AN ACCESS-CONTROL MODEL FOR MOBILE COMPUTING WITH SPATIAL CONSTRAINTS

Location-aware Role-based Access Control with a Method for Consistency Checks

Michael Decker
Institute AIFB, University of Karlsruhe (TH), Englerstr. 11, 76 128 Karlsruhe, Germany

Keywords: Mobile Information Systems, Access Control, Location-based Services, Mobile Security.

Abstract: Some of the most salient challenges that come along with the employment of mobile information systems stem from security issues: portable devices like PDAs, smartphones and notebooks easily get stolen or lost and wireless data transmission could be eavesdropped, so that unauthorized individuals gain access to confidential resources. One approach to tackle these problems is location-aware access control, i.e. based on knowledge about the user's position the information system can decide if access to a resource should be granted or not. For example a nurse using a PDA should only be allowed to access confidential patient data while staying on the premises of the hospital. In our article we present a data model for location-aware access control based on the concepts of roles. Using our model it is possible to assign location restrictions to several entities, e.g. to users, to roles or permissions. We also propose a method to analyze the consistency of spatial constraints expressed by an instance of our model.

1 INTRODUCTION

Nowadays it is possible to build portable computers like PDAs and smartphones which are more powerful than stationary systems were a few years ago. At the same time technologies for wireless data communication like GPRS, WiFi or UMTS were developed and are nowadays widely available. Based on these two technologies mobile information systems (MIS) can be realized to make computer support available almost anywhere and anytime. These MIS have a great potential for many novel application scenarios, especially with regard to support mobile workers. But the utilization of mobile technologies comes along with serious security challenges, because due to their portability and small size mobile devices easily get stolen or lost. If unauthorized individuals get into the possession of a mobile device they can gain access to confidential resources (e.g. health records, telephone numbers of board members, financial data).

In our work we employ the concept of location-aware access control to address this problem field. The basic idea is that for the decision wether a particular user is allowed to access a given resource the current location of the mobile device is regarded. For example the access to the customer database using a mobile device should only be granted if the mobile user just stays on the premises of the company; downloading a confidential document to a notebook computer could be denied if the user stays abroad. To determine the location of a mobile device there are several technologies available (e.g. GPS or CellID), see Küpper (2005) or Hightower & Borriello (Hightower and Borriello, 2001) for an overview.

To implement location-aware access control a formalism is required to express which resources can be accessed at which location by which users. Such a formalism is called "access control model" (ACM). In our article we will introduce a location-aware ACM that is based on the notion of role-based access control (RBAC). Location-restrictions can be assigned to several elements in this model to provide a great degree of versatility. We also introduce methods to analyze model instances based on geometric operations; as far as we are aware this concept is novel and not mentioned in pertinent literature yet.

The remainder of the article is structured as follows: We will first give a short introduction to access control models in section 2.1 before we introduce our own model and give an example scenario in section 2.2. A formal presentation of the model will be given in section 3; this comprehends also a simple location model. Based on this formalization we describe some ways to analyze model instances by geometric con-

Decker M. (2008).
AN ACCESS-CONTROL MODEL FOR MOBILE COMPUTING WITH SPATIAL CONSTRAINTS - Location-aware Role-based Access Control with a Method for Consistency Checks.
In *Proceedings of the International Conference on e-Business*, pages 185-190
DOI: 10.5220/0001911001850190

siderations in section 4. We also implemented a small system to visualize the location constraints of model instances and to run the analyzing algorithms; this implementation will be sketched in section 5 before we conclude in section 6.

2 ACCESS CONTROL MODELS

2.1 Basics

Access Control Models (ACM) are formal models to describe under which conditions an information system should grant access to resources. There are three main groups of ACM (Samarati and di Vimercati, 2001): mandatory access control (MAC), discretionary access control (DAC) and role-based access control (RBAC).

Using MAC systems subjects as well as objects are classified according to their level of confidentiality (e.g. ”top secret”, ”secret”, ”confidential”, ”unclassified”). Then there are rules that impose restrictions like ”a subject should only be allowed to read an object if the subject’s classification (or *clearance*) is at least as high as the one of the object.” One prominent example for such MAC is the Bell-LaPadula-model. DAC models follow the owner principle: each object under protection of the access control model belongs to an owner. This owner can grant rights to perform certain operations on that object to other subjects. Examples for DAC-models are those used by operating systems, e.g. on Unix a user can grant the right to read a file he just created to users of a so called group. For the article at hand the most important group of ACM is RBAC (Ferraiolo et al., 2003). The basic notion is to assign permissions not to individual subjects but rather to so called roles. Roles represent job descriptions or positions within an organization. So if Alice is hired as new ”manager” the permissions required for this job (e.g. read payroll file, approve order) don’t have to be assigned individually; we just have to assign the ”manager” role to Alice’s user account.

