation is not defined in the example in Section 4. The *spSoD* fact allows *Adam* to activate role *DaySurgeon* but de-activates *Adam* in role *NightSuregeon*.

In this section we have demonstrated the proposed methodology for composition of GST-RBAC policy using the GST-RBAC specification model developed in Section 4. Note that this composition methodology can be employed to analyze policy components before implementing in CPS. Also, Alloy policy model can assist the administrator to add new constraints, permissions, roles to the policy in a consistent and conflict free manner.

6 Conclusion

In this paper, we have presented a spatial temporal RBAC model for CPS. We have formally developed the notion of spatial constraints in which participating locations have semantic relationship with each other and access control decisions are determined based on these relationships. We have also defined spatial separation of duty and role hierarchy with spatial constraints. We have analyzed the proposed GST-RBAC model by defining conflicts which may arise while using spatial constraints in addition to temporal ones. We have also illustrated the complete formalism with example from the health care domain where location of a user/device has direct bearing on the access control privileges available to him.

In order to analyze the proposed GST-RBAC model, we have developed its formal specifications using Alloy. The specification model is analyzed utilizing the accompanying Alloy constraint analyzer for identifying conflicts and subsequent resolution. We have illustrated the composition of an organization's access control policy using the GST-RBAC policy specification model. Conflicts which may arise while composing an access control policy become evident using the Alloy constraint analyzer. We have simulated an access control policy in a light-weight formal environment that helps to uncover security flaws. We have demonstrated that conflict resolution for access control policies may be done piece-meal allowing the policy to be analyzed step by step during its engineering phase.

Although, Alloy offers a practical formalism for access control specifications, its performance can be a shortcoming, especially when the policy model includes hundreds of roles. Although, modeling portions of the policy at design time mitigates this drawback, a full analysis can be complex. Further, the Alloy formalism is expressive enough for developing an access control model, but it is more desirable to it with some form of visual tools. This is more relevant to policy administrator who may not have the desire to fully understand Alloy syntax. In the future we plan to develop such a visual interface for access control policy specification so that Alloy coupled with this interface can be adopted for complex CPS.

References

[Abo97] G. D. Abowd, C. G. Atkeson, J. I. Hong, S. Long, R. Kooper, and M. Pinkerton, "Cyberguide: A Mobile Context-aware Tour Guide," Wireless Networks, vol. 3, no. 5, pp. 421–433, 1997.

[Ahm03] T. Ahmed and A. R. Tripathi, "Static Verification of Security Requirements in Role Based CSCW Systems," in SACMAT '03: Proceedings of the eighth ACM symposium on Access control models and technologies, (New York, NY, USA), pp. 196–203, ACM, 2003.

[Alp84] B. Alpern and F. B. Schneider, "Defining Liveness," Cornell Tech. Rep., Ithaca, NY, USA, 1984.

[Bar10] S. Barnum, S. Sastry, J.A. Stankovic, "Roundtable: Reliability of Embedded and Cyber-Physical Systems," IEEE Security & Privacy, Vol. 8, No. 5, 2010, pp: 27 – 32

[Ber05] E. Bertino, B. Catania, M. L. Damiani, and P. Perlasca, "GEO-RBAC: a Spatially Aware RBAC.," in ACM SACMAT, pp. 29–37, 2005.

[Bha06] R. Bhatti, E. Bertino, and A. Ghafoor, "X-FEDERATE: A policy engineering framework for federated access management.," IEEE Trans. Software Eng., vol. 32, no. 5, pp. 330–346, 2006.

[Hull06]B. Hull, V. Bychkovsky, Y. Zhang, K. Chen, M. Goraczko, A. Miu, E. Shih, H. Balakrishnan, and S.Madden. 2006. CarTel: a distributed mobile sensor computing system. In Proceedings of the 4th international conference on Embedded networked sensor systems (SenSys '06). New York, NY, USA, pp: 125-138.

[Bur90] J. Burch, E. Clarke, K. McMillan, D. Dill, and L. Hwang, "Symbolic Model Checking: 1020 States and Beyond," Logic in Computer Science, 1990. LICS '90, Proceedings., Fifth Annual IEEE Symposium on e, pp. 428–439, 4-7 Jun 1990.

[Ege91] M. J. Egenhofer and R. D. Franzosa, "Point Set Topological Relations," International Journal of Geographical Information Systems, vol. 5, pp. 161–174, 1991.

