Access Control in Location-Based Services

C.A. Ardagna, M. Cremonini, S. De Capitani di Vimercati, P. Samarati

Dipartimento di Tecnologie dell'Informazione
Università degli Studi di Milano
Via Bramante, 65 - Crema, Italy
{*claudio.ardagna,marco.cremonini,sabrina.decapitani,pierangela.samarati*}*@unimi.it*

Abstract. Recent enhancements in location technologies reliability and precision are fostering the development of a new wave of applications that make use of the location information of users. Such applications introduces new aspects of access control which should be addressed. On the one side, precise location information may play an important role and can be used to develop Location-based Access Control (LBAC) systems that integrate traditional access control mechanisms with conditions based on the physical position of users. On the other side, location information of users can be considered sensitive and access control solutions should be developed to protect it against unauthorized accesses and disclosures. In this chapter, we address these two aspects related to the use and protection of location information, discussing existing solutions, open issues, and some research directions.

1 Introduction

In the last decade, the diffusion and reliability achieved by mobile technologies have revolutionized the way users interact with the external world. Today, most people always carry a mobile device and can stay online and connected from everywhere. Location information is then available as a new class of users' information that can be exploited to develop innovative and valuable services (e.g., customer-oriented applications, social networks, and monitoring services). Several commercial and enterprise-oriented location-based services are already available and have gained popularity [1]. These services can be partitioned in different categories [2]. For instance, there are services that provide information on the position of the users or on the environment surrounding the location of a user (e.g., point of interest, traffic alerts), or services which can help in protecting human lives or highly sensitive information/resources. As an example, the enhanced 911 in North America [3] can exploit location information of users to immediately dispatch emergency services (e.g., emergency medical services, police, or firefighters) where they are needed, reducing the margin of error. In an environment offering location-based

services (LBSs), *users* send a request for using such services to a *LBS provider*. The provider collects the user personal information, possibly interacting with a *location server* (LS), to decide whether the service can be granted and how it can be possibly personalized. The location server works as the positioning system that measures the location information of users carrying mobile devices, and provides such information at different levels of granularity and with different Quality of Service (QoS). The types of location requests that a Location Server can satisfy depend on the specific mobile technology, the methods applied for measuring users position, and the environmental conditions.

Among the different issues that need to be addressed in the development of location-based services, *access control* is becoming increasingly important. Access control represents a key aspect to the success of location-based services, and can be radically changed by the availability of location information, which includes position and mobility of the users. In this chapter, access control issues are analyzed from two different perspectives: *1)* we analyze how current access control systems can integrate and exploit location information in evaluating and enforcing access requests, thus introducing Location-Based Access Control (LBAC) systems; *2)* we analyze how access control mechanisms should change for evaluating and enforcing access to location information, which might be *highly sensitive*.

In the first case, precise and accurate location information is used to enhance and strengthen access control systems by adding functionalities for defining, evaluating, and enforcing location-based policies, i.e., access control restrictions based on the position of the users. LBAC extends access control to the consideration of contextual location information, in particular the location of the user requesting access. Obtaining reliable and accurate location information with software applications reachable via a telecommunication infrastructure (e.g., wireless network) is a challenging aspect due to the intrinsic error of location measurements. An important requirement is then to provide a way to perform *location verification*, meaning that the location of a user has to be securely verified to meet certain criteria (e.g., being inside a specific room or within a geographical area). A stable and reliable verification mechanism can represent an important driver towards the development of a location-based access control system. Once a user's location has been verified using a protocol for location verification, the user can be granted access to a particular resource according to the desired policy. The location verification process must be able to tolerate rapid context changes, since mobile users,

involved in transactions by means of their mobile devices, can wander freely and change their position in the network.

In the second case, location-based information is considered sensitive and therefore needs to be protected against unregulated access and disclosure. The unauthorized release of location information can result in several privacy breaches (e.g., [4]), and make the users target of fraudulent attacks [5] such as *unsolicited advertising*, when products and services are advertised by exploiting the user position without her consent; *physical attacks or harassment*, when the location of a user is used to carry physical assaults; and *users profiling*, when the location of a user is used to infer other sensitive information. This scenario poses a new set of requirements that need to be accomplished by access control systems for protecting location information. For instance, access control may be enriched with mechanisms that obfuscate the location information before its release to other parties [6, 7]. Also, access control systems should be able to manage time-variant information, since location of users can change over time.