Almost all location-aware access control models that can be found in literature are extensions of RBAC. The basic idea to obtain a location-aware RBAC model is to switch particular elements of the RBAC on or off depending on the user’s location, e.g. roles (Bertino et al., 2005), the role-permission-assignment (Hansen and Oleshchuk, 2003) or the user-role-assignment (Chandran and Joshi, 2005).

2.2 A Location-Aware Access Control Model based on RBAC

In this subsection we will introduce an RBAC-based model for location-aware access control which is depicted in figure 1 as entity-relationship diagram. The cardinalities in the diagram are given in the form of (min, max)-values. For example ”$(1,*)$” as cardinality of entity ”role” in the ”role-location”-relationship means that each role entity occurs at least once in the relation representing that assignment. Since ”$max = *$” a role entity may occur as many times as needed in that relation.

Each user can be assigned to several roles. Roles again are collections of permissions. Permissions in RBAC-models are usually considered as symbols that have to be interpreted for the respective deployment. In the figure we interpret a permission as a set of operations (e.g. read, write, update) that can be performed on a given class of objects. An object class might represent a class of documents like ”customer record”. For each object class there might be several object instances, e.g. ”customer record for Mr. Meyer” and ”customer record for Ms. Miller”. The entity location represents a location like a city, a room or a country. We will later give a formal definition for what a location is.

Several components of the model have a relation to at least one location. A component is only enabled if the respective user stays at that location, e.g. if the role ”manager” is restricted to the locations ”local branch Munich” and ”local branch Lisbon” this means the role can only be used when the user is in Munich or Lisbon. If there should be no location restriction we just assign the location ”universe” (which covers the whole reference space) to the respective component. The remaining points for the assignment of location restrictions are:

- Restricting a user to locations means that the user can use the system only at certain locations, e.g. we can restrict a user to ”southern Spain” because he has only to work in this region for the organization or he bought a software licence that covers only this part of the country.
- Restricting a permission to locations means that this permission should only be accessible if the user is at the specified locations, no matter how powerful his role is. An example is that ”access payroll” should only be allowed while being on the company’s premises.
- It is also possible to assign location restrictions for individual objects, e.g. the object ”research report no. 123” is not allowed to be accessed outside a

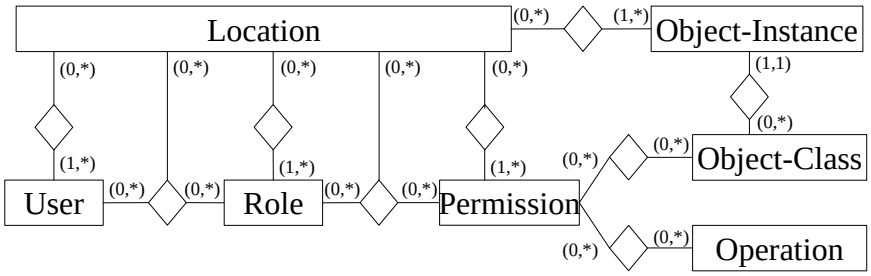

Figure 1: Access Control Model as Entity-Relationship-Diagram.

certain country because espionage is feared.

- Attaching a location restriction to the user-role-assignment means that the given user is allowed to play that role only at certain locations, e.g. a mobile worker is only allowed to play the role ”travelling salesman” in his personal sales district.
- Attaching location restrictions to the role-permission-assignment means that the given permission can be only performed using that role when the user is at certain locations, e.g. we could have a role ”manager” that encompasses the permission ”approve order” but only within Europe.

We don’t assume that for a real-world instance of our model all six location restrictions will be used; but having six possible components where location-restrictions can be assigned to gives the freedom to choose the points that are most appropriate or natural for the respective scenario. If location restrictions are assigned to more than one point in the model it might be necessary to calculate the intersection of different location restrictions to obtain the area where the user is indeed authorized to perform a certain permission.

To exemplify our model we give the following example: The scenario encompasses a company with three local branches that sends mobile employees to facilities in a given area (e.g. to deliver goods or to perform some kind of maintenance work). In figure 2 the location restrictions for four roles (dotted lines) and three permissions (solid lines) are depicted as rectangles: p_1 is the permission to access the payroll file; since this is confidential information this should only be possible while staying at one of the three local branches. p_2 is the permission to access customer data; this permission is restricted to three service districts where the facilities are that have to be served by the company’s employees. p_3 is a navigation service licensed from an external service provider; this service is accessible only in an area that covers most of the three services districts (licensing for a larger area would be more expensive).

