9

Access Control Systems for Geospatial Data and Applications

Maria Luisa Damiani[1,3] and Elisa Bertino[2]

[1] University of Milan, Milan (Italy)
[2] Purdue University, West Lafayette (USA)
[3] EPFL, Lausanne (Switzerland)

9.1 Introduction

Data security is today an important requirement in various applications because of the stringent need to ensure confidentiality, integrity, and availability of information. Comprehensive solutions to data security are quite complicated and require the integration of different tools and techniques as well as specific organizational processes. In such a context, a fundamental role is played by the access control system (ACS) that establishes which *subjects* are authorized to perform which *operations* on which *objects*. Subjects are individuals or programs or other entities requiring access to the protected resources. When dealing with protection of information, the resources of interest are typically objects that record information, such as files in an operating system, tuples in a relational database, or a complex object in an object database. Because of its relevance in the context of solutions for information security, access control has been extensively investigated for database management systems (DBMSs) [6], digital libraries [3, 14], and multimedia applications [24]. Yet, the importance of the spatial dimension in access control has been highlighted only recently. We say that access control has a spatial dimension when the authorization to access a resource depends on position information. We broadly categorize *spatially aware* access control as *object-driven*, *subject-driven*, and *hybrid* based on whether the position information concerns objects, subjects, or both, respectively. In the former case, the spatial dimension is introduced because of the spatial nature of resources. For example, if the resources are georeferenced Earth images, then we can envisage an individual be allowed to only display images covering a certain region. The spatial dimension may also be required because of the spatial nature of subjects. This is the case of mobile individuals allowed to access a resource when located in a given area. For example, an individual may be authorized to view secret information only within a military base. Finally, position information may concern both objects and subjects like in the case of an individual authorized to display images of a region only within a military office.

There is a wide range of applications which motivate spatially aware access control. The two challenging and contrasting applications we propose as examples

are the spatial data infrastructures (SDI) and location-based services (LBS). An SDI consists of the technological and organizational infrastructure which enables the sharing and coordinated maintenance of spatial data among multiple heterogeneous organizations, primarily public administrations, and government agencies. On the other side, LBS enable mobile users equipped with location-aware terminals to access information based on the position of terminals. These applications have different requirements on access control. In an SDI, typically, there is the need to account for various complex structured spatial data that may have multiple representations across different organizations. In an SDI, the access control is thus object-driven. Conversely, in LBS, there is the need to account for a dynamic and mobile user population which may request diversified services based on position. Access control is thus subject-driven or hybrid. However, despite the variety of requirements and the importance of spatial data protection in these and other applications, very few efforts have been devoted to the investigation of spatially aware access control models and systems.

In this chapter, we pursue two main goals: the first is to present an overview of this emerging research area and in particular of requirements and research directions; the second is to analyze in more detail some research issues, focusing in particular on access control in LBS. We can expect LBS to be widely deployed in the near future when advanced wireless networks, such as mobile geosensor networks, and new positioning technologies, such as the Galileo satellite system will come into operation. In this perspective, access control will become increasingly important, especially for enabling selective access to services such as Enterprise LBS, which provide information services to mobile organizations, such as health care and fleet management enterprises. An access control model targeting mobile organizations is GEO-RBAC [4]. Such a model is based on the RBAC (role-based access control) standard and is compliant with Open Geospatial Consortium (OGC) standards with respect to the representation of the spatial dimension of the model.

The main contributions of the chapter can be summarized as follows:

- We provide an overview of the ongoing research in the field of spatially aware access control.
- We show how the spatial dimension is interconnected with the security aspects in a specific access control model, that is, GEO-RBAC.
- We outline relevant architectural issues related to the implementation of an ACS based on the GEO-RBAC model. In particular, we present possible strategies for security enforcement and the architecture of a decentralized ACS for large-scale LBS applications.

The chapter is organized as follows. The next section provides some background knowledge on data security and in particular access control models. The subsequent section presents requirements for geospatial data security and then the state of the art. Afterward the GEO-RBAC model is introduced. In particular, we present the main concepts of the model defined in the basic layer of the model, the Core GEO-RBAC. Hence, architectural approaches supporting GEO-RBAC are presented. Open issues are finally reported in the concluding section along with directions for future work.

9.2 Background Knowledge on Data Security

9.2.1 Data Security

Broadly speaking, data security aims at protecting information against security breaches. Security breaches can be categorized as *unauthorized data observation*, *incorrect data modification*, *and data unavailability*.

Data unavailability occurs when information which is crucial for the functioning of the organization is not ready when needed. Information security means assuring [8]:

- confidentiality, thus protecting data against unauthorized disclosures;
- integrity, thus preventing unauthorized data modification;
- availability, thus recovering from hardware and software errors and malicious data denials.

These requirements raise practically in all application contexts. For example, Chap. 10 in this book focuses on confidentiality and integrity requirements in the context of outsourced-based architectures. In location-based applications, which is our concern, confidentiality is requested to protect various types of information which include, besides the information resources requested by the user, the location of mobile users. Integrity is needed to protect, among others, the information which is transmitted to and from the mobile devices against unauthorized modifications. Data availability is also needed to ensure the availability of the position.

Data security is supported by various tools and systems, including authentication, access control, and audit. Among these, access control is fundamental to ensure confidentiality and integrity of information resources made available to multiple users. The component in charge of access control is the ACS. When a user tries to access an information resource, a component of such system, the *access control mechanism* checks the rights of the requester against a set of authorizations, usually specified by a *security administrator*. An *authorization* states which user, or more generally, which subject can perform which operation on which objects. An authorization is defined as the triple: $< s, op, obj >$, where s, op, and obj refer to subject, operation, and object, respectively. The *subject* identifies the holder of the authorization. It can be a single or group of individuals or programs. The *object* identifies the resource. The nature and granularity of the resource depend on the application. The *operation* defines what the subject is authorized to do with the specified object. The set of authorizations defined by the organization constitutes the *access control policy*. Such a policy is defined using a *policy language* based on an *access control model*. In what follows, we present a classification of access control models.

9.2.2 Access Control Models

Access control models can be categorized into: *mandatory*, *discretionary*, and *role-based* access control (RBAC) models. Mandatory access control (MAC) developed in a military and national security setting; discretionary access control (DAC) has

its roots in academic and commercial research laboratories; the RBAC model, since the seminal paper of [23], has gained increasingly consensus in the research community as well as in industry to finally become a standard widely adopted by organizations [13].

Mandatory Access Control

Mandatory access control models control accesses on the basis of a predefined classification of *subjects* and *objects* in the system. Objects are the passive entities storing information, such as relations in a DBMS. Subjects are active entities performing data accesses. The classification is based on a number of *access classes*, also called *labels* which are associated with every subject and object in the system. A subject is granted authorization to access a given object if and only if some relationship, depending on the access mode, is satisfied between the classifications of the subject and the object. An access class generally consists of two components: a security level and a set of categories. The security level is an element of a hierarchically ordered set. A very well-known example of such a set is the one including the levels TopSecret (TS), Secret (S), Confidential (C), and Unclassified (U), where $TS > S > C > U$. The set of categories is an unordered set (e.g. NATO, Nuclear, Army). Access classes are partially ordered as follows. An access class c_i dominates $\geq$ an access class c_j iff the security level of c_i is greater than or equal to that of c_j and the categories of c_i include those of c_j.The security level of the access class reflects the sensitivity of the information contained in the object. Categories are used to provide finer-grained security classifications of subjects and objects. Access control is based on two principles, formulated by Bell and LaPadula, which are followed by all models enforcing a mandatory security policy. The first states that a subject can read only those objects whose access class is dominated by the access class of the subject; the second states that a subject can write only those objects whose access class dominates the access class of the subject. The application of MAC policies to relational DBMSs has been extensively investigated over the past years. The introduction of such a model entails the solution of several difficult issues. Because of this complexity, the adoption of such a model in DBMSs is not as common as the next model.