[Haa94] V. Haarslev, R. Moller, and C. Schroder, "Combining Spatial and Terminological Reasoning," Advnaces in AI, Proc. 18th Annual Conference on AI, Saarbrucken, Sep. 1994, Vol. 861, pp: 142-153.

[Hig01] J. Hightower, G. Borriello, "Location Systems for Ubiquitous Computing," IEEE Computer, Vol. 34, No. 8., pp. 57-66, August 2001

[Jac02] D. Jackson, "Alloy: A Lightweight Object Modeling Notation," ACM Transactions on Software Engineering Methodology., vol. 11, no. 2, pp. 256–290, 2002.

[Jac03] D. Jackson, "Alloy: A Logical Modeling Language," in ZB, p. 1, 2003.

[Jos05] J. Joshi, E. Bertino, U. Latif, and A. Ghafoor, "A Generalized Temporal Role-Based Access Control Model.," IEEE Transactions on. Knowledge and Data Engineering., vol. 17, no. 1, pp. 4–23, 2005.

[Lee08] E. Lee, “Cyber Physical Design Challenges, “ Proceedings of 11th IEEE International Symposium on Object Oriented Real-Time Distributed Computing, May 2008, Orlando, FL, pp: 363 – 369

[Lee10] I. Lee, and O. Sokolsky, “Medical Cyber Physical Systems,” Proceedings of the 47th ACM Design Automation Conference, DAC '10, June 2010, Sweden, pp: 743-748

[Leo98] U. Leonhardt and J. Magee, “Multi-sensor Location Tracking.,” in 98’ MOBICOM, pp. 203–214.

[Mcd09] P. McDaniel, S. McLaughlin, "Security and Privacy Challenges in the Smart Grid," IEEE Security and Privacy, Vol. 7, no. 3, May/June 2009, pp. 75-77,

[Mit11] R. Mitchell, C. Ing-Ray, “A Hierarchical Performance Model for Intrusion Detection in Cyber Physical Systems,” Proceedings of the IEEE Wireless Communications and Networking Conference, (WCNC), March 2011, Cancun, Mexico, pp: 2095 - 2100

[Nit09] “High Confidence Medical Devices: Cyber Physical Systems for 21st Century Health Care,” NITRD Report, February 2009.

[Pea02] L. Pearlman, V. Welch, I. Foster, C. Kesselman, and S. Tuecke, “A Community Authorization Service for Group Collaboration,” Policies for Distributed Systems and Networks, 2002. Proceedings. Third International Workshop on, pp. 50–59, 2002.

[Poo10] R. Poovendran, “ Cyber–Physical Systems: Close Encounters Between Two Parallel Worlds”, Proceedings of the IEEE, Vol. 98, No. 8, August 2010, pp: 1363-1366.

[Rou10] C. Roussey , F. Pinet, “DL Based Automated Consistency Checking of Spatial Relationships,” Dans Conférence internationale de Géomatique et Analyse Spatiale SAGEO’10, Toulouse. November 2010, pp. 306-320.

[Sam07] A. Samuel, A. Ghafoor, E. Bertino, “A Framework for Specification and Verification of Generalized Saptio-Temporal Access Control Model,” CERIAS Tech Report 2007-08, Purdue University

[San00] R. S. Sandhu, D. F. Ferraiolo, and D. R. Kuhn, “The NIST Model for Role-based Access Control: Towards a Unified Standard.,” ACM Workshop on Role-Based Access Control, pp. 47–63, 2000.

[Sch93] B. N. Schilit, N. Adams, R. Gold, M. M. Tso, and R. Want, “The PARCTAB Mobile Computing System.,” in Workshop on Workstation Operating Systems, pp. 34–39, 1993.

[Ten08] Chee-Wooi Ten, Chen-Ching Liu, G. Manimaran, “Vulnerability Assessment of Cyber Security for SCADA Systems,” IEEE Transactions on Power Systems, Vol. 23, No. 4, 2008, pp: 1836 – 1846

[Wan93] R. Want, A. Hopper, V. Falcao, and J. Gibbons, “The Active Badge Location System.,” ACM Trans. Inf. Syst., vol. 10, no. 1, pp. 91–102, 1992.

[Wol09] W. Wolf, “Cyber-Physical Systems,” IEEE Computer, Vol. 42, No. 3, March 2009, pp: 88 - 89

[Zim10] C. Zimmer, B. Bhat, F. Mueller, S. Mohan, “Time-based Intrusion Detection in Cyber Physical Systems,” Proceedings of the 1st ACM/IEEE International Conference on Cyber-Physical Systems, ICCPS '10, April 2010, Stockholm, Sweden, pp: 109-118