Roles r_1 to r_3 are roles for mobile workers (e.g.

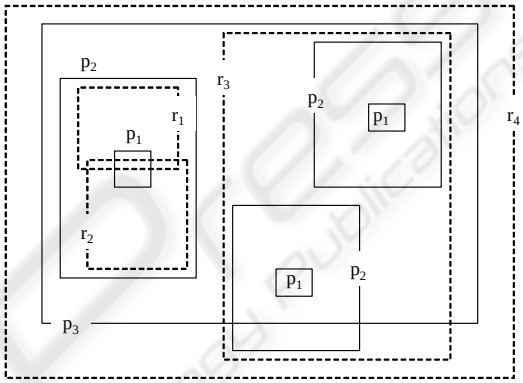

Figure 2: Example for spatial constraints.

service technicians): r_1 is for the northern part of district 1, r_2 for the southern part of that district. The number of jobs to perform in districts 2 and 3 isn’t big, so a single role r_3 is sufficient for both. r_4 is the manager’s role who has to visit facilities in all restricts, e.g. to handle complaints or to acquire new customers. Therefore this role’s location constraints encompass all districts. Role r_3 has the permissions p_2 and p_3; this means that a user having r_3 can access customer data (p_2) in the areas where the rectangle of r_3 intersects with the two rectangles for permission p_2; the navigation service is only accessible where the rectangle for r_3 intersects with the rectangle for p_3.

3 FORMAL DESCRIPTION OF THE MODEL

3.1 Location Model

To give a formal description of our model and to describe spatial analyzes to be performed on instances of the model we need a formalization of the concept of a ”location”:

We denote the area to be covered by the ACM as reference space or *universe*. Depending on the scope

of the MIS *universe* may be the premises of a company, a country or earth's surface. All other locations l like cities, regions, buildings or rooms are real subsets of the *universe*: $l \subset universe$. The set of all locations l and the *universe* is named *locs*:

$$locs = \{l | l \subseteq universe\}$$

With regard to the later implementation we demand that all locations are polygons because locations will be stored in a database system. Since we can approximate every area by a polygon with arbitrary precision this is a weak restriction. If PT denotes the location "Portugal" and LIS the location "LISBON" we get: $locs = \{universe, PT, LIS, \dots\}$.

For each location or set of locations we can calculate the covered area with the function $area()$, which will return a real number greater than zero:

$$area() : 2^{locs} \longrightarrow \mathbb{R}_+$$

If two locations of an input set overlap the intersecting area will be counted only once.

For two locations $l_1, l_2 \in locs$ we can calculate the intersection and the union. When calculating the intersection of two polygons we may obtain one polygon, several polygons or the empty set as result, so the intersection operator's range is the power set of *locs*:

$$l_1 \cap l_2 \longrightarrow 2^{locs}$$

The union set of l_1 and l_2 can consist of one location (if l_1 and l_2 intersect or touch each other) or two locations, but the empty set isn't possible:

$$l_1 \cup l_2 \longrightarrow 2^{locs} \setminus \emptyset, \quad |l_1 \cup l_2| \in 1, 2$$

There is also the set C of location classes. Location classes are semantic categories for locations, e.g. $C = \{cities, countries, buildings, \dots, unclassified\}$. Each location in *loc* is mapped to exactly one location class in C:

$$class() : locs \longrightarrow C$$

One class in C is "unclassified" which is the "dummy class" to be used if there is no reasonable semantic category for a location. Location classes are a mean to support human users when browsing through the set of pre-defined locations.

3.2 Core Model

We have a set for each of the main entities of the ACM: the set of users $U = \{u_1, u_2, \dots\}$, the set of roles $R = \{r_1, r_2, \dots\}$ and the set of permissions $P = \{p_1, p_2, \dots\}$. For the sake of brevity we don't consider the classes and objects behind a permission. Further there are the sets UR and RP: if user u_0 has role r_0 then $(u_0, r_0) \in UR$ and if permission p_0 is assigned to role r_0 then $(r_0, p_0) \in RP$. Together these sets form the set of main entities E:

$$E = U \cup R \cup P \cup UR \cup RP$$

The function $loc()$ takes as input one of these entities and returns the subset of locations the respective entity is restricted to:

$$loc() : E \longrightarrow 2^{locs} \setminus \emptyset$$

The locations returned by $loc()$ are also denoted as a user's, a role's resp. a permission's area. Example:

$$loc(u_1) = \{PT, MUN\}$$

This means the user u_1 is allowed to use the system in Portugal or Munich. If an entity isn't restricted to a location at all $loc()$ just returns the *universe*. If we want to express that user u_2 can activate role r_2 only in Lisbon this would be:

$$loc((u_2, r_2)) = \{LIS\}$$

4 SPATIAL ANALYSES OF MODEL INSTANCES

In this section we will describe some approaches to perform spatial analyses on instances of our model to find inconsistent configurations. For the sake of simplicity we don't consider location restrictions assigned to user-role-assignments and role-permission-assignments.