Discretionary Access Control

These models are discretionary in the sense that they allow users to grant other users authorization to access the data. Specifically, DAC policies regulate the access of users on the basis of the user's identity and authorizations that specify, for each user (or group of users) and each object in the system, the access modes (e.g. read, write, or execute) the user is allowed on the object. Each request of a user to access an object is checked against the specified authorizations. If there exists an authorization stating that the user can access the object in the specific mode, the access is granted, otherwise it is denied [22]. Because of such flexibility, discretionary policies are adopted in many applications. An important aspect of DAC is related to the *authorization administration* policy. Authorization administration refers to the function of

granting and revoking authorizations. It is the function by which authorizations are entered (removed) into (from) the ACS. Common administration policies include the *centralized administration* policy, by which only some privileged users may grant and revoke authorizations, and the ownership-based administration, by which grant and revoke operations on a data objects are issued by the creator of the object. The ownership-based administration is often extended with features for *administration delegation*. Administration delegation allows the owner of an object to assign other users the right to grant and revoke authorizations, thus enabling decentralized authorization administration.

Role-based Access Control

Role-based access control is centered on the notion of *role*. A role is a semantic construct which represents a job function within an organization. Specifically, the RBAC standard consists of four basic sets of elements: *users*, *roles*, *permissions*, and *sessions*.

- *User* is as a human being or an autonomous agent.
- *Role* represents the function of a user within a community. The community can be a structured organization, for example, a business enterprise or a more informal community, for example the citizens of a city. A role confers a set of permissions on the user.
- *Permission*. Permission is an approval to perform an operation on one or more objects. An object is a resource that shall be protected. An operation is an executable image of a program, which on invocation executes some function for the user over some object. The types of operations and objects depend on the application context in which RBAC is deployed. For example, in a file system, operations might include read, write, and execute; in a DBMS, operations might include insert, delete, append, and update.
- *Session*. When the user logs in, a session is established, during which the user activates some subset of roles that he or she is assigned. The permissions available to the user of the session are thus the permissions assigned to the roles that are currently active across all the user's sessions.

Over the above sets of elements, a number of relations are defined. The *user assignment* relates users to roles through a many-to-many relationship, a user can therefore be assigned multiple roles and the same role assigned to different users. The *permission-assignment* relation relates roles and permissions again through a many-to-many relationship; thus a role can be assigned multiple permissions and similarly each permission can be assigned to multiple roles. The function *SessionUser* maps each session into a user, whereas the *SessionRole* function maps a session onto a set of roles, namely the roles that are active in the session. The basic concepts are formally summarized as follows:

Definition 9.1 (Basic Concepts of RBAC). *Let* U, R, PRMS*, and* SES *denote the set of users, roles, permissions, and sessions, respectively. We define:*

- $UA \subseteq U \times R$. *The user assignment relation that assigns users to roles.*
- $AssignedUser : R \rightarrow 2^U$. *The mapping from a role to a set of users.*
- $PA \subseteq R \times PRMS$. *The permission assignment relation that assigns permissions to roles.*
- $PrmsAssignment : R \rightarrow 2^{PRMS}$. *The mapping of a role into a set of permissions.*
- $SessionUser : SES \rightarrow U$. *The mapping from a session to a user.*
- $SessionRole : SES \rightarrow 2^R$. *The mapping from a session to a set of roles.* □

RBAC standard is defined at three levels, which are referred as Core, Hierarchical, and Constrained RBAC, respectively. *Core RBAC* defines the minimum collection of RBAC elements, element sets, and relations for a RBAC system to be defined. The Hierarchical RBAC component adds relations for supporting role hierarchies. A hierarchy is a partial order defining a seniority relation between roles, whereby senior roles acquire the permissions of their juniors. Constrained RBAC adds *separation of duty* (SoD) constraints to the RBAC [13]. SoD constraints are introduced to prevent conflicts of interest arising when a single individual can simultaneously perform sensitive tasks requiring the use of mutually exclusive duties. In the RBAC standard, enforcement of SoD policies is realized by specifying exclusive role constraints. As an additional level, administrative functions are defined to enable policy specification.

9.3 Motivations and State of the Art

In this section we discuss in detail relevant access control requirements arising in the context of geospatial applications and contexts, and then we survey related research proposals and approaches.

9.3.1 Requirements for Spatially Aware Access Control

Geospatial data have a strategic relevance in various contexts, such as emergency response, homeland security, marketing analysis tools, and environmental risks control procedures. Most applications in these areas require a fine-granularity flexible access control to geospatial data. A possible naive approach to support access control is to build ad hoc data sets (maps) for each type of access the administrator wants to grant: this has been the cartographic approach applied for many years in the past. Such an approach is not suitable when the user community is large and dynamic—which is today often the case in Web-based systems—and when the data and the access control policies dynamically change—which is the case in applications such as emergency response. Another major drawback of the conventional approach is that it does not support flexible protection granularities in the access control policies, and it does not account for various levels of representation that data may have in a geospatial database. The introduction of integrated data management systems for geospatial data, such as current integrated GIS systems, characterized

by comprehensive data models, and current developments in standards for geospatial data such as GML [17] are today making possible the development of advanced ACS that go beyond such naive approaches. However, despite the importance of data protection, almost no efforts have been devoted to the investigation of access control models and systems; only very preliminary proposals exist. In order to position current research, in what follows we overview major requirements that arise in access control for geospatial data.

Richness and Multiplicity of Data Representations

Modern data management systems for geospatial data typically support multiple data representations (such as attributive, vector-based, and topological representations); additional representations are also often available, such as raster images. Not only the same entity may be represented in the data repository according to multiple representations, it can also be represented according to multiple dimensions (such as a point, 0 dimension, or a region, 2 dimensions). Also geospatial objects may be complex objects, consisting of subobjects. In some cases, one may want to hide some of the components of a given spatial object. A suitable access control model for geospatial data must thus: (i) Support the specification of authorizations against geospatial objects at a very fine granularity level. (ii) Account for the various spatial representations. This means that if a user can see an aerial image from which certain objects have been hidden, it is important that the same objects be hidden from the vector-based representation of the same area. (iii) Account for the various object dimensions and resolutions. This means that an administrator should be able to authorize a user to see a given object at 0 dimension and not at higher dimensions, thus hiding detailed information about the object shape. Similarly, in a raster representation, the administrator should be able to specify the resolutions according to which certain objects can be seen. (iv) Support various access rights. In access control models, operations that can be performed on the protected objects correspond to the access rights. This allows one to express authorizations in terms of the operations supported by the model according to which data are represented. It is thus important that in addition to access rights such as read and write, access rights that correspond to meaningful operations on geospatial objects be supported.