The remainder of this chapter is organized as follows. Section 2 describes the basic concepts of access control languages. Section 3 introduces the concept of location-based access control and describes some solutions implementing LBAC. Section 4 provides an overview of existing approaches to protect and manage access and disclosure of location information. Section 5 presents open problems and future work. Finally, Section 6 gives our conclusions.

2 Access Control Languages

Access control systems are based on *policies* that define authorizations concerning access to data/services. Authorizations establish who can (positive authorizations), or cannot (negative authorizations), execute which actions on which resources [8]. Recent advancements allow the specifications of policies with reference to generic attributes/properties of the parties (e.g., name, citizenship, occupation) and the resources (e.g., owner, creation date) involved [9–11]. A common assumption is that these properties characterizing users and resources are stored in *profiles* that define the name and the value of the properties. Users may also support requests for certified data (i.e., credentials), issued and signed by authorities trusted for making statements on the properties, and uncertified data, signed by the owner itself. For instance, an authorization can state

that "a user of age greater than 18 and with a valid credit card number (requester) can read (action) a specific set of data (resource)". When an access request is submitted to the access control system, it is evaluated against the authorizations applicable to it.

From a modeling point of view, each authorization can be seen as a triple of the form $\langle subject, object, actions \rangle$, whose elements are generic boolean formulas over the subject requesting access, the object to which access is requested, and the actions the subject wants to perform on the object. The *subject* is an expression that allows referring to a set of subjects satisfying certain conditions, where conditions can evaluate the user's profile/properties, or the user's membership in groups, active roles, and so on. The *object* is an expression that allows referring to a set of objects satisfying certain conditions, where conditions evaluate membership of the object in categories, values of properties on metadata, and so on. The conditions specified in the policies can be built over generic predicates that can evaluate the information stored at the site or can evaluate state-based information (e.g., the role adopted inside an application, the number of access to a given object, time/date restrictions). For instance, an authorization stating that "professors with age greater than 35 can read critical documents created before the 2008" can be expressed as:

- subject: `equal`(`job`,*Professor*) $\wedge$ `greater_than`(`age`,*35*)
- object: `equal`(`level`,*critical*) $\wedge$ `less_than`(`creation`,*2008/01/01*)
- actions: *read*

where we assume that `equal`, `greater_than`, `less_than` are predefined predicates used to evaluate information stored in the user and/or object profiles, and whose semantic is self-explanatory. Access control policies can then be implemented by using different languages, like logic-based languages (e.g., [12]), which are expressive and characterized by a formal foundation, or XML-based languages (e.g., [9, 11]), which are more suited to the Internet context.

In the next section, we discuss how access control policies based on boolean formula of conditions can be enriched by adding *location-based conditions*, which are expressed using ad-hoc location predicates. In the discussion, we do not make any assumption about the specific language used for implementing the policies and we refer to the abstract model just described.

3 Location-based Access Control Systems

The diffusion and reliability reached by mobile technologies provide a means to use location information for improving access control systems in a novel way. Although, research on LBAC is a recent topic, the notion of LBAC is in itself not new. Some early mobile networking protocols already relied on linking the physical position of a terminal device with its capability of accessing network resources [13]. Extensive adoption of wireless local networks has triggered new interests in this topic. Some studies focused on location-based information for monitoring users movements on Wireless Lan [14] and 802.11 Networks [15]. Myllymaki and Edlund [16] describe a methodology for aggregating location data from multiple sources to improve location tracking features. Other researchers have investigated a line closer to LBAC by describing the architecture and operation of an access server module for access control in wireless local networks [1, 17, 18]. Controlling access to wireless networks, complying with IEEE 802.11 family protocols, is principally aimed at strengthening the well-known security weaknesses of wireless network protocol rather than at defining a general, protocol-independent model for LBAC. The need for a protocol-independent location technique has been highlighted by a study exploiting heterogeneous positioning sources like GPS, Bluetooth, and WaveLAN for designing location-aware applications [18]. Cho et al. [17] present a location-based protocol (Location-Based network Access Control) for authentication and authorization, in infrastructure-based WLAN systems based on IEEE 802.11. The protocol is used to securely authenticate the location claims released by wireless users, and exchange the keys shared for data encryption. The infrastructure is composed of three parties: the *key server* responsible for authentication, location claim verification, and key distribution, the *access points*, and the *mobile stations*. The solution is based on the fact that a mobile station is in a given location if and only if it receives all the relevant information from the corresponding access points. The protocol uses a Diffie-Helmann algorithm to authenticate location claims, authorize network access, and generate the shared keys for communications between mobile stations and access points. Location-based information and its management have been also the subject of a study by Varshney [1] in the area of mobile commerce applications. This is a related research area that has strong connection with location systems and is a promising source of requirements for LBAC models.