4.1 Coverage

Coverage is a function that takes one entity from the set of all users, roles and permissions as first argument (pivot entity) and one of the target categories $T = \{$"*user*","*role*","*permission*"$\}$ as the second argument:

$$cover() : (U \cup R \cup P) \times T \longrightarrow 2^{locs}$$

If the pivot entity is of the type specified by the target category then $cover()$ just returns the location restriction of the pivot entity, e.g.

$$cover(u_0, \text{"}user\text{"}) = loc(u_0)$$

If the pivot entity stands "more left" than the target category in figure 1 then the result is the set of locations where the pivot entity can activate at least one entity of the target category:

- $cover(u_0, \text{"}perm\text{"})$ returns the area where user u_0 can activate at least one permission.

- $cover(u_0, "role")$ returns the area where user u_0 can activate at least one role.
- $cover(r_0, "perm")$ returns the area where with role r_0 at least one permission can be activated.

In the opposite case (the pivot element is on the right-hand side of the target category) the locations returned are the area where the pivot element can be activated by at least on entity of the target category:

- $cover(p_0, "role")$ returns the area where permission p_0 can be activated by at least one role.
- $cover(p_0, "user")$ returns the area where permission p_0 can be activated by at least one user.
- $cover(r_0, "user")$ returns the area where role r_0 can be activated by at least one user.

Due to space limitations we will only discuss $cover(p_0, "user")$ in detail, which is also the most interesting case: if permission p_0 is location-restricted to certain locations (e.g. area where customers have to be served, area for which a software license was bought) we may want to check if every point of the permission area is covered by at least one user, i.e. is there a part of the permission area where no user according to his roles and his own location restriction can use that permission? If this case is given it would mean that we had customers that cannot be served or that we bought the software license for a too big area. The formal definition is as follows:

$$cover(p_0, "user") =$$
$$loc(p_0) \cap \left(\bigcup_{(r_0,p_0)\in RP} cover(r_0, "user") \right)$$

So we have to calculate the intersection of the permission area with the union of all coverages for roles that are assigned to p_0. The coverage of a role again is:

$$cover(r_0, "user") =$$
$$loc(r_0) \cap \left(\bigcup_{(u_0,r_0)\in UR)} cover(u_0, "user") \right)$$

Here the location area of role r_0 has to be intersected with the union of the area of all users that are assigned to role r_0.

If the permission area is smaller than the coverage, i.e.

$$area(cover(p_0, "user")) < area(loc(p_0))$$

we have locations where the permission could be performed, but no user is available who could. This is a strong hint for an erroneous configuration.

To exemplify this we give an example with two roles and three users. For the user-role-assignment we have:

$$UR = \{(u_1, r_1), (u_2, r_1), (u_2, r_2), (u_3, r_2)\}$$

Permission p_0 is assigned to both roles. The respective location restrictions are depicted in figure 3. The coverage is smaller than the location restriction assigned to p_0 because there is an uncovered area (the shaded part of the uppermost permission-area). This part is covered by role r_2. User u_2 and u_3 are assigned to that role, but their location restrictions don't cover the shaded area.

4.2 Empty Assignments

We talk about an "empty user-role assignment" when for $(u_0, r_0) \in UR$ the following holds:

$$loc(u_0) \cap loc(r_0) = \emptyset$$

The intersection between the locations assigned to the user and the role is empty, so the respective assignment is redundant, i.e. it could be removed without changing the policy.

Further the following case shouldn't occur:

$$loc(u_0) \cap cover(r_0, "perm") = \emptyset$$

This means that role r_0 was granted to user u_0, but this assignment doesn't allow him to perform any permission. In the same way we can check for "empty role-permission assignment" for $(r_0, p_0) \in RP$:

$$loc(r_0) \cap loc(p_0) = \emptyset$$

Each assignment made by the administrator of a model should be checked for these two cases; if detected the administrator should be asked if he is sure about this assignment.

