Requirements and Challenges of Location-Based Access Control in Healthcare Emergency Response

- Position paper -

Carmen Ruiz Vicente
Aalborg University
carmrui@cs.aau.dk

Michael Kirkpatrick
Purdue University
mkirkpat@cs.purdue.edu

Gabriel Ghinita
Purdue University
gghinita@cs.purdue.edu

Elisa Bertino
Purdue University
bertino@cs.purdue.edu

Christian S. Jensen
Aalborg University
csj@cs.aau.dk

ABSTRACT

Recent advances in positioning and tracking technologies have led to the emergence of novel location-based applications that allow participants to access information relevant to their spatio-temporal context. Traditional access control models, such as role-based access control (RBAC), are not sufficient to address the new challenges introduced by these location-based applications. Several recent research efforts have enhanced RBAC with spatio-temporal features. Nevertheless, the state-of-the-art does not deal with mobility of both subjects and objects, and does not support complex access control decisions based on spatio-temporal relationships among subjects and objects. Furthermore, such relationships change frequently in dynamic environments, requiring efficient mechanisms to monitor and re-evaluate access control decisions. In this position paper, we present a healthcare emergency response scenario which highlights the novel challenges that arise when enforcing access control in an environment with moving subjects and objects. To address a realistic application scenario, we consider movement on road networks, and we identify complex access control decisions relevant to such settings. We overview the main technical issues to be addressed, and we describe the architecture for policy decision and enforcement points.

1. INTRODUCTION

The availability of mobile devices with positioning capabilities fostered the development of location-based applications that allow users to access information relevant to their spatio-temporal context. For instance, visitors of a museum may be able to access some electronic on-line guide system, as long as they are situated within the museum's perimeter. Such applications introduce specific security challenges that reach beyond the capabilities of traditional access control systems, such as Role Based Access Control (RBAC). Several models have been proposed [1, 2, 4] that extend RBAC to allow the specification of policies in which the access decision is determined by the spatio-temporal context of subjects and objects. However, these models relate the role extents to fixed locations, restricting the policies to static zones. Therefore, they may not be suitable for dynamic mobile environments where both subjects and objects are continuously moving. In addition, in certain location-based application scenarios, the access control decision may depend on complex spatio-temporal relationships among subjects and objects.

In this position paper, we outline an access control framework for healthcare emergency response that illustrates the requirements of access control in a dynamic environment with mobile subjects and objects. In this scenario, a patient who requires an emergency triggers an *event*, that results in the creation of an emergency care request in the system. Such an event can be generated by the patients themselves, or by an automatic system designed to alert emergency response teams (e.g., in-vehicle GPS-enabled units such as On-Star provide automatic car crash notification). The record inserted in the system contains a number of *event attributes*, for instance, symptoms, vital signs, as well as the age and gender of the patient. Based on the patient identifiers (e.g., cell phone number, or identifier of the reporting On-Star unit) additional information can be associated with the emergency record request, such as the patient's medical history. The location of the emergency, the patient symptoms, medical history, etc., represent *objects* that need to be accessed by several categories of mobile *subjects*, such as medical doctors, ambulance personnel or medical-trained volunteers. Note that the objects may also be mobile. For instance, a patient suffering from a heart attack may be in a moving vehicle drove by a family member.

An event may refer to a single patient, or may involve several patients, e.g., in the case of a car accident. In response to an event, the emergency response system must find subjects that are authorized to access the event's attributes and provide on-site help to patients. We identify two key requirements. First, as medical information is sensitive, the system needs to ensure that access to confidential data is thoroughly controlled. Note that distinct subjects may have different privileges in information disclosure: for example, doctors may be allowed to access medical history, whereas volunteers should not learn confidential information about the patient that it is not related to the current event. Second, an emergency requires fast response. The authorized subjects should be able to arrive to the location of the emer-

gency within a maximum period of time, thus the decision about which subjects to authorize must take into account the distance between the event location and the subjects.

