STRBAC - AN APPROACH TOWARDS SPATIO-TEMPORAL ROLE-BASED ACCESS CONTROL

Mahendra Kumar
CISE Department
University of Florida
Gainesville, FL 32608
email: makumar@cise.ufl.edu

Richard E. Newman
CISE Department
University of Florida
Gainesville, FL 32608
email: nemo@cise.ufl.edu

ABSTRACT
The rapid emergence of GPS enabled devices, sensors and mobile equipment in commercial as well as government organizations has led to considerable research in time- and location-based access control schemes. Location-based access policies enhance the security of an application by restricting access to an object only from specified locations. On the other hand, temporal constraints provide granularity in security features and also limit damage to an application to a specific time interval (e.g. when staff are present to respond if necessary). This paper introduces a novel approach to location- and time-based access control mechanism using Role-Based Access Control (RBAC). We believe that it is well-suited for organizations that require time- and location-based access control over static or mobile objects.

KEY WORDS
Location-aware Computing, Time-dependent Access, Security Policies, RBAC, Access Control Model, Visibility.

1 Introduction

Time and location are important attributes that may determine the access control policies in an organization. Consider the following situation. The chief manager of a bank should be able to access the most important electronic files if and only if he is physically present inside his office and that operation is performed during normal banking hours. It is clear that in the above scenario we require the access policy to be defined in such a manner that it depends not only on the role of the individual but also on the location and the time during which the resource can be accessed.

To address similar issues, we propose a Spatio-Temporal Role-Based Access Control (STRBAC) model, an extension of the original RBAC model that supports temporal and location constraints. Although some approaches have been proposed in this area, our model provides an entirely different context to this work and deals with issues that have not been addressed previously, including notions of visibility and accessibility. Our model supports dynamic activation of roles and permissions depending upon the specified constraints. It also gives a new direction to the categorization of logical location and time.

The remainder of the paper is organized as follows. In Section 2, we provide some background information. Section 2.1 summarizes the RBAC model on which our work is based while Section 2.2 discusses issues surrounding determination of location of an entity. In Section 3, we discuss some related work in this area. Sections 4 and 5 describe the spatial and temporal models used by the STRBAC model respectively . Section 6 formalizes the concept of spatio-temporal expressions while Section 7 describes the complete framework of STRBAC model including the concept of visibility and accessibility. Finally, Section 8 concludes the paper.

2 Background

2.1 Role-Based Access Control

Role based access control (RBAC)[7] has emerged as a promising alternative to traditional discretionary and mandatory access control. The main components of RBAC consist of a set of users ($USERS$), a set of roles ($ROLES$), a set of objects ($OBJS$), a set of operations (OPS) and a set of permissions ($PERMS$). Roles are assigned to a user while a set of permissions related to each object are assigned to a role. Hence, when a role is assigned to a particular user, the user automatically acquires all the permissions associated with that specific role. Since the roles directly represent job functions in an organization, we do not have to define separate permissions for each and every user in an organization, thus making it easier to implement access policies where the number of users is very large. The assignment of roles greatly simplifies security policy management within an organization.

2.2 Location Determination

To determine the position of a subject/object, we need devices that can provide this information to the position determination system (PDS) reliably. GPS is one of the most popular and inexpensive methods to determine location information. Infrared sensors are another type of device that can determine the location of a user with high accuracy. In such a location-determining system, the infra-red base sensors occupy fixed positions. Every object has built-in infra-

red transponders and the users are required to wear active badges that can respond to the sensors [1]. This type of system is ideally suited for small organizations. Another type of position-determining system is a base station that can estimate that a user is within a certain geographical area. For example, in a wireless network, a particular base station can determine the location of the mobile users if those users are within its sensing range. Signal strength information from multiple stations may be used to estimate location more accurately. Other type of sensors include RF-IDs and biometric identification systems [5].