Other papers consider location information as a means for improving security. Sastry et al. [19] exploit location-based access control in sensor networks. Zhang and Parashar [20] propose a location-aware extension to Role-Based Access Control (RBAC) suitable for grid-based distributed applications. Atallah et al. [21] study the problem of key management and derivation in the context of geospatial access control. In this work, a geographical space is modeled as a grid of $m \times n$ cells and policies are used to define whether users can access a given rectangular spatial area composed of one or more cells. Each cell is associated with a key and contains information of interests for the users. When a user gains access to an area, a set of keys is derived. Each key enables the user to access a different cell in the area together with its information. Here, a user location is treated as a single point without explicitly considering the intrinsic uncertainty of location measurements. Atluri et al. [22] consider the problem of providing an efficient security policy enforcement for mobile environments. The authors briefly introduce an authorization model based on moving entities and spatio-temporal attributes, and consider three types of authorizations: *i)* on moving subjects and static objects, *ii)* on static subjects and moving objects, and *iii)* on moving subjects and moving objects. The paper concentrates on the enforcement of such authorizations by providing data structures suitable for the management of moving entities, and spatio-temporal authorizations. The paper presents an index structure called S^{PPF} that maintains past, present, and future locations of moving entities together with authorizations, using a partial persistent storage. An evaluation approach is then described where authorizations are compared with nodes modeling moving entities, by analyzing the spatio-temporal extents of both authorizations and moving entities. This solution allows efficient evaluation of access requests that also include *locate* and *track* privileges.

While all these approaches have made significant steps in the development of models and systems supporting location-based information, the definition of a LBAC model that takes into consideration the special nature of location information is still an emerging research issue that has not been yet fully addressed by the security and access control research community. Only few works provides solutions for defining and evaluating location-based policies. In the following, we first describe a solution providing a LBAC infrastructure [23] (Section 3.1) and then an extension to XACML [11] for the definition of geospatial predicates [24] (Section 3.2).

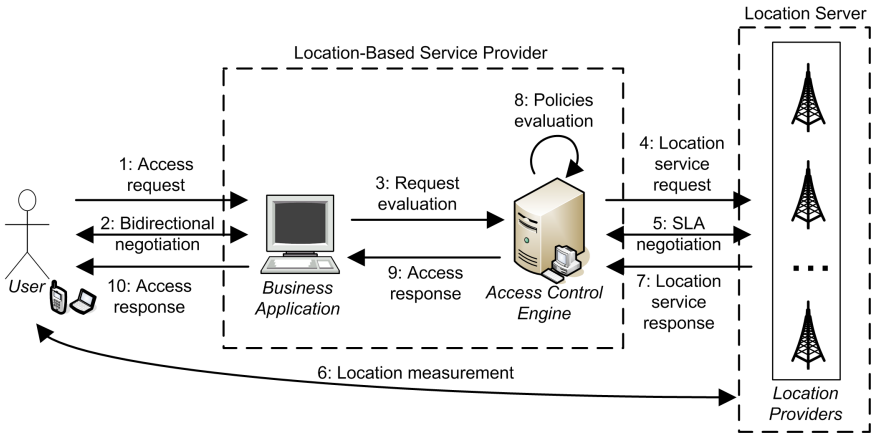

Fig. 1. LBAC architecture

3.1 An Access Control System for LBAC Policies

Ardagna et al. [23] define a LBAC system that supports location-based policies. Intuitively, a location-based policy exploits the physical location of users to define when they can access a service or a resource. The authors identify three main steps towards the development of a LBAC system: *i)* the design of a reference LBAC architecture that can support the evaluation and enforcement of location-based policies; *ii)* the definition of location-based conditions; and *iii)* the definition of a mechanism for the evaluation and enforcement of location-based conditions.