Two representative types of location constraints are *range* and *nearest-neighbor* constraints. For instance, the former corresponds to requirements such as, *"A subject is only allowed to access an event record if its distance to the event site is less than one mile,"* whereas the latter addresses requirements such as, *"A subject is only allowed to access an event record if it is the nearest subject to the event site."* In addition, the position of both subjects and objects may change over time, therefore the spatial relationship between subjects/objects must be continuously monitored. Note that spatial and non-spatial (e.g. level of medical education, specialization, required equipment, etc.) constraints may be combined. For instance, if a volunteer and a doctor are located at the same distance, only the doctor should be granted access. Such complex access control decisions require specialized mechanisms for the design of policy enforcement and decision points (PEP/PDP).

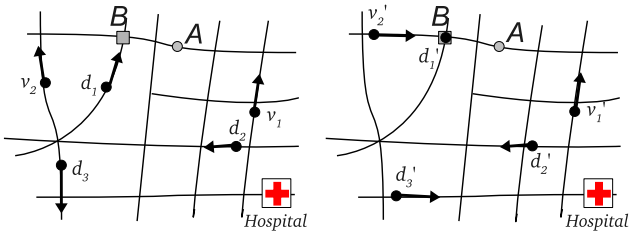

Figure 1: Emergency Response Scenario

The example in Figure 1(a) shows several volunteers v_i and off-duty doctors d_i driving around a city, and a hospital located nearby. Consider that a car accident is reported at location A, and the car crash victim has a pre-existing heart condition and suffers from diabetes. In response to the event, the emergency response system takes the following actions: First, an ambulance is requested from the nearest hospital. Next, doctor d_1 who is nearest to the accident site is notified, and starts moving towards the victim to perform a preliminary evaluation and treatment before the ambulance arrives. The notification sent to d_1 includes the coordinates of the accident site together with some basic information about the nature of the emergency, the victim's age and pre-existing conditions.

As d_1 moves towards the accident site, a traffic jam may occur at intersection B, so d_1 is no longer able to reach the patient in time (Figure 1(b)). The system considers additional subjects, such as volunteers v_1 and v_2. Although v_2 is closer to the accident site than v_1, the route of v_2 crosses the traffic jam, so v_1 is chosen instead. In addition, doctor d_2 is selected, and both are granted access to the event record. However, the doctor d_2 is granted permission to access more data about the victim, such as medical history. Such information is not made available to v_1.

While the volunteer v_1 and the doctor d_2 stabilize the patient, the hospital may not find an available ambulance. Then, the system may authorize d_2 to drive the patient to the hospital, if the patient is in stable condition. Otherwise, an additional doctor (e.g., d_3) may be authorized to assist in the emergency. Note that the system may decide the amount of data disclosed to a subject based on proximity to the accident site. This prevents the disclosure of sensitive information to subjects that are not likely to participate in the emergency, such as subjects that are delayed due to environment conditions (as d_1 in the example above). However, some general details can be provided to such entities, in order to provide a backup plan. This approach applies to location information as well: for instance, only coarse-grained location information may be disclosed to remote subjects, whereas accurate location data is sent only to nearby subjects.

The objective of this position paper is to identify the most representative requirements and challenges of location-based access control in a dynamic environment with mobile subjects and objects. We investigate access control decisions and enforcement with respect to a combination of spatial and non-spatial (e.g., role-based) constraints. Although our focus is on healthcare emergency response, many of the aspects discussed are relevant to other classes of location-based applications as well (e.g., public transportation). We emphasize that our work discusses initial directions and identifies key challenges in location-based access control, but the realization of a complete access-control mechanism is beyond the scope of this paper. However, we do give an overview of the architectural components involved, as well as the main functionality fulfilled by each component.

The rest of the paper is organized as follows. Section 2 describes the event and policy models. Section 3 illustrates the proposed architecture of the system and the tasks performed in each component. Section 4 concludes the paper and outlines future research directions.

2. EVENT AND POLICY MODELS

As mentioned in Section 1, our envisioned access-control system is event-centric, i.e., objects are generated following the occurrence of an event such as a medical emergency (e.g., car accident). After object creation, the system must decide which subjects (e.g., medical doctors, volunteers, etc.) are granted permission to access the event-related data based on a mix of spatial and role-based constraints.