3 Related Work

Due to the popularity increasing use of geo-spatial devices, much work has been done on spatial RBAC. A spatial RBAC model proposed by Bertino et al. [11] is based on the OGC (Open GIS Consortium) spatial model. They have also extended their model to incorporate role hierarchies and separation of duty constraints. Various methods of location representation and determination have been discussed in [6, 8, 4]. Hengartner and Steenkiste [3] proposed a system for locating people at different places. They show how location information can be verified using secure methods called service trust. They have also pointed out the need for location granularity and time constraints. Many real world examples showing the need for location-based authentication have been mentioned by Denning and MacDoran [2]. They have suggested the use of GPS devices for obtaining location information. They have also discussed CyberLocator, a location-based authentication tool. Harter and Hooper [1] modeled a location system for an active office where they applied the use of infra-red devices for obtaining the location of people. Each room has an infra-red sensor and the people wear infra-red transponders called Active Badges. Lee et al. [9] discussed how to manage data in a location dependent system. They consider location to have a hierarchical structure and propose two different models, viz. the Geometrical model and the Symbolic model.

Joshi et al. [13] discusses a Generalized Temporal RBAC (GTRBAC) model, which has a set of language constructs for specifying various temporal constraints. Bertino [10] describes a way to express periodicity constraints in access control. We adopt some ideas from this model for representation of temporal expressions in STRBAC. Joshi et al. [12] discuss temporal hierarchies in RBAC .

4 Spatial Model in STRBAC

In order to make RBAC spatially capable, we should be able to express location in a convenient way that can be interpreted by humans easily. We should also have a standard way of representing location in raw format, as stored by the system. Hence, we define two levels of locations, namely $PrimitiveLocation$ and $LogicalLocation$. The system maintains a means of mapping a raw position to its corresponding primitive location. The position returned to the system by the user device is always in raw format. Whenever a location check is performed on the user, raw position is converted to the corresponding primitive location to evaluate the location constraint. This is because a location constraint is always defined in terms of logical location, which is built based on primitive locations.

4.1 Primitive Location Model

A primitive location is a basic 3-D shape which can be combined with other primitive locations to form a logical location. For example, the range of a wireless base station can be defined as a primitive location. Primitive location is formally defined as follows.

Definition 1 [Primitive Location] *A primitive location L_p is either the volume associated with basic unit of position that is returned by the PDS, or an artificially created volume defined by the administrator for PDSs that have high resolution. These may be created using Constructive Solid Geometry from basic geometric shapes defined by their coordinates.*

The position of a user is returned as a single point to the system. Hence, system can always determine whether user lies inside a primitive location by simple mathematics.

4.2 Logical Location Model

Logical locations help in two ways: aliasing and aggregation. Aliasing allows logical locations to have identifiers that are more suitable for human understanding. Aggregation helps in combining a set of primitive or logical locations into a single logical unit and hence helps in addressing a group of locations by a single name instead of addressing each of them individually. For example, it is more appropriate for the humans to refer to a set of rooms as lecture rooms (aggregation) or a particular room as the department office (aliasing). STRBAC divides logical locations into four different types: (1) *Location-by-Address*, (2) *Location-by-Use*, (3) *Location-by-Organization* and (4) *User-Defined Location.*

Location-by-Address: Location-by-Address specifies a location according to its physical position. For example, the postal address of a building can represent a set of several primitive locations.

Location-by-Use: In order to associate location with its function, we introduce the concept of Location-by-Use. For example, we may designate a set of rooms as classrooms, and another set as auditoriums.

Location-by-Organization: This is helpful in defining locations when they can be distinguished based on organizational ownership. As members of an organizational

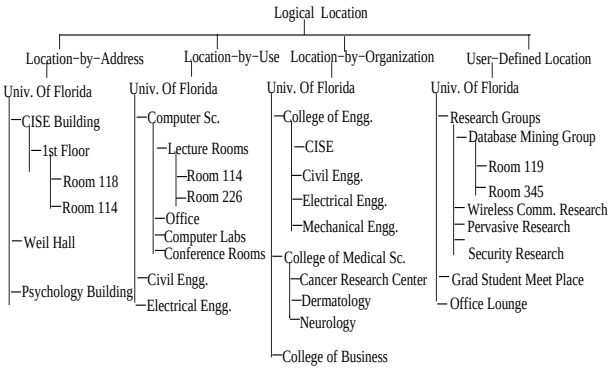

Figure 1. Hierarchy of different types of locations with examples

unit may be expected to have different permissions for resources owned by that unit than other users, Location-by-Organization is useful in STRBAC.

User-Defined Location: This type of logical location is used to create custom locations which do not fall in any extant logical category. This is helpful if the administrator wants to create some shortcut to a set of locations. For example, the Security Research group in our department does not fall under any logical location type and is comprised of a set of rooms and a lab. Hence, if we want to define this type of logical location, we can place it under user-defined location types. This allows finer granularity on cross-cutting locational units.