5 IMPLEMENTATION

We developed an application with a Java/Swing-based graphical user interface to have a runtime environment for instances of the proposed ACM. The application can perform the spatial analyses covered in the last section and is also capable of visualizing location constraints assigned to individual entities.

The instances of the ACM are stored in a PostgreSQL/PostGIS database management system which provides support for working with spatial data. For the visualization of spatial data we resorted to OpenJump.

OpenJump can load spatial data from a database into different layers. We use this feature to load the location areas for different entities of a model instance into various layers to see how they are related. Figure 4 is a screenshot from the module of the application that implements the coverage calculation as described in section 4.1. On the right side we can see the OpenJump-GUI that displays the covered and the uncovered area of the example given in figure 3.

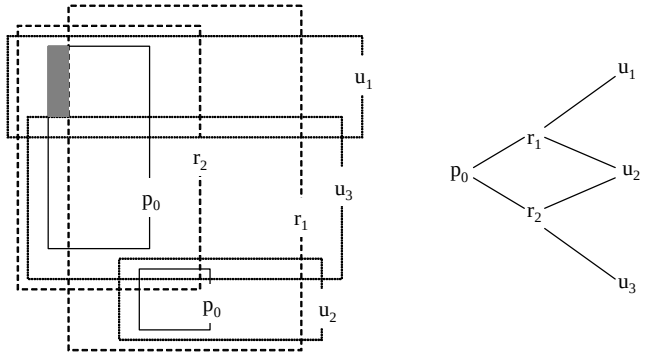

Figure 3: Example for permission-coverage.

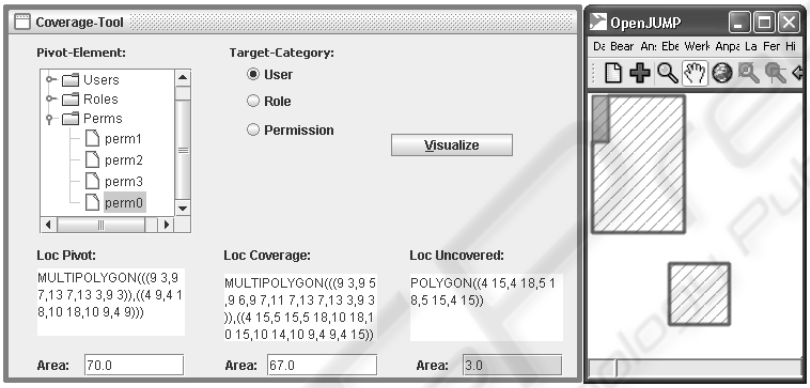

Figure 4: Screenshot of module for calculation of coverage along with visualization provided by OpenJump.

6 CONCLUSIONS

We started by motivating the need for location-aware access control and proposed a RBAC-based model where location-restrictions can be assigned at several components to provide a great degree of flexibility. Based on a formalization of the model itself and the underlying location model we introduced some approaches to analyze model instances based on spatial calculations; this helps to detect erroneous location restrictions. We also sketched the prototypical implementation of a system as runtime environment for the model and visualization purposes.

REFERENCES

Bertino, E., Catania, B., Damiani, M. L., and Perlasca, P. (2005). GEO-RBAC: A Spatially Aware RBAC. In *Proceedings of SACMAT '05*, pages 29–37, Stockholm, Sweden.

Chandran, S. M. and Joshi, J. (2005). LoT-RBAC: A Location and Time-Based RBAC Model. In *Proceedings of the 6th International Conference on Web Information Systems Engineering (WISE '05)*, pages 361–375.

Ferraiolo, D. F., Kuhn, D. R., and Chandramouli, R. (2003). *Role-Based Access Control*. Artech House, Boston and London.

Hansen, F. and Oleshchuk, V. (2003). SRBAC: A Spatial Role-Based Access Control Model for Mobile Systems. In *Proceedings of Nordsec '03*, pages 129–141, Gjovik, Norway.

Hightower, J. and Borriello, G. (2001). Location Systems for Ubiquitous Computing. *IEEE Computer*, 34(8):57–66.

Küpper, A. (2005). *Location-based Services – Fundamentals and Operation*. John Wiley & Sons, Chichester, U.K.

Samarati, P. and di Vimercati, S. D. C. (2001). Access Control: Policies, Models, and Mechanisms. In *FOSAD '00: Revised Versions of Lectures Given during the IFIP WG 1.7 International School on Foundations of Security Analysis and Design on Foundations of Security Analysis and Design*, pages 137–196, London, UK. Springer.