2.1 Event Model

An **event** is specified as a tuple:

$$E = \{location, category, maxResponseTime, attributes\}$$

where

- *location* indicates the position of the event.
- *category* describes the event type.
- $maxResponseTime$ indicates the maximum allowed time in which the event must be handled (e.g., the maximum allowed time for the responders to arrive).
- *event_attributes* is a list of event characteristics.

Depending on the required functionality of the system, the *location* may be specified either in physical terms (e.g., GPS coordinates) or logical terms (e.g., at the corner of Main Street and Elm Drive). The use of *category* determines the number of responders and their roles. For example, the organization may stipulate that $category = carCrash$ requires two *doctors* and one *volunteer*.

Indicating a $maxResponseTime$ authorizes the system to relax the data confidentiality and other constraints if the

request cannot be fulfilled. This mechanism allows flexibility for events that require immediate focus and response, such as severe automobile crashes. However, we also assume that most events will provide this piece of data to ensure that help is provided within a reasonable timeframe.

Event attributes are provided when the event is triggered (e.g., the cell phone number where the 911 call originated). Based on such attributes, additional data objects can be originated, such as the age and gender of the person who owns the cell phone (such data is relevant if the victim herself placed the call).

2.2 Policy Model

The system must support both traditional (e.g. role-based) and spatio-temporal policies. Traditional policies can integrate other access control or RBAC rules, such as *"A volunteer is not allowed to access a patient's previous medical records."*

Spatio-temporal policies indicate constraints and obligations that cannot be expressed in traditional policies. Examples of spatio-temporal policies include *"A volunteer is not allowed to access a patient's information for more than 10 minutes"* or *"A volunteer is not allowed to access a patient's symptoms if there is a doctor nearby."* Note that, the latter policy must be dynamically instantiated to refer to specific elements in an event. That is, the rule should only apply for a volunteer and a doctor that are part of the same emergency call, thus belonging to the same event context. Rules that are not related to a specific event are described as *global*. The attribute *scope* will distinguish between event and global policies.

An access authorization policy is expressed as a tuple:

$$P = \{scope, subject, object, feature, granularity, ST\}$$

where

- *scope* is the context in which the rule applies (*global* or *event*).
- *subject* denotes the entity (e.g. *volunteer*) to which the authorization is applied.
- *object* corresponds to the protected entity (e.g., patient whose medical record is accessed).
- $feature$ denotes the protected information (e.g., medical history, location, symptoms, diagnosis).
- *granularity* symbolizes the granularity of the spatio-temporal feature, or it can be *null*. For instance, in the case of location information, we could specify an accuracy level of *neighborhood* or *street*.
- $ST = \{ST_1, ..., ST_k\}$ represents the list of spatio-temporal constraints that control the policy.

A spatio-temporal constraint represents a feature that has a spatial and/or a temporal dimension. Each spatio-temporal constraint ST_i can be mapped to the following schema:

$$ST_i = \{type, attribute, S, object, sign, time\}$$

The elements of the schema are defined as follows:

- *type* is the type of the constraint. We consider range constraint RC, k-nearest neighbor constraint kNN, reverse k-nearest neighbor constraint $RkNN$, or *null* if there is no spatial dimension.
- *attribute* denotes an additional parameter of the constraint. For a range constraint, this represents travel time, and k for (reverse) k-nearest neighbor constraints.
- S represents the list of *subjects* that participate in the constraint.
- *object* corresponds to the subject, object, or location under consideration relative to S.
- *sign* indicates if the constraint is *positive* or *negative*.
- *time* specifies the time validity (expiration), which varies within the interval $(0, \infty)$.

As a example of a policy, consider the rule, *"A volunteer is not allowed to access a patient's symptoms if there is a doctor in less than 10 meters."* This policy would be expressed as the tuple

$$P = \{\text{"}event\text{"}, \text{"}volunteer\text{"}, \text{"}patient\text{"}, \text{"}symptoms\text{"}, \phi, S_1\}$$

The spatio-temporal constraint would be expressed as

$$S_1 = \{\text{"}RC\text{"}, \text{"}10m\text{"}, \text{"}volunteer\text{"}, \text{"}doctor\text{"}, \text{"}negative\text{"}, \infty\}$$

Although the subjects in both the policy and the constraint are the same in this example, this will not always be the case. That is, to prevent redundant constraint expressions, S_1 could also be used in a policy tuple describing the doctor's permissions given that constraint.