Dynamic and Mobile User Population

Many of the geospatial applications are characterized by a user population that constantly changes and moves. Also, in many cases, users from different administrative domains or agencies may need to access the data. A suitable access control model for geospatial data must thus perform the following: (i) Support attribute-based and profile-based user specification. Relying on user login names for grant and revoke authorization is very cumbersome and it is also a low-level approach. Recently, several attribute-based and profile-based access control mechanisms have been proposed, supported by techniques such as attribute certificates and standards such as SAML [20]. Also recent extensions to the RBAC model [7] have been developed

supporting access control across different domains; under such an approach, a user authorized to use a role in a domain is automatically able to enroll in a given role in another domain. These techniques should certainly be incorporated into an access control model for geospatial data. (ii) Supporting authorizations that are dynamically enabled or disabled depending on the user location. It is important to notice that an important requirement is related to the support of mobile users and in particular to the fact that a user may be authorized to access data depending also on the user location or geographical region. For example, a taxi driver authorized to pick up passengers only in a given area of Milano cannot access passenger information when located in another area of the city. It is also important to be able to express user location in terms of the physical position or the logical position (for example "being inside of Milano train station"). Addressing such requirement also entails using secure positioning techniques.

Dynamic Application Contexts

Geospatial data are today increasingly used in various circumstances. For example, consider some data representing roads in a given region; such data can be used for planning the road maintenance schedule or for managing an emergency. It is likely that whether such data may be accessed or not by specific users depend on the current situation, and it is thus likely that different sets of access control policies may need to be activated over time, also depending on the occurrence of specific events. A suitable access control model for geospatial data must thus perform the following: (i) Support mechanisms for policy grouping and modularization. It is important that security administrators be given mechanisms according to which they can group policies into modules that are specific for handling specific situations. To better address compliance requirements, such modules should also include metadata to provide description and information about the policies inside the various modules. (ii) Support event-based activation/deactivation of policy modules. This requirement entails defining an event language and developing suitable event-monitoring techniques. Techniques such as triggers and active rules, developed in the database and the AI field, could be extended to support the automatic activation/deactivation of policies. Notice, however, that a crucial issue is represented by techniques providing high assurance event detection, to avoid an attacker preventing a relevant event from being reported or injecting a false event.

9.3.2 State of the Art

As we have seen, geospatial applications have challenging requirements of access control. Yet, such requirements have been only partially addressed by current research. The spatially aware access control models proposed in literature can be categorized as two broad classes, based on whether they are more focused on geospatial representation of data or user mobility, which are presented in what follows.

Access Control for Geospatial Data

The first access control model for geographical data has been proposed [1] to control the access to georeferenced Earth images. The purpose of such an approach is the controlled dissemination of satellite images at different levels of resolution through a DAC policy. In particular, authorizations state which images users are allowed to access and at which resolution. The system, however, is limited in that it only deals with satellite images. To overcome these limitations, an ACS has been recently developed for the protection of vector-based data available on Web [5]. The underlying model is simple but anticipates some ideas that have been more extensively developed by GEO-RBAC. The central concept in such model is that of *spatial authorization*. A spatial authorization is defined by the tuple $< u, ft, p, w >$, where u denotes the user, ft the object specified in terms of spatial feature types (such as building or house), p the operation in the form of Web service to be performed on spatial objects of the specified type (such as $InsertObject$, to introduce a new spatial element), and w is the *authorization window*. The authorization window indicates the geographical scope of the authorization, that is, the portion of space in which the authorization applies. Accordingly, one can state, for example, that user u is allowed to carry out operation p on all objects of type ft located in the authorization window w. Furthermore, the model supports the specification of administrative functions for the creation and update of spatial authorizations based on a decentralized administration policy. The notion of authorization window has also been integrated in a different access control model, which has been developed for regulating the access to a spatial database [2]. The peculiarity of this approach is that the database is based on a complex spatial data model, enabling multiple levels of spatial representation at multiple granularities. The access control model thus enables the specification of which objects at which granularity and in which portion of space can be selected and modified.

A more recent approach integrating geospatial and security standards to support controlled access to spatial information through geo-Web services has been recently developed [19]. In such an approach, a policy specification language, referred to as GeoXACML, is defined as a geospatial extension of the OASIS eXtensible access control markup language (XACML). GeoXACML supports the specification of rules which enable or deny the access to geospatial objects based on spatial criteria, such as topological relationships. As an example, consider the rule which states that an operation can be performed only on buildings which are located with the administrative boundary of Washington, DC. This approach has some similarities with the notion of authorization window, though in the window-based model [5] not only restrictions over objects can be specified but also on an administration policy.

Access Control in Mobile Applications

None of the previous models is conceived for use in a dynamic environment, which instead is the main concern of spatial and non-spatial context aware access control models.

Non-spatial context-aware access control models include generalized TRBAC (GTRBAC) [18], a RBAC-based model which incorporates a set of language constructs for the specification of various temporal constraints on roles, including constraints on role enabling, role activation, user-to-role assignments, and permission-to-role assignments. X-GTRBAC [7] is another approach which augments GTRBAC with XML for supporting the policy enforcement in a heterogeneous, distributed environment. In addition to temporal constraints, the model also supports non-temporal contextual constraints. The approach, however, is more focused on the software engineering aspects of the access control rather than on the expressivity of the policy specification language. A notable approach is the one proposed through the Generalized RBAC (GRBAC) [10]. GRBAC introduces the concept of environment roles; that is, roles that can be activated based on the value of conditions in the environment where the request has been made. Environmental conditions include time, location, and other contextual information that is relevant to access control. If compared with GEO-RBAC, the concepts of role extent and user position are close to that of context variables. However, the mechanism of contexts is very general and does not account for the specificity of spatial information, such as the multi-granularity of position and the spatial relationships that may exist between the spatial elements in space. Moreover, in GEO-RBAC a common spatial data model is adopted in order to provide a uniform and standard-based representation of locational aspects that, notably, involve not only roles but also protected objects.

The spatial dimension of access control is the basic component of the approach presented in [15]. In such work, an extension of the RBAC model is proposed based on the notion of spatial role, intended as a role that is automatically activated when the user is in a given position. The space model is however very simple and targeted to wireless network applications. It consists of a set of adjacent cells and the position of the user is the cell or the aggregate of cells containing it. The spatial granularity of the position is thus fixed while the space is rigidly structured and the position itself does not have any semantic meaning but simply a geometric value. By contrast, in GEO-RBAC the granularity of the user position may depend on the role of the user; thus no assumption is made on the space layout. Moreover, in GEO-RBAC the spatial dimension integrates geometric and semantic knowledge about the world.

A different approach which combines space and time is presented in [9]; this approach borrows from GEO-RBAC the distinction between real position and logical position and from GTRBAC the notion of temporal context. Such a model, however, does not include the notion of schema, neither supports important features of GEO-RBAC such as hierarchies of enabled roles and spatially aware SoD constraints.

9.4 An Access Control Model for Location-aware Applications: GEO-RBAC

GEO-RBAC is a comprehensive access control model specifically developed to address access control requirements of applications characterized by users that are members of mobile organizations. By mobile organization, we mean a community of

individuals that, because of the role they have in the community, need to access common information resources through LBS. Mobile organizations thus consist not only of enterprises operating on field such as fleet management and resources tracking but also of health care and leisure organizations, and military and civilian coalitions created in response to a crisis, for example natural disaster and war. In these organizations, the mobile members are characterized not only by an identity but also by a functional role. Because of the different activities of the organization's members, it is reasonable to consider that LBS (services for short) are requested based on the roles of individuals and also their position.