3.1.1 LBAC Architecture

LBAC definition changes the conventional access control architecture, since there are more parties involved. Figure 1 presents a LBAC architecture that involves four logical components.

User. The entity whose access request to a location server must be authorized by a LBAC system. Users carry terminals enabling authentication and some form of location verification.

Business application. Customer-oriented application that offers services whose release is regulated by location-based policies.

Access Control Engine (ACE). The entity that is responsible for evaluating access requests according to some location-based policies. The ACE communicates with one or more Location Providers for acquiring location information. The ACE does not have direct access to

the location information; rather, it sends requests to external services and waits for the corresponding answers.

Location Providers (LPs). The trusted entities that provide the location information (e.g., context data about location and timing, location-based predicate evaluation) by implementing Location Server interfaces.

Interactions among the User, the Business Application, the Access Control Engine, and the Location Providers are carried out via request/response messages (see Figure 1). The process is initiated by a user that submits an access request to a Business Application (step 1). A negotiation process between the two parties is then used to exchange those data that are relevant to the policy evaluation (step 2). The request is further forwarded to the ACE (step 3) that interacts (if needed) with the Location Providers (steps 4-7), evaluates policies (step 8), and returns an access decision (steps 9-10). Communications between the ACE and the Location Providers may be driven by a service level agreement (SLA) negotiation phase (step 5). This negotiation is used to agree upon and set quality of services attributes and the corresponding service cost.

3.1.2 Location-Based Conditions

The location-based conditions that might be useful to include in access control policies and whose evaluation is possible with today's technologies fall within three main categories:

- *position-based* conditions on the location of the users (e.g., to evaluate whether users are in a certain building or city, or in the proximity of other entities);
- *movement-based* conditions on the mobility of the users (e.g., velocity, acceleration, or direction where users are headed);
- *interaction-based* conditions involving relationships among multiple users or entities (e.g., the number of users within a given area).

Table 1 presents some specific predicates corresponding to the conditions of the kind identified by the classes above. In particular, predicates inarea, disjoint, and distance are of type position and evaluate the location of the users; velocity is of type movement and evaluates the mobility of the users; density and local_density are of type interaction and evaluate spatial relationships between entities. Other predicates can be added as the need arises and technology progresses. Conditions are expressed as boolean

Table 1. Examples of location-based predicates

Type	Predicate	Description
Position	inarea(*user, area*)	Evaluate whether *user* is located within *area*.
	disjoint(*user*, *area*)	Evaluate whether *user* is outside *area*.
	distance(*user*, *entity*, *min_dist*, *max_dist*)	Evaluate whether distance between *user* and *entity* is within interval [*min_dist*, *max_dist*].
Movement	velocity(*user*, *min_vel*, *max_vel*)	Evaluate whether *user*'s speed falls within range [*min_vel*, *max_vel*].
Interaction	density(*area*, *min_num*, *max_num*)	Evaluate whether the number of users currently in *area* falls within interval [*min_num*, *max_num*].
	local_density(*user*, *area*, *min_num*, *max_num*)	Evaluate the density within a 'relative' area surrounding *user*.

queries of the form *predicate*(*parameters,value*), stating whether *predicate* over *parameters* has the specified *value*. The evaluation of a boolean query returns a triple [*bool_value*,*confidence*,*timeout*] stating whether the predicate is true or false (*bool_value*), the time validity associated with the assessment (*timeout*), and a *confidence* value expressing the reliability associated with the assessment. This confidence may depend on different aspects such as the accuracy, environmental and weather conditions, granularity of the requested location, and measurement technique.

The language for location-based predicates assumes that each user, who is unknown to the location server responsible for location measurements, is univocally identified via a user identifier (UID). For instance, a typical UID for location-based applications is the SIM number linking the user's identity to a mobile terminal. A unique identifier is also associated with each object, and any physical and/or moving entity that may need to be located (e.g., a vehicle with an on-board GPRS card). Moreover, to simplify the specification of location-based conditions, a set of map regions identified either via a geometric model (i.e., a range in a n-dimensional coordinate space) or a symbolic model (i.e., with reference to entities of the real world such as streets, cities, or buildings) are assumed to be predefined in the system [25]. For instance, let `alice` be a user identifier, and `Manhattan_NY` and `University_Campus_Secretary` be two map regions. Predicate inarea(`alice`,`Manhattan_NY`) requests `alice` to be located in `Manhattan_NY`; predicate velocity(`alice`,0,10) requests `alice` to be (semi-)static (speed included in [0,10]).