3. SYSTEM ARCHITECTURE

3.1 Overview

In traditional access control systems, a user initiates a request and submits his/her relevant credentials, such as a username and password. The system then evaluates the combination of the subject, the object requested, the credentials provided, and the relevant policies, and grants or denies access accordingly. In our proposed approach, a request is automatically generated in response to an event. The system then grants permissions to subjects, given the current spatio-temporal environment and policy constraints.

There are a number of key novelties in our approach. First, the system can grant permissions to multiple subjects simultaneously as part of a single request. Second, the subjects receiving access are not known when the request is initiated. That is, the system must determine the most appropriate subjects in response to a request. Finally, the dynamic nature of spatio-temporal environments necessitates continuous monitoring of subjects and objects. For example, if the traffic congestion increases in one location, the system may need to revoke or reassign authorizations in response. Despite these differences, though, we find that the traditional notions of Policy Decision Point (PDP) and Policy Enforcement Point (PEP) are still applicable.

In addition to the PDP and PEP components, we also introduce the Policy Database (PolicyDB) and the Moving Objects Database (MOD). PolicyDB contains the access control policies as specified by Section 2.2. On the other hand, the positions of objects and subjects are indexed by the Moving Objects Database, where the weights of the edges represent the driving time of each road segment (which forms a travel time network [3]). In our application, where a timely response to an emergency is essential, the

travel time is a better choice than the network distance, as this metric captures and estimates the *time* that an entity requires to travel to a certain location within the network. Moreover, we assume that the database is updated with real-time traffic information.

For the sake of simplicity, we consider that the MOD communicates directly with the users. However, a more realistic assumption would be that the position of the users may be given by a external location provider, and the database would be populated through an interface.

Figure 2 illustrates the interaction among PEP/PDP and the rest of the system components when a new event is created (steps 1 and 2). As stated above, the PDP decides which entities will be granted access to the event information (step 3). In order to do this, PDP queries a MOD to find subjects who have the required skills and are nearby the event location (the number and skills of the subjects that are required in each case is determined by the type of the event). Note that the MOD may need to provide more subjects than are required by the event type, in case one or more of the subjects is unable or unwilling to respond to the event. Then, the access policies involving those subjects are obtained by querying PolicyDB (step 4). As described in Section 2, these policies can contain static (organizational) and spatio-temporal rules that need to be continuously monitored by the PEP. Therefore, the PEP receives the subjects that will be involved in the event and the constraints (step 5). Finally, the subjects are notified (step 6) and the monitoring of the constraints is started (step 7).

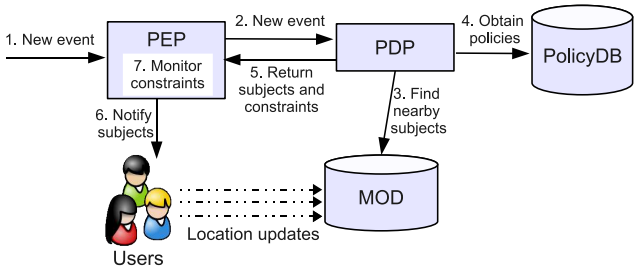

Figure 2: System Architecture

3.2 Access Control Decision

The PDP needs to perform two main decision tasks. When an access request arrives in the system, it makes the decision of whether the access is granted or denied, based on the policies that refer to the subject. The second task, as mentioned previously, finds nearby and suitable subjects that will respond to a new event. The first operation corresponds to a traditional access control system and will not be considered in this paper. For the latter, the PDP performs an incremental expansion of the network from the location of the event using the Incremental Network Expansion algorithm (INE) [6]. This algorithm retrieves the nearest neighbors from a starting point, incrementally and ordered by distance to the point. For our case, we will assume that the function $retrieve\_Next\_NN\_INE$ gets the next nearest neighbor that the INE algorithm would return.