We formalize the definition of logical location as follows.

Definition 2 [Logical Location] *A logical location L_l is a combination of one or more logical or raw locations joined by a $\cup, \cap$ and/or $\setminus$ operator.*

Figure 1 shows the relationship among different type of locations.

5 Temporal Model in STRBAC

Desired policies require our temporal model to represent any recurring or non-recurring time interval for use in temporal expressions. Recurring intervals are used for defining access policies that occur periodically. Primitive non-recurring intervals specify a range of dates. Non-recurring intervals of different lengths are formed by combining them with other recurring and non-recurring intervals. For convenience, we define two types of notation to represent time intervals. If an interval is represented as $(x_1 - x_2)$, then the time interval is said to be continuous starting at the beginning of x_1 and ending at the end of x_2. If a set is represented as (x_1, x_2), then the set of time intervals is said to be discrete and expression will evaluate to true only during interval x_1 or interval x_2 and not in between.

5.1 Primitive Non Recurring Interval

A primitive non-recurring interval is a range of dates in STRBAC. It is represented by Δ where,

$$\Delta = (\delta_s - \delta_e) \mid \delta_d \quad (1)$$

where δ_s and δ_e denote the start date and end date of a range, and δ_d is shorthand for $\delta_d - \delta_d$ (i.e., a single day). A value of $\star$ for Δ indicates every day of every year. Dates are represented in the STRBAC system in the format $yyyy/mm/dd$.

5.2 Primitive Recurring Interval

Primitive Recurring intervals are used to define policies where access is granted on a periodic basis. Recurring intervals are constructed from primitive recurring intervals through set operations. Suppose that access to a bank vault is only allowed on business days between 9:00 am and 5:00 pm. In this case, we need an access policy that can specify this recurring interval. We represent these interval in a manner similar to Bertino et al. [10]. Primitive recurring intervals are sub-divided into four categories according to their period: $Daily, Weekly, Monthly$ and $Yearly$. Each category may be further divided according to the atomic interval of time used: $second$, day, $week$, or $month$. Time intervals are denoted by $\{x_1, \cdots, x_n\}.unit.period$, where x_i is either a unit interval u_i or a contiguous range of unit intervals, $s_i - e_i$, where s_i is the starting unit interval and e_i is the ending unit interval. For the STRBAC model, we assume that every entity in STRBAC is synchronized to a global clock which is independent of any geographical location.

Daily Period: A daily interval denotes a time interval in a day during which the access is granted. A Daily Interval is denoted by Γ_D where the definition follows similar to the one defined for non-recurring interval, except that instead of dates, its format is *hours:minutes:seconds*, with the second as the unit interval.

Weekly Period: A weekly interval denotes the days in a week during which the access is granted. We denote this constraint as Γ_W, and represent it as $\{x_1, x_2, \cdots, x_n\}$.day.week where $1 \leq x_i \leq 7$. x_i signifies the day of week during which access is granted. Sunday is assumed to be the first day of every week. Hence, $-6\}$.day.week represents that access is granted from Monday through Friday of every week.

Monthly Period: Monthly intervals use either days or weeks as unit intervals. Day interval in a month is denoted by Γ_{Md} and is represented as $\{x_1, x_2, \cdots, x_n\}$.day.month where $1 \leq x_i \leq ldm$. ldm represents the number of days in a month and can be set by simple calculations. x_i denotes the day of the month on which the access is valid. Hence, $\{2, 14 - 16\}$.day.month represents that access is valid only on the 2^{nd}, 14^{th}, 15^{th}, and 16^{th} days of each month. Similarly, week interval in a month is denoted by Γ_{Mw} and is represented as

$\{x_1, \cdots, x_n\}$.week.month where $1 \leq x_i \leq 5$ or $x_i = lwm$. Here, x_i denotes the week of the month during which the access is valid. Hence, $\{2, 4\}$.week.month represents that access is granted only during the 2^{nd} and 4^{th} week of each month. When $x_i = lwm$, then we assume that access is granted for the last week of the month i.e. the last seven days of the month.