The GEO-RBAC model addresses the above issues; it relies on RBAC standard and, like RBAC, is based on the concept of *role*. Moreover, it is structured in levels, referred as Core, Hierarchical, and Constrained GEO-RBAC, respectively. Of these levels, we describe here the Core level, which constitutes the basic component. For the additional levels, we refer the readers to [12].

Before proceeding, we introduce the running example that will be used throughout the chapter. Consider an LBS application for a university campus. The campus consists of various buildings and areas, such as departments, libraries, and recreative areas, each occupying a position in space. From an organizational point of view, people in the campus can play one or more functions, such as teachers, students, visitors, and, say, library subscribers. The members of the campus are connected to a wireless network and equipped with a location-aware PDA by using which they access various LBSs. Assume that John is a student and also a library subscriber. When located within one of the libraries of the campus, John is presented with various services, for example the *book-loan* service to request the loan of a book. Instead, when in a department, he can request the class timetable.

9.4.1 Overview of the Core Level

By adapting the definition in [13] to our model, we say that Core GEO-RBAC defines *a minimum collection of elements which are required to completely achieve a GEO-RBAC system*. The basic notion is that of spatial role, which describes a spatially bounded function for a user. Two other concepts, however, are noteworthy: (a) the *position model* which describes the position of the mobile user; (b) the notions of *role schema and role instance* which provide a representation of the role at two levels, respectively, the intensional and the extensional level. For the representation of the position of the user and the spatial properties of roles, we rely on current geospatial standards [16]. In particular, we describe the spatial objects which are located in the reference space in terms of *simple features* [16] (hereinafter, features). Features have an identity, thematic properties, and can be mapped onto a position in the reference space. A feature is denoted by its identifier. For example, *UniMi* is the identifier of the spatial object which describes the properties of a campus and its position. The position of a feature is represented through a *geometry* which can be of type point, line or polygon, or recursively a collection of geometries. Further, geometries can be related by different types of relationship. Among them, the reference set of topological relations is

{*Disjoint*, *Touch*, *In*, *Contains*, *Equal*, *Cross*, *Overlap*}. These relations are binary, mutually exclusive (if one is true, the others are false), and they are a refinement of the well-known set of topological relations [16]. Moreover, features have an application-dependent semantics that is expressed through the concept of *feature type*. A feature type captures the intensional meaning of the entity, for example, *Campus* and *Department*. The *extension* of a feature type is a set of semantically homogeneous features.

9.4.2 Basic Concepts

The Position Model

The notion of position is fundamental since it characterizes the mobile user. Unlike most proposals in mobile computing which describe the position uniquely in geometric terms, for example a point, we introduce, for the sake of flexibility, the distinction between the *real* and the *logical* position. The real position corresponds to the position of the user on Earth acquired through some positioning technology. Real positions can thus be represented as geometric elements of different types since, depending on the chosen technology and accuracy requirements, they may correspond to points or polygons. Conversely, the *logical position* is defined at a higher level of abstraction to represent positions in a way that is almost independent from the underlying positioning technology. Further, besides a geometry, the logical position has a semantics. For example, logical positions can be a house, an address number, or a road, which are represented in terms of *spatial feature types*. The logical position is computed from real positions by using a *location mapping function*. For example, a location mapping function can be defined to map a position acquired through global positioning system (GPS) onto the closer road segment. More formally, given a feature type ft, a *position mapping function for* ft is a function m_{ft} which, given a real position rp, returns a logical position corresponding to an instance of ft having rp as real position. As we will see, since the localization may respond to different application requirements, for example, with respect to accuracy, the meaning of location may vary depending on the role. For example, the position of a generic campus member may be coarsely defined in terms of campus sectors, assuming that the campus area is subdivided in sectors, whereas the position of the teacher can be represented at a higher resolution by an address.

Spatial Role

The concept of *spatial role* is the distinguishing aspect of our model. To account for the spatial context, a *spatial role* is defined not only by a name as in RBAC but also by a *role extent*. The extent of a role defines the boundaries of the region contained in the reference space. A user, who has been conferred a role r, is thus recognized to effectively play such a role only when logically located within the extent of r. For example, *CampusMember*(*UniMi*) is a spatial role: *CampusMember* is the role

name and *UniMi* is the role extent, specifically the identifier of a spatial feature of type *Campus* denoting, among others, a polygonal space. An individual, which is a member of the campus, is recognized to play the role of *CampusMember* and thus authorized to invoke the services associated to the role, only when located in the polygonal extent of the campus.

Each role is assigned a set of *permissions*. A permission corresponds to a service. For example, the service *BookLoan* of the running example is assigned to the spatial role *LibarySubscriber(MyLibrary)*. The terms *permission* and *service* are used in this chapter in an interchangeable way. Like RBAC, when a user connects to the LBS application, a new *session* is started. Then the user selects the roles, among those which have been assigned to him/her that they wish to play. This set represents the *session roles* and the elements of the set are said to be *activated*. Unlike RBAC, however, in our model, session roles have in addition a status which is *enabled* or *disabled*. For a session role to be *enabled*, the user should be logically located within the space of the corresponding role extent. As the user moves, the status of roles change. Therefore, depending on the position of the user, only a subset of the session roles is enabled and permissions granted. As a result, the set of services which the user can access at a given point in time depends on both the session roles and the position of the user.

Role Schema and Instance

To provide a more compact representation of semantically homogeneous roles defined over different extents, we introduce the distinction between role schema and role instance. A *role instance* is a role defined over a specific extent, in compliance with the role schema. Note that the terms *roles instance* and *spatial role* are used as synonyms. A *role schema* defines common properties of roles with a similar meaning and thus simplifies the specification of roles and ultimately role engineering. Specifically, a role schema defines: (a) a common name for a set of roles; (b) the type of role extent; (c) the type of logical location; (d) the mapping function relating the real position with the logical position. For example, $CampusMember(Campus, Sector, m_{sector})$ is a schema the instances of which are roles having the following properties: the roles have the same name *CampusMember*; *Campus* is the type of the role extent; *Sector* is the type of logical position which is computed by applying function m_{sector} to the real position. Once a schema is specified, the corresponding instances can be simply created by specifying the role name and its extent, for example *CampusMember(UniMi)*. Notice that *UniMi* is a feature of type Campus and defines the boundary of the spatial role.

Because roles are assigned permissions and because of the two different levels of role representation, it seems reasonable to assign permissions to both role schemas and role instances. The permissions which are assigned to a schema are then inherited and shared by all the instances of the schema. For example, if we assign the service *getMap* to the role schema *CampusMember*, it means that such a service can be accessed by all the members of the campus. For the sake of flexibility, however, permissions can be assigned also to single-role instances.

9.4.3 Specification of Core GEO-RBAC

Now, we present the above concepts in more formal terms. Core GEO-RBAC is presented as organized in a number of logical parts, one for each major set of the RBAC model, that is roles, permissions, users, and sessions.

The general structure of the model is illustrated in Fig. 9.1.

We use the graphical representation adopted for RBAC. In defining the model, we refer to the notation introduced in the previous section and summarized in Table 9.1.

Preliminarily, we introduce the notion of partial ordering of features and feature types. Let FT be the set of feature types and $ft_i, ft_j \in FT$, with $i \neq j$, be two elements of the set. We say that ft_i is contained in ft_j, denoted by $ft_i \subseteq_{ft} ft_j$, if for each feature f_i of type ft_i, a feature f_j of type ft_j exists such that the geometry of f_i is contained in the geometry of f_j. For example, the relation of containment between *Town* and *Region* is written as $Town \subseteq_{ft} Region$. Similarly, we say that the feature f_i is contained in feature f_j, denoted by $f_i \subseteq_f f_j$, when the geometry of f_i is contained in the geometry of f_j. As we will see, such relationships will be useful in characterizing the relationships between locations and roles.