Table 2. Examples of access control rules regulating access to the Mobile Network Console and databases of a mobile network

	subject		object	actions
	generic conditions	location-based conditions		
1	equal(**user**.role,admin) ∧ valid(**user**.username, **user**.password)	inarea(**user**.sim, Server_Room) ∧ density(Server_Room, 1, 1) ∧ velocity(**user**.sim, 0, 3)	equal(**object**.name,MNC)	execute
2	equal(**user**.role,admin) ∧ valid(**user**.username, **user**.password)	inarea(**user**.sim, Inf._System_Dept.) ∧ local_density(**user**.sim, Close_By, 1, 1) ∧ velocity(**user**.sim, 0, 3)	equal(**object**.category, Log&Bill)	read
3	equal(**user**.role,CEO) ∧ valid(**user**.username, **user**.password)	local_density(**user**.sim, Close_By, 1, 1) ∧ inarea(**user**.sim, Corp._Main_Office) ∧ velocity(**user**.sim, 0, 3)	equal(**object**.category, customer)	read
4	equal(**user**.role,CEO) ∧ valid(**user**.username, **user**.password)	local_density(**user**.sim, Close_By, 1, 1) ∧ disjoint(**user**.sim, Competitor_Location)	equal(**object**.category, StatData)	read
5	equal(**user**.role,guest) ∧ valid(**user**.username, **user**.password)	local_density(**user**.sim, Close_By, 1, 1) ∧ inarea(**user**.sim, Corporate_Location)	equal(**object**.category, StatData)	read

Besides location-based information, users and objects may be characterized by other properties that, for simplicity, are assumed to be stored in a profile, and to be referenced via the usual dot notation. For instance, `alice.address` indicates the address of user `alice`. Here, `alice` is the identity of the user (and therefore the identifier of the corresponding profile), and `address` is the name of the property. Also, since policies may need to refer to the user and object of the request being evaluated without need of introducing variables in the language, two keywords are used: **user**, which indicates the identifier of the requester, and **object**, which indicates the identifier of the object to which access is requested.

Location-based access control policies can then enrich the expressive power of current languages by allowing the evaluation of location-based conditions in the context of subject/object expressions. This way authorizations can result applicable to some access depending on conditions, such as, the location of the requester or of the resource.

Example 1. Consider a company responsible for the management of a mobile network that needs both strong authentication methods and expressive access control policies. Suppose that the Mobile Network Console (MNC) is the software that permits to reconfigure the mobile network. Table 2 presents some examples of protection requirements for such a service [26]. Managing a nation-wide mobile network is an extremely critical activity because reconfiguration privileges must be granted to strictly selected personnel only, that is, the execution of the MNC must be allowed according to high security standards. To this aim, Rule 1 states that only registered administrators that are static and alone in the server room can execute the MNC. In addition to the MNC execution privileges, also

the access to mobile network's databases must be managed carefully and according to different security standards, depending on the level of risk of the data to be accessed. Access to logging and billing data is critical, because they include information about the position and movements of mobile operator's customers. Rule 2 is then defined and permits registered administrators that do not have other users in their proximity, static, and located in the information system department, to read logging and billing data. Access to customer-related information is usually less critical but still has to be handled in a highly secured environment and has to be granted only to selected personnel. Rule 3 states that registered CEOs that do not have other users in their proximity, static, and located in the corporate main office can read customer data. Finally, while statistical data about the network's operation is at a lower criticality level, access to them must be controlled, e.g., by preventing disclosure to competitors. To this aim, Rules 4 and 5 are defined: Rule 4 states that registered CEOs that do not have other users in their proximity and that are not located in a competitor location can read statistical data; Rule 5 states that registered guests that do not have other users in their proximity and located in the corporate location can read statistical data.