In addition to the set of subjects, the PDP builds another set that corresponds to a back-up plan. This set consists of the subjects that will be deployed if any of the subjects in the primary plan fail to reach the emergency location. For example, a subject may not reply to the notification in a reasonable time or get delayed due to traffic conditions. If this is the case, the system can quickly choose another subject without re-evaluating the entire request by consulting the back-up plan.

The pseudocode for the complete algorithm is shown in Figure 3. Our algorithm performs as follows. First, given the category of the event, the type and number of required entities is retrieved for both the primary and the backup plan (steps 2 and 3). These values are stored in pairs $[t_i, n_i]$, where $t = type$ and $n =$ number of subjects of that type still required (i.e., this value will be maximum at the beginning of the execution, and will decrease when subjects are found). The value k represents the maximum number of subject types in the system. Then, INE is initialized (step 4). The algorithm executes while not all the subjects have been found (represented by the condition in step 5). For the next nearest neighbor found (step 6), the type is extracted (step 7). If there is a subject of that type required for either the primary plan (lines 8-10) or the back-up plan (lines 11-13), the subject is added to the corresponding set. The algorithm will finish when all the required subjects are found.

FindSubjects
Input: event location l and category c
Output: set of subjects for the primary plan S_p, set of subjects for the back-up plan S_b
1. $S_p, S_b = \emptyset$
2. $p = \{[t_1, n_1]...[t_k, n_k]\} = entitiesPrimaryPlan(c)$
   /* t_i = subject type, n_i = number of subjects */
3. $b = \{[t'_1, n'_1]...[t'_k, n'_k]\} = entitiesBackupPlan(c)$
4. $initializeINE(l)$
5. **while** $\exists n_i > 0 \in p \vee \exists n'_i > 0 \in b$
   /* while more subjects are needed */
6. $s = retrieve\_Next\_NN\_INE()$ /* uses INE algorithm */
7. $t = typeOf(s)$
8. **if** $p[t].n > 0$ **then**
   /* if the plan needs a subject of that type, add it */
9. $S_p = S_p \cup \{s\}$
10. $p[t].n \leftarrow p[t].n - 1$
11. **else if** $b[t].n > 0$ **then**
   /* if the backup needs a subject of that type, add it */
12. $S_b = S_b \cup \{s\}$
13. $b[t].n \leftarrow b[t].n - 1$

Figure 3: Find nearby subjects

Recall that the distance used in $retrieve\_Next\_NN\_INE$ is measured as the expected time to travel from the current position to the location of the accident, thus the distance considered is the travel time distance instead of road network distance.

If the event has a $maxResponseTime$, the algorithm is modified to maintain a queue containing the discarded subjects (after line 13). In addition, the loop terminates when the distance (travel time) from the last nearest neighbor is bigger than $maxResponseTime$. If there are no results found when the expansion reaches $maxResponseTime$, or if the results are not enough, the entities belonging to the discarded queue will be considered. We can assume that in these cases the confidentiality can be relaxed in order to provide a fast response time. For example, a volunteer may be sent to the emergency instead of a doctor. Moreover, the system could create some range constraints that trigger an alert when some of the required subjects are in the proximity.

Note that we have illustrated a simplified version of the algorithm in which the roles are not interchangeable. That

is, the algorithm would look for a volunteer even if there is a doctor nearby that can perform the same (and more) operations. If roles can be ordered or organized in a hierarchical manner, the algorithm should be modified to identify the subjects in a more efficient way.

3.3 Constraint and Policy Processing

After the subjects that belong to the primary and the back-up plan are found, the PDP queries the PolicyDB to retrieve the policies affecting the subjects in both plans. As mentioned in Section 2.2, some of these policies will need to be instantiated for the specific objects of the event. From the set of retrieved policies, the spatio-temporal constraints are extracted and sent to the PEP, which will start the monitoring process.