Yearly Period: A yearly interval denotes the daily, weekly or monthly intervals in each year during which access is valid. Daily interval in a year denoted by Γ_{Yd}, is represented as $\{x_1, \cdots, x_n\}$.day.year where $1 \leq x_i \leq ldy$. ldy represents the number of days in a year. x_i denotes the day of the year on which the access is valid. Weekly interval in a year is denoted by Γ_{Yw} and is represented as $\{x_1, \cdots, x_n\}$.week.year where $1 \leq x_i \leq 53$. Similarly, Monthly interval in a year is denoted by Γ_{Ym} and is represented as $\{x_1, \cdots, x_n\}$.month.year where $1 \leq x_i \leq 12$. x_i denotes the month in a year during which access is valid.

6 Spatio-Temporal Constraint Formalization

For spatio-temporal based access policy support, the system should be able to perform operations on the location and temporal constraints. In this section we formalize the concept of spatio-temporal constraint, and its representation and evaluation.

Definition 3 [Spatial Constraint] *A spatial constraint P_s is a combination of one or more logical location spaces joined by an $\cup$, $\cap$ or $\setminus$ operator. Thus,*

$$P_s = L_{l1}\ op\ L_{l2}\ op\ \cdots L_{ln} \tag{2}$$

operator op $\in \{\cup, \cap, \setminus\}$
$P_s = \star$ denotes any location.

The minus ($\setminus$) is a special operator used to indicate a negative constraint.

One location constraint can be contained inside another location constraint.

Definition 4 [Containment of Spatial Constraint] *A spatial constraint P_{s1} is said to be contained inside another spatial constraint P_{s2}, denoted by $P_{s2} \succeq P_{s1}$, if the following condition holds:*

$$\forall (x_i, y_i, z_i) \in P_{s1}, (x_i, y_i, z_i) \in P_{s2} \tag{3}$$

where (x_i, y_i, z_i) is a valid point in P_{s1}.

Temporal constraints are a combinations of one or more recurring and non-recurring time intervals joined by a set of operators. A temporal constraint is formally defined as follows.

Definition 5 [Temporal Constraint] *A temporal constraint P_t is an expression combining one or more primitive time intervals joined by an $\cup$, $\cap$ or $\setminus$ operator. Thus,*

$$P_t = C_1\ op\ C_2\ op\ C_3 \cdots C_n \tag{4}$$

$C_i \in \{\Delta, \Gamma_D, \Gamma_W, \Gamma_{Md}, \Gamma_{Mw}, \Gamma_{Yd}, \Gamma_{Yw}, \Gamma_{Ym}\}$
operator op $\in \{\cup, \cap, \setminus\}$.
$P_t = \star$ denotes any time.

Definition 6 [Containment of Temporal Constraint] *A temporal constraint P_{t1} is said to be contained inside another temporal constraint P_{t2}, denoted by $P_{t2} \succeq P_{t1}$, if every unit of time in P_{t1} is also a valid unit of time in P_{t2}.*

Evaluation of spatial and temporal constraints by the system will always return a Boolean value depending on whether or not the user satisfies these constraints. Since both the spatial and temporal constraints are Boolean expressions, we can combine these two constraints into a single spatio-temporal constraint.

Definition 7 [Spatio-Temporal Constraint] *A spatio-temporal constraint P_{st} is a conjunction of a spatial and a temporal constraint, or a Boolean expression combining one or more spatio-temporal constraints joined by an AND ($\wedge$), OR ($\vee$) or NOT ($\neg$) operators. Thus,*

$$P_{st} = (P_s \wedge P_t)\ \mid\ (P_{st})\ \mid\ P_{st} \vee P_{st'}\ \mid\ \neg P_{st'} \tag{5}$$

Table 1 shows some examples of spatio-temporal constraints and their intended meaning.

7 Spatio-Temporal RBAC

In this section, we integrate the different components of RBAC with location and time to develop our composite STRBAC model. We also explain the association of different components of RBAC with location and time.

7.1 User's Mapping to Location

Users have a direct association with location since a user has to be in a particular location at any time. For evaluation of the spatio-temporal constraint, we need the location of the user as one of the parameters. When a user tries to perform an operation on an object inside a STRBAC system, his current location is checked against the locations where the role associated with him is allowed to perform the requested operation. User-to-location relation is many-to-one since one user can be present at only one location while one location can have many users associated with it (depending on spatial resolution).

7.2 Object Mapping to Location

The location of the object is required for setting visibility criteria for that object as discussed in section 7.3. One location can contain many objects. However one object is associated with at most one location at any time. Hence, the object-to-location mapping is many-to-one.