3.1.3 Location-Based Conditions Evaluation and Enforcement

The introduction of location-based conditions changes the usual way in which access control policies are evaluated and enforced. In particular, an ad-hoc solution must be designed to fully address both uncertainty and time-dependency of location-based information. The solution presented in [23] is based on two semantically uniform SLA parameters, *confidence* and *timeout*, returned by a LP to the ACE in response to the evaluation of a boolean query. Before illustrating how the access control process operates, we need to solve a basic problem: location-based predicates appear in rules as parts of a boolean formula (see Table 2), while the responses to boolean location queries are in the form of a triple [*bool_value*,*confidence*,*timeout*]. Then, to process a response from the Location Provider, the Access Control Engine will need to assign a truth value to the response. Intuitively, the transformation of a location predicate value into a boolean one requires the Access Control Engine to determine whether or not the value returned by the Location Provider can be considered valid for the purpose of controlling access. Such an evaluation will depend on parameters timeout and confidence returned by the Location Provider. Responses with a timeout that has already ex-

pired automatically trigger the re-evaluation of the predicate regardless of the other parameter values because considered as unreliable for any decision. Responses with a timeout that has yet not expired are evaluated with respect to the confidence value. The confidence value is compared with a *lower* and *upper* thresholds, specified for each location predicate. According to the result of this comparison (i.e., whether the confidence value is greater than the upper threshold, less than the lower threshold, or between the two), the boolean value contained in the response to a boolean query will be treated differently. More precisely, for each predicate in Table 1, an *Extended Truth Table* (ETT) defines a *lower* and *upper* thresholds, and a *MaxTries* parameter. If the *confidence* level for a given predicate evaluation is greater than the preset upper threshold, then *bool_value* returned by the LP is confirmed. If the *confidence* level is below the lower threshold, the location-based condition is evaluated to $\neg$*bool_value*. Otherwise, if the confidence level is between lower and upper thresholds neither the returned value nor its negation can be considered sufficiently reliable. Predicate re-evaluation is then triggered at the LP. In this case, the predicate is re-evaluated, at most *MaxTries* times, until the returned relevance is not between the thresholds. If after *MaxTries* re-evaluations of the predicate the outcome remains unchanged, the location-based condition evaluates *Undefined*.

Example 2. Suppose that for inarea predicate the lower and upper thresholds are 0.2 and 0.8, respectively, and that
inarea(`Alice`,`Manhattan_NY`) = [*True*,*0.85*,*2009-01-20 9:00pm*]
is the triple returned by the LP to the ACE stating that `Alice` is located in `Manhattan_NY` with *confidence* of 85%. Such an assessment is to be considered valid until *9:00pm* of *January 20th, 2009*. The ACE evaluates inarea(`Alice`,`Manhattan_NY`) to *True*, since *0.85*>*0.80*.

3.2 GeoXACML

The *Geospatial eXtensible Access Control Markup Language* (GeoXACML) [24] has been introduced by the Open Geospatial Consortium (OGC) as an extension to the XACML Policy Language [11], to support the declaration and enforcement of predicates based on geographic information. GeoXACML, which becomes an OGC standard in February 2008, defines ad-hoc extensions to XACML for including geometric attributes and spatial functions (predicates). The attributes introduced are derived from the Geographic Markup Language (GML) and defined

in the GeoXACML Core Geometry Model. Examples of geometric attributes are: *Point*, that models a single location; *LineString*, that represents a curve with linear interpolation between Points; *Polygon*, that identifies a planar area defined by an exterior boundary, and zero or more interior boundaries; *MultiPoint*, *MultiLineString*, and *MultiPolygon*, that represent a collection of Points, LineStrings, and Polygons, respectively. The GeoXACML predicates can be partitioned into different categories: *topological*, *geometric*, *bag*, *conversion*, and *set*. Table 3 presents some predicates, which can be used for testing topological relations between geometries (we refer to the OGC proposal [24] for the complete set of predicates). A geometry provides a description of geographic characteristics (e.g., locations, shapes). The encoding of geometry depends on the coordinate reference system (CRS) or spatial reference system (SRS) that is used. It is important to note that some predicates provides supporting functionalities only. For instance, the predicates in the conversion category assist in the conversion of other measurement units in terms of meters or square meters (the only accepted by GeoXACML). The use of these conversion functions should however be minimized to avoid unnecessary delays in information processing. Another set