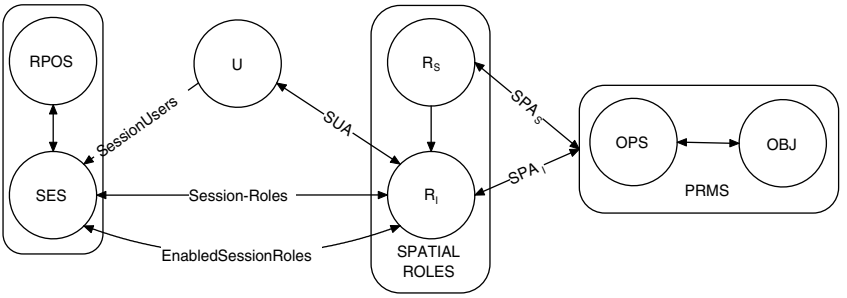

Fig. 9.1. Core GEO-RBAC

Table 9.1. Notation for the main sets used in GEO-RBAC

Notation	Meaning
FT	Feature types
F	Features
R	Role names
SES	Sessions
U	Users
$REXT\_FT$	Role extent types
$LPOS\_FT$	Logical positions types
$LPOS$	Logical positions
$RPOS$	Real positions
M	Position mapping functions
OPS	Operations
OBJ	Objects

Role Schema and Instance

A role schema defines a common name for a set of roles, the role extent type, the logical position type, and the position mapping functions relating the real position with the logical position. A role instance is defined over an extent of the type specified in the corresponding schema, while the logical position of the individual playing such a role is determined by the position mapping function specified in the schema alike. The formal definitions of role schema and instance are reported below [4]:

Definition 9.2 (Role Schema). *A Role Schema is a tuple* $< r, ext, loc, m_{loc} >$ *where:*

- $r \in R$;
- $ext \in REXT\_FT$;
- $loc \in LPOS\_FT$;
- $loc \subseteq_{ft} ext$;
- $m_{loc} \in M$ *is a location mapping function for feature type loc.*

We denote with R_S *the set of role schemas and we assume that, given a role name* $r \in R$, *r is unique in* R_S. *A role schema is also denoted as* $r(ext, loc, m_{loc})$.

Definition 9.3 (Role Instance). *Given a role schema* $r_s \in R_S$, *an instance* r_i *of* r_s *is a pair* $< r, e >$ *where r is the name of the role in schema* r_s, *thus* $r = r_s.r$ *and* $e \in F$ *is a feature of type* $r_s.ext$. *The schema of* r_i *is denoted by* $SchemaOf(r_i)$. *We denote with* R_I *the set of role instances for all role schemas. A role instance is also denoted as r(e).*

Permission

A permission is associated with each service. In our model, permissions can be associated either with the role schema and inherited by all role instances of the schema or directly with the role instances. Such different granularities are formalized by introducing two functions: *S_PrmsAssignment*, relating roles schemas and permissions sets; *I_PrmsAssignment* relating spatial roles, thus role instances, to specific permissions. Function *I_PrmsAssignment** is then introduced to combine permissions directly assigned to spatial roles with permissions inherited from their role schema. Formally [4]:

Definition 9.4 (Permissions). *The set of permissions PRMS is defined as* $PRMS = 2^{(OPS \times OBJ)}$. *We also define:*

- SPA_S: $R_S \times PRMS$, *a many-to-many mapping permission-to-spatial role schema assignment relation;*
- $S\_PrmsAssignment: R_S \rightarrow 2^{PRMS}$, *the mapping of spatial role schema onto a set of permissions. Given a role schema* r_s, $S\_PrmsAssignment(r_s) = \{p \in PRMS | < r_s, p > \in SPA_S\}$;
- SPA_I: $R_I \times PRMS$, *a many-to-many mapping permission-to-spatial role instance assignment relation;*

- *I_PrmsAssignment* : $R_I \rightarrow 2^{PRMS}$ *the mapping of spatial role instance onto a set of permissions. Given a role instance* r_i, $I\_PrmsAssignment(r_i) = \{p \in PRMS | < r_i, p > \in SPA_I\}$;
- $I\_PrmsAssignment^*$: $R_I \rightarrow 2^{PRMS}$ *such that given a role instance* r_i, $I\_PrmsAssignment^*(r_i) = I\_PrmsAssignment(r_i) \cup S\_PrmsAssignment(SchemaOf(r_i))\}$. *Hence, the permissions of a role are those assigned to its schema plus those directly assigned to the instance.*

Users and Session

Spatial roles are assigned to users. The definition of the model for this part is conceptually analogous to that in RBAC. In particular, given a set of users, the following relations are defined: the many-to-many relation SUA relates users and role instances; the function $SR\_AssignedUser$ maps a role instance onto the set of users which can activate that role. Formally [4]:

Definition 9.5 (Users). *We define:*

- $SUA \subseteq U \times R_I$, *a mapping user-to-spatial role instance assignment relation;*
- $SR\_AssignedUser : R_I \rightarrow 2^U$, *the mapping of spatial role instance onto a set of users. Formally,* $SR\_AssignedUser(< r, e > \in R_I) = \{u \in U | (u, < r, e >) \in SUA\}$.

When a user logs in, a new session is activated and a number of roles are selected to be included in the session role set. Given a session s, the following two functions are defined: $SessionUser(s)$ corresponds to the user of the session; $SessionRoles(s)$ corresponds to the role that can be potentially activated in s. Formally [4]:

Definition 9.6 (Sessions). *We define:*

- $SessionUser$: $SES \rightarrow U$, *the mapping from a session s to the user of s;*
- $SessionRoles$: $SES \rightarrow 2^{R_I}$ *with* $SessionRoles(s) \subseteq \{< r, e > \in R_I | (SessionUser(s), < r, e >) \in SUA\}$.

Access Control Mechanism

The session roles are the roles that the user of the session has selected. However, for a session role to be *enabled*, the user should be logically located within the space of the corresponding role extent. Therefore, depending on the user position during that session, only a subset of the session roles is enabled and permissions granted. In order to compute the logical position of a user playing a role r in a session, the location mapping function defined in the schema of r is applied to the user real position, provided by the external environment. Hence, if the logical position of the user is spatially contained in the extent of r, the role is *enabled* and thus the set of permissions assigned to the corresponding role is determined. Given a user's request, the access

control mechanism determines whether the permission requested by the user belongs to the set of permission associated to the set of enabled roles ER. If it is the case the permission is granted otherwise it is rejected. The set of enabled roles ER in session s and real position rp is computed by the function $EnabledSessionRole(s, rp)$.