We identify two kinds of location constraints:

- nearest neighbor constraints (e.g. *"Volunteer A is a 2-nearest neighbor of the accident site"*)
- proximity (range) constraints (e.g. *"Volunteer A is 1 mile away from the accident site"*)

Note that, as the final decision regarding an access request is done by the PDP, both PEP and PDP need to maintain the list of active constraints with their state (however, recall that the active monitoring of the constraints is performed only by the PEP). For example, the policy, *"A volunteer is not allowed to access a patient's symptoms if there is a doctor nearby,"* will initially allow the volunteer to access the information, but will revoke the access when a doctor arrives. Similarly, the monitoring of the constraint of the policy, *"A volunteer is allowed to access a patient's symptoms when the arrival time to the patient is less than 1 minute,"* will allow the system to send a notification to the subject when s/he is close enough, without the subject requesting it. In other cases, for example if the system detects that a subject is delayed due to a traffic jam, the PDP may choose to send another subject to the emergency.

The PEP monitors the spatio-temporal constraints and notifies the PDP when a constraint changes state (from hold to not hold, and vice versa). The factors that may change the state of a constraint, and thus need to be monitored, are the following:

1. movement of the subjects and/or the constraints
2. changes in environment conditions (e.g. traffic jams, represented as a change on the weight of the edges)
3. time expiration of the constraint
4. requirements of the event (e.g. change of the *category* of the event)
5. a subject becomes not available (subject changes status from *available* to *not available*)

The location constraints at the PEP can be monitored as described in [5]. In this work, the authors propose two methods to monitor kNN in road networks, the first one maintains the query results by processing only updates that may invalidate the current NN sets, while the second one follows the shared execution paradigm to reduce processing time. Although their research is focused on kNN, the structures presented can be applied in range queries and RkNN queries as well. Moreover, the methods support object and query movement patterns and modifications of edge weights (therefore, supporting factors 1 and 2).

To monitor the time expiration (factor 3), the PEP can maintain a queue ordered by expiration time. When a constraint reaches its expiration time, it is removed from the system and the PDP is notified.

For the last two factors (factors 4 and 5), the PEP can maintain two indices, based on the events (factor 4) and the subjects (factor 5). When these entities change status, all the constraints in which they are involved are eliminated from the system (and the PDP is notified as in the previous cases).

The PDP maintains a list of the active policies and constraints and the current state. This list is queried when a subject requests access to an object of the system. When a constraint changes state, the factor that originated this change is examined. If the reason was a change of the event requirements (factor 4), the whole plan for the event needs to be re-evaluated. For the rest of the cases, the system may make use of the back-up plan.

4. FUTURE WORK

In this position paper, we have identified several challenging research issues in location-based access control for helthcare emergency response. In the future, we plan to formalize the proposed event and policy models, and to analyze in depth the functionality that must be implemented in the PEP and PDP. We also envision extending our models to take into account uncertainty in reporting location data, as well as the privacy issues that arise when subjects are required to report their location to the PEP.

5. REFERENCES

[1] S. Aich, S. Mondal, S. Sural, and A. K. Majumdar. Role based access control with spatiotemporal context for mobile applications. *Transactions on Computational Science IV: Special Issue on Security in Computing*, pages 177–199, 2009.

[2] S. Aich, S. Sural, and A. K. Majumdar. Starbac: Spatio temporal role based access control. In *OTM Conferences (2)*, volume 4804 of *Lecture Notes in Computer Science*, pages 1567–1582. Springer, 2007.

[3] W.-S. Ku, R. Zimmermann, H. Wang, and C.-N. Wan. Adaptive nearest neighbor queries in travel time networks. In Proc. of ACM GIS, pages 210–219, 2005.

[4] M. Kumar and R. E. Newman. Strbac - an approach towards spatio-temporal role-based access control. In *Communication, Network, and Information Security*, pages 150–155, 2006.

[5] K. Mouratidis, M. L. Yiu, D. Papadias, and N. Mamoulis. Continuous nearest neighbor monitoring in road networks. In Proc. of VLDB, pages 43–54, 2006.

[6] D. Papadias, J. Zhang, N. Mamoulis, and Y. Tao. Query processing in spatial network databases. In Proc. of VLDB, pages 802–813, 2003.