Spatio-Temporal Constraint, P_{st}	**Meaning**
$\star \wedge \star$	Every day at all times at every location
$(L_{s1} \cup L_{s2} \cup L_{s3}) \wedge$ ((2006/02/04 − 2006/02/15) ∩ (09:00:00−17:00:00))	From 4th of Feb. 2006 to 15th of Feb 2006 between 9am and 5pm and only from location L_{s1}, L_{s2} or L_{s3} .
$\star \wedge$ ((09:00:00−17:00:00) \ (12:30:00 −13:30:00))	Between 9 AM to 5 PM except $12:30$ PM to $1:30$ PM everyday
$\star \wedge \{2, 4, 6\}$.day.week	On Mondays, Wednesdays and Fridays of every week at every location
$(\star \wedge \{1, 15,$ldm$\}$.day.month) $\vee (L_{s1} \wedge \star)$	On the first, fifteenth and the last day of every month or from location L_{s1}.
$\star \wedge (\{11\}.month.year \wedge \{3\}.week.month \wedge \{5\}.day.week)$	3rd Thursday in November

Table 1. Different types of spatio-temporal expressions and their intended meaning

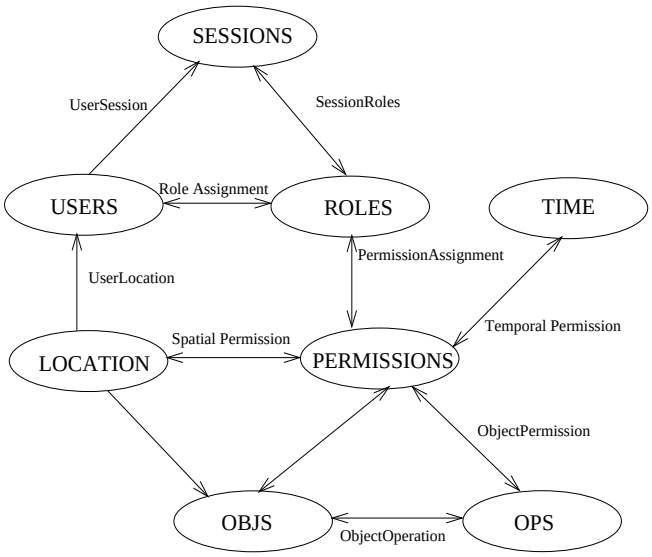

Figure 2. Association of Location and Time with different components of RBAC

7.3 Permission

A permission may be categorized into two basic types: Accessibility and Visibility.

7.3.1 Accessibility:

If role r is able to avail the services of the object i.e. actually performing some operations on the object, then the object is said to be accessible to role r.

One of the major issues is how to assign/deassign the accessibility to the roles when the time period starts or ends. Joshi et al. [12] suggested the use of triggers to assign/deassign permissions to role when each time period starts or ends. A drawback in their model is that they assign and de-assign all the permissions at the start and end times respectively even when the operations associated with that access are never used by any role member.

We use a different approach to solve this issue. We don't use triggers to activate an accessibility at the start period since the request to access an object is initiated by the user. Rather than follow the approach that the RBAC Access Matrix M, indexed by roles and objects, contains a time-varying set of access privileges, we take the approach that $M[r, o]$ contains pairs consisting of an access type α and the spatio-temporal predicate, P_{st} associated with that α for role r on object o.

When a user u attempts to access an object o using access type α, the entry (α, P_{st}) = $M[r, o]$ is obtained, then evaluated using the current global time t and the user's current position $pos(u)$. A spatial constraint component P_s evaluates to true iff $pos(u) \in P_s$, and a temporal constraint component P_t evaluates to true iff $t \in P_t$.

7.3.2 Visibility:

Location may be a sensitive piece of information. We may not want it to be revealed to everybody because it might lead to security risks. For example, military units may want to keep the location of the weapons a top secret. Similarly, an object's identity (or other attributes) may also represent sensitive information. An FBI agent may not want to reveal his identity to everybody for security reasons. To implement these constraints, we propose a new type of permission called visibility. With the help of visibility policies we can specify whether a set of roles is allowed to know the location and identity of an object or not.

• *LOCATION VISIBILITY*: Location visibility defines the resolution of the object's location information returned to the role member. Thus, location visibility of object o to role r, denoted by $Vis_L(r, o)$ is a function of of r and o that returns the spatial resolution to apply to the request by a user in role r for the position of object o. Resolution may take any of a number of forms, including distance, number of primitive locations, noise added to the position,

etc.