Definition 9.7 (Authorization Control Function). *An access request is a tuple* $ar = \langle s, rp, p, o\rangle \in SES \times RPOS \times OPS \times OBJ$. *ar can be satisfied at position* $rp \in RPOS$ *if:*

$$(p, o) \in \bigcup_{y \in EnabledSessionRoles(s,rp)} I\_PrmsAssignment^*(y).$$

Basic objects

$FT = \{Campus, Library, Sector, Address)\}$ with: $Sector \subseteq_{ft} Campus$

$F = \{Purdue, MyLib\}$ with: $MyLib \subseteq_f Purdue$

$$PRMS = \{p_1, p_2, p_3, p_4\} \text{ with } \begin{cases} p_1 = GetMap \\ p_2 = ShowClassTimetable \\ p_3 = BookLoan \\ p_4 = BookSearch \end{cases}$$

Schema

$R = \{Teacher, Student, LibrarySubscriber\}$

$$R_S = \{St, Te, Li\} \text{ with } \begin{cases} St =< Student, Campus, Sector, m_{Sector} > \\ Te =< Teacher, Campus, Address, m_{Address} > \\ Li =< LibrarySubscriber, Library, Library, m_{Library} > \end{cases}$$

Instances

$$R_I = \{r_{St}, r_{Te}, r_{Li}\} \text{ with } \begin{cases} r_{St} = Student(Purdue) \\ r_{Te} = Teacher(Purdue) \\ r_{Li} = LibrarySuscriber(MyLib) \end{cases}$$

Permission assignment

$SPA_S = \{(p_1, St), (p_1, Te), (p_2, St), (p_2, Te), (p_3, Li), (p_4, Li)\}$

User assignment

$U = \{Sara, John\}$

$$SUA = \{s_{ua_1}, s_{ua_2}, s_{ua_3}\} \text{ with } \begin{cases} s_{ua_1} = \langle John, Student(Purdue)\rangle \\ s_{ua_2} = \langle Sara, Teacher(Purdue)\rangle \\ s_{ua_3} = \langle John, LibrarySubscriber(MyLib)\rangle \end{cases}$$

Sessions

$SES = \{s_1\}$, $UserSession(s_1) = \{John\}$

$SessionRoles(s_1) = \{Student(Purdue), LibraySubscriber(MyLib)\}$

EnabledRoles

$EnabledSessionRoles(s_1, loc_1) = \{\{Student(Purdue),$
$LibraySubscriber(MyLib)\}\}$ if John is in $MyLib$

Fig. 9.2. An example of a GEO-RBAC policy

Example

Finally, we summarize the GEO-RBAC concepts (see Fig. 9.2) by presenting the access control policy defined for the running example. The policy is built as follows. First, we define the sets of feature types, features, and permissions used in our policy. Then we specify the following role schemas: the *Student* schema specifies that an individual can play the role of student only when located within a campus. *Campus* is a spatial feature type and *Purdue* is an instance of campus. Further, the logical position of a student is represented by a sector of the campus, of feature type *Sector*. The *Teacher* schema is defined in a similar way for the individuals who are teachers in the campus. The logical position of a teacher is represented by an address. The *Library Subscriber* schema specifies that individuals can play such a role only within a library. The logical position of a subscriber is of type *library*. Notice that in this case, the logical position has the same granularity of the role extent; therefore, the actual position within the library is not relevant for the application. The role instances we consider are student at Purdue (i.e. Student(Purdue)), teacher in the same campus (i.e. Teacher(Purdue)), and subscriber of the campus library named MyLib (i.e. LibrarySubscriber(MyLib)). We then specify how permissions, namely information services, are tied to the previous schemas. For example, the service *book-loan* of the running example is assigned to the *LibrarySubscriber* schema and thus is inherited by its role instances. Now suppose that Sara is a teacher at Purdue, while John is a student in the same campus and also a subscriber of MyLib. Assume John to be the only user connected to the system. Until John is outside the Purdue campus, John is not allowed to access any information service. Conversely, when inside the library MyLib, John is recognized to be a library subscriber, besides a student, and thus can request a book loan.

9.5 A Reference Architecture for GEO-RBAC

The GEO-RBAC model defines the conceptual constructs supporting the specification of access control policies. We now complement the discussion by presenting a reference architecture for an ACS based on GEO-RBAC. The purpose is to devise the main building blocks of the access control system and at the same time the key operational aspects. We present in particular two different architectural frameworks, the first relying on a centralized architecture and the other on a distributed architecture for access control in large-scale applications.

9.5.1 A Centralized Architecture

The centralized ACS assumes that all access control functionalities, namely policy administration and enforcement, are implemented by a unique component which filters user requests accepting only those which are authorized by the current security policy. The general architectural framework that we assume is the one shown in Fig. 9.3. Such framework consists of three fundamental components [11]:

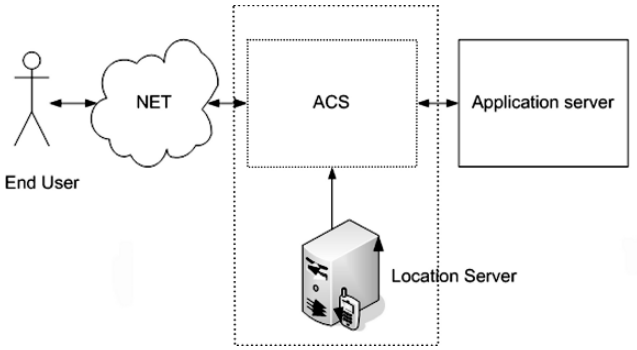

Fig. 9.3. Centralized architecture

- a set of *mobile users* equipped with mobile terminals and connected to a wireless network. Users are assumed to be identified by their terminal ID.
- the *Application Server* providing a set of location-aware information services. Such services are mapped onto *permissions* of GEO-RBAC.
- the ACS. It is a trusted component filtering the users' requests. To obtain the position of the user, the ACS accesses a *Location Server* which aggregates location data from different sources such as the network and mobile terminals equipped with GPS (connected by dotted lines in Fig. 9.3) and responds to queries such as *retrieve the position of terminal ID*. Notice that to prevent uncontrolled disclosure of location data, the ACS is the only component enabled to query the Location Server.

A request for a service is processed as follows: the user requests the service by sending a request message to the ACS. The ACS then determines whether the request can be accepted, and if it is the case the request message is properly restructured and sent to the Application Server. Finally, the requested information is then sent back to the ACS and then, through it, to the user.

In this section, we focus in particular on the key choices underlying the architecture, concerning in particular the definition of an event-driven approach to the specification of the access control mechanism.

The Access Control Mechanism

The access control mechanism applies the access control function to determine the set of enabled roles and thus the services which are accessible to the user at a given location. For the computation of the enabled roles,we need to define at what stage these roles are determined. We devise two possible strategies:

- The status of roles is computed exclusively upon user request. When the user sends a request to the ACS, the system determines which roles are enabled and then based on this information determines whether the request is accepted or rejected. The approach is thus *user-driven*.

- The status of roles is autonomously checked by an agent tracking the position of the user connected to the LBS system. As a state transition occurs for a role, that is, its status changes from enabled to disabled or vice versa, the new status is recorded and the event is notified to interested system components. The strategy is thus *event-driven*.

The simplest approach is the user-driven one because the position of the user is computed on demand and thus the status of roles is determined only when required. This approach has, however, a major drawback, in that the services which are available in a given session at a given instant are not known until the user makes an explicit access request. Therefore, it may occur that the user requires a service which is not accessible in that position or that the user is not aware of available services at a given position.

Conversely, when the event-driven strategy is adopted, the status of each role is determined asynchronously with respect to user requests. Therefore, such information needs to be recorded by the ACS. Although more complex, this approach overcomes the drawbacks of the user-driven strategy: user requests can be more efficiently processed because the current status of roles is available at the time of the request and thus need not they be computed; the users can determine the effective roles can play, before a request is made. Because it is arguably more flexible, the event-driven approach is the one we adopt.

Event-driven Architecture of the ACS

From an architectural point of view, the proposed organization for the ACS is based on the following major components:

- The *Policy DB* is a database storing the security policies specifying, among other information, the spatial roles, the services available to each role, and the roles assigned to each user.
- The *Session DB* records the *status* of sessions. The status at time t of session s is represented by the tuple $< s, t, SR, ER >$, where SR is the set of session roles, thus the roles selected by the user among those which have been assigned to her; ER is the set of roles enabled in s at time t.
- An agent called *Role Tracker*. It periodically retrieves from the Location Server the position of the users of current sessions and determines whether a state transition has occurred for the roles of each session. If this is the case, the event is communicated to the Event Manager.
- In response to a Role Tracker event, the *Event Manager* updates the status of sessions in *Session DB* and notifies the event to the corresponding terminal.

This architecture focuses on the operational meaning of access control enforcement. A number of issues however remain open. Among these, a challenging issue is related to the fact that LBS may be not only of *pull* type, like for example, directory services (e.g. Where is the closest restaurant), but also of *push* type. Under the push model, services are still requested by the user (i.e. subscribed), but the information is provided on a continuous basis. An example is the service which allows one

to be automatically notified about nearby traffic jams. Because in a location-aware context the availability of the service depends on the position of the user, if the user changes position in time, it may occur that the user is no longer in a position which authorizes him/her to access the service. To our knowledge, this issue has not been addressed yet.

9.5.2 The Challenge of a Distributed Architecture for Location-based Applications

The above centralized architecture has two major drawbacks. First, the architecture is not scalable. Suppose that an organization introduces an application in which LBSs are provided by several applications servers. In such a case, a centralized policy enforcement creates a bottleneck since all requests for all application servers should be first sent to the ACS and eventually dispatched to the recipients. Similarly, the administrative operations, such as the addition or removal of a service by a service provider, should be managed by the central security administrator. The second drawback is that it does not address the authentication issue. On the other hand, as already remarked, relying on user login names for grant and revoke authorization is cumbersome and it is also a low level approach.

To overcome the limits of the centralized approach, we propose an access control framework based on a decentralized architecture. Consider the following scenario: suppose that, in our campus, services are provided by various service providers through a number of application servers $AS_1, ..., AS_n$. Moreover, roles are assigned permissions to request services from different providers. Therefore, a many-to-many relationship exists between the set of application servers and the set of roles: an individual, because of his role, can access multiple application servers, and vice versa an application server can be accessed by individuals playing different roles. To enable an efficient access to services and at the same time allow a simple interaction with the application, we propose an architectural framework based on two key design choices: (a) the access control is distributed among a set of autonomous ACS, $ACS_1, .., ACS_n$ where ACS_j with $j \in \{1, ..., n\}$ controls the access to application server AS_j. The access to each application server is then governed by a *local policy* administered by a *local security administrator*. Therefore, there are n local policies and n local security administrators. (b) The enforcement of the local policy is initiated on the client side represented by the mobile terminal and then completed on the server side.

We now describe in more detail this approach focusing in particular on the problem of user-role authentication and policy enforcement.

User–Role Authentication

We introduce the problem of user–role authentication through an example. Suppose that John is assigned therole *student*. Then when John presents such a role to the

ACS, the system is expected to grant access to the services assigned to students. Now the question which arises is how can the system trust John. In a centralized architecture, the answer is simple since the ACS maintains information on users and on all pairs user–role (user–role assignment relation). Therefore when John is first authenticated and then presents his role, the ACS verifies whether the binding $< John, student >$ is specified in the user–role assignment relation. In a distributed context, however, this approach is not flexible enough: when a new user is entered into or removed from the system or a new role is assigned or deassigned to a user, such information must be replicated in every local policy. Such operation results not only are tedious and complex to manage but can also lead to inconsistent polices.

We adopt thus a different approach. The idea is that the user first obtains a digital certificate which specifies the user's role; then this certificate is presented to the ACS to certify the role assignment for the user and to subsequently grant the access to the requested service. The advantage of this approach is that the user-role assignment relation does not need to be specified at the level of local policy.

Digital certificates contain a number of attributes, such as the name, a serial number, expiration dates, and the digital signature of the certificate-issuing authority so that a recipient can verify that the certificate is real. Moreover, certificates hold spatial role information (*role certificates*). As far as we know, the idea of certificates containing spatial information has been not explored yet. The certification authority in charge of issuing role certificates is called *role provider*. User–role authentication is then performed as follows: the user downloads certificates from the role provider, one for each role, and stores them on his mobile terminal. To invoke a service, the user first selects the certificate corresponding to the role he wants to play in the interaction and then transmits it to the application server along with the service requests. In such a way, the user is authenticated along with his role.

Decentralized Policy Enforcement

Suppose now that a user, after being authenticated, requests the permission of invoking a service. Following the GEO-RBAC model, such permission is granted only if the user is located within the extent of a spatial role authorized to request that service. The question we now address is how to assess whether a relationship of spatial containment holds between the user position and the role extent. A first solution is to apply the same approach adopted in the centralized architecture, that is, the ACS, which receives the user's request, obtains the position of the user from the location server and then, based on role information, determines whether the containment constraint is satisfied. The drawback of this approach is that in large applications, it introduces a significant communication burden. We thus opt for a different solution in which policy enforcement is initiated on clients. The idea is as follows: since clients have knowledge of the roles assigned to users because of the role certificates stored on terminals, clients are potentially able to reject requests which cannot be satisfied and thus avoid sending useless requests to the servers. For this approach to

be viable, however, clients must be aware of the position of users and then be able to match such position against role extents. Clients can obtain the position from the Location Server or from the positioning device eventually installed on terminal. In any case, since clients may be not trusted, it is important to ensure the integrity of position data. To that purpose, we assume position data to be provided along with a digital signature by a Secure Location Server.

Access control operations on clients are then carried out by a software component called *local access control* (LAC). Based on role certificates and user's position, the LAC determines the roles, which are enabled, and then, if there is at least an enabled role, it transmits to the application server an encrypted access request that includes the role certificate and the position. Note that the application server trusts the position sent by the client because the position has a digital signature affixed. Therefore, a malicious client cannot forge such information.

Now we consider what happens on the server side when a request is sent by LAC. The request is received by one of the ACS defined in the framework, say ACS_j. Based on the role certificate and position data sent by the client, ACS_j verifies whether the role is enabled. Notice that this operation is performed twice, first at the client and then at the server side. The reason is that the client may not be trusted, and thus, the server needs to make sure that constraints are fulfilled, with the advantage that in the case of trusted clients, unnecessary request processing at the server is avoided. Finally, the system checks whether the requested service is one of the services, which have been assigned to the specified role based on the local security policy. If this is the case, the access is permitted.

The decentralized architecture comprehensive of the LAC layer, the Role Provider, the Secure Location Server, and the set $\{ACS_1, .., ACS_n\}$ of ACS is shown in Fig. 9.4.