• *ATTRIBUTE VISIBILITY*:Attribute visibility defines the nature of the value of an object's (static) attribute returned to a role member. Attribute visibility provides granularity to object information which can be used to prevent threats and information leakage. Attribute visibility, denoted by Vis_A depends on role r, object o (i.e., is an element in M), and is a function of attribute a. It specifies what value should be returned to a user in role r when querying attribute a of object o.

7.3.3 Permission Formalization:

Permissions within STRBAC are thus specified as a (relatively static) access matrix $M[r, o]$ that consists of cells specifying the access that a user u in role r has for object o. There are three flavors of entries in $M[r, o]$: those for ordinary access types, an entry for the (possibly dynamic) location attribute of o, and an entry for the (relatively static) other attributes of o. If an ordinary access type α is allowed under any circumstances for r on o, then $M[r, o]$ contains a pair (α, P_{st}) with the spatio-temporal constraint P_{st} that must evaluate to true on the current time t and the current location of the user, $pos(u)$ for access α to be allowed. If u requests the current location of o, then a range of locations is returned that depends on the current location of o and Vis_l, where $(Loc, Vis_L) \in M[r, o]$. If u requests the value of attribute a of object o, then the value $Vis_A(a)$ is returned, where $(Attr, Vis_A) \in M[r, o]$.

8 Conclusion

This paper models spatio-temporal policies for access control extending the existing RBAC model in a novel way. We have formalized the concept of spatio-temporal constraints in the form of Boolean expressions and provide some examples to show their intended meaning. It introduces concepts of logical location and provides some real world examples to show their benefits. An important characteristic of STRBAC is that it deals with visibility issues, which have not been taken into consideration in other models. This is important in controlling a user's privacy information.

In the future, we will consider implementation and performance issues as well as usability issues related to the semantics of spatio-temporal constraints. We plan to extend this model to manage privacy. The current model does not have spatio-temporal restrictions on user-role bindings, which we also plan to consider.

References

[1] Andy Harter and Andy Hopper. A Distributed Location System for the Active Office. In *University of Cambridge,UK*, November 1993.

[2] Dorothy E. Denning and Peter F. MacDoran. Location-Based Authentication:Grounding Cyberspace for Better Security. In *Computer Fraud and Security,Elsevier Science Ltd*, February 1996.

[3] Urs Hengartner and Peter Steenkiste. Implementing Access Control to People Location Information. In *Proceeding of the SACMAT'04 Yorktown Heights,California,USA*, June 2004.

[4] Richard James Glassey and Robert Ian Ferguson. SpaceSemantics: An Architecture for Modeling Environments. In *Proceedings of the Workshop on Location-Aware Computing (UbiComp)*, Seattle,Washington,USA, October 2003.

[5] Robert W. Frischholz and Ulrich Dieckmann. BioID: A Multimodal Biometric Identification System. *Computer*, 33(2):64–68, February 2000.

[6] Jay Cadman. Deploying Commercial Location-Aware System. In *Proceedings of the Workshop on Location-Aware Computing (UbiComp)*, Seattle,Washington,USA, October 2003.

[7] D. Ferraiolo et al. Proposed NIST Standard for Role-Based Access Control. *ACM Transaction on Information ans System Security*, 4:224–274, 2001.

[8] D. Fox et al. Bayesian Techniques for Location Estimation. In *Proceedings of the Workshop on Location-Aware Computing (UbiComp)*, Seattle,Washington,USA, October 2003.

[9] D. Lee et al. Data Management in Location-Dependent Information Services. In *Context-Aware Computing*, July 2002.

[10] E. Bertino et al. An Access Control Model Supporting Periodicity Constraints and Temporal Reasoning. *ACM Transaction on Database Systems*, 23(3):231–285, 1998.

[11] E. Bertino et al. GEO-RBAC: A Spatially Aware RBAC. *ACM Transactions on Information Systems and Security*, 00(00):1–34, March 2006.

[12] J. Joshi et al. Temporal hierarchies and inheritance semantics for GTRBAC. In *Proceedings of the Symposium on Access Control Models and Technologies*, pages 74–83, June 2004.

[13] J. Joshi et al. A Generalized Temporal Role-Based Access Control Model. *IEEE Transactions on Knowledge and Data Engineering*, 17(01):4–23, January 2005.