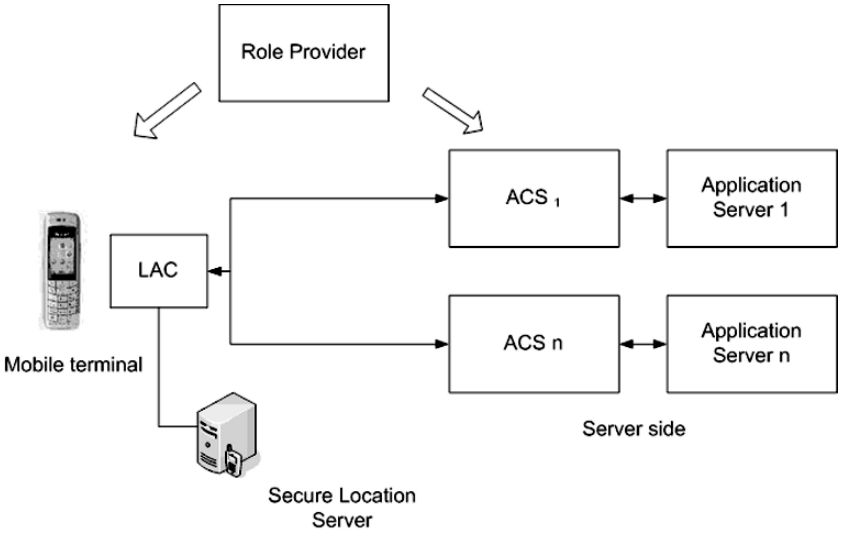

Fig. 9.4. Decentralized architecture

Concluding Remarks

It may have been noticed that in discussing the two architectural approaches, the centralized and the distributed one, we have focused on different aspects. In the centralized solution, the major concern is for the strategy to apply for the enforcement of the security policy which can be user-driven or event-driven. In the decentralized solution, the focus is on distributing the administrative and enforcement functionalities. The question which has not been addressed yet is whether in the decentralized context the enforcement strategy is user-driven or event-driven. This problem deserves a careful analysis, especially if we want to account for services that are not only of pull but also of push type. This issue will be addressed as part of our future work.

9.6 Conclusions

In this chapter, we have discussed issues related to access control in location-based applications. Such applications are characterized by a large variety of requirements affecting both the conceptual definition of access control models and the architectures of ACS. In this chapter, we have shown a rich model, GEO-RBAC, specifically tailored to geospatial applications with mobile users and then illustrated possible architectures supporting such model.

Several issues, however, are still to be addressed concerning the architectural aspects. In particular, the integration of the privacy dimension into access control, following the guidelines proposed in [11], is an important issue, given the existing privacy regulations and the increased privacy concerns by citizens and organizations. The mapping of GEO-RBAC onto an existing architectural framework, X-GTRBAC, is also crucial to obtain an ACS supporting the specification and enforcement of a rich set of context-based access control policies.

Acknowledgments

The work of M.L. Damiani has been partially supported by the European project GEOPKDD "Geographic Privacy-aware Knowledge Discovery and Delivery." The work of E. Bertino has been supported by USA NSF under the project "A Comprehensive Policy-Driven Framework for Online Privacy Protection: Integrating IT, Human, Legal and Economic Perspectives" and by the sponsors of CERIAS.

References

1. Atluri V, Mazzoleni P (2002) A Uniform Indexing Scheme for Geospatial Data and Authorizations. In: Proc. 6th Conf., on Data and Application Security, IFIP TC11/WG11.3, Cambridge, UK, 207–218

2. Belussi A, Bertino E, Catania B, Damiani M L, Nucita (2004) An Authorization Model for Geographical Maps. In: Proc. 12th Int., Symp. of ACM GIS, Washington DC, USA, 82–91
3. Bertino E, Ferrari E, Perego A (2002) MaX : An Access Control System for Digital Libraries and the Web. In: Proc. 26th Computer Software and Application Conference, Oxford, UK, 945–950
4. Bertino E, Catania C, Damiani ML, Perlasca P (2005) GEO-RBAC: A Spatially Aware RBAC. In: Proc. 10th ACM Symposium on Access Control Models and Technologies (SACMAT'05), Stockholm, Sweden, 29–37
5. Bertino E, Damiani ML, Momini D (2003) An Access Control System for a Web Map Management Service. In: Proc. 14th International Workshop on Research Issues in Data Engineering (RIDE-WS-ECEG), Boston, USA, 33–39
6. Bertino E, Sandhu R (2005) Database Security-Concepts, Approaches, and Challenges. IEEE Transactions on Dependable and Secure Computing 2(1):2-19
7. Bhatti R, Ghafoor A, Bertino E, Joshi JBD (2005) X-GTRBAC: an XML-Based Policy Specification Framework and Architecture for Enterprise-wide Access Control. ACM Transactions on Information and System Security 8(2):187–227.
8. Bishop M (2005) Introduction to Computer Security. Addison-Wesley
9. Chandran S M, Joshi JBD (2005) LoT RBAC: A Location and Time-based RBAC Model. In: Proc. 6th International Conference on Web Information Systems Engineering (WISE'05), New York, USA, 361–375.
10. Covington M, Long W, Srinivasan S, Dev AK, Ahamad M, Abowd GD (2001) Securing Context-aware Applications Using Environment Roles. In: Proc. 6th ACM Symposium on Access Control Models and Technologies (SACMAT'01), Chantilly, USA, 10–20
11. Damiani ML, Bertino E (2006) Access Control and Privacy in Location-aware Services for Mobile Organizations. In: Proc. of the 7th International Conference on Mobile Data Management, Nara, Japan
12. Damiani ML, Bertino E, Perlasca P (2005) Data Security in Location-Aware Applications: an Approach Based on RBAC. International Journal of Information and Computer Security (IJICS), in press
13. Ferraiolo D, Sandhu R, Gavrila S, Kuhn R, Chandramouli R (2001) Proposed NIST Standard for Role-Based Access Control. ACM Transactions on Information and System Security 4(2):224–274
14. Ferrari E, Adam NR, Atluri V, Bertino E, Capuozzo U (2002) An Authorization System for Digital Libraries. VLDB Journal 11(1):58–67
15. Hansen F Oleshchuk V (2003) Spatial Role-based Access Control Model for Wireless Networks. In: Proc. IEEE Vehicular Technology Conference VTC2003-Fall, Orlando, FL, USA, 2093–2097
16. ISO/TC211 (2003) 19107: Geographic information - Spatial schema
17. ISO/TC211 (2004) 19136: Geographic information - Geography Markup Language
18. Joshi JBD, Bertino E, Latif U, Ghafoor A (2005) A Generalized Temporal Role-Based Access Control Model. IEEE Transactions on Knowledge and Data Engineering, 17(1):4–23
19. Matheus A (2005) Declaration and Enforcement of Fine-grained Access Restrictions for a Service-based Geospatial Data Infrastructure. In: Proc. 10th ACM Symposium on Access Control Models and Technologies (SACMAT'05), Stockholm, Sweden, 21–28
20. OASIS SAML (2006) http://xml.coverpages.org/saml.html
21. Sandhu R, Ferraiolo D, Kuhn R (2000) The NIST Model for Role-Based Access Control: Towards a Unified Standard. In: Proc. 5th ACM Workshop on Role-Based Access Control, Berlin, Germany, 47–63

22. Sandhu R, Samarati P (1994) Access control: Principles and Practice. IEEE Communications, 32(9):40–48
23. Sandhu R, Coyne EJ, Feinstein HL, Youman CE (1996) Role-Based Access Control Models. IEEE Computer 29(2):38–47
24. Thuraisingham B (1990) Multilevel Security for Multimedia Database Systems. In: Proc. IFIP WG 11.3 Workshop on Database Security (DBSec 1990), Halifax, UK, 99–116

References containing URLs are validated as of October 1st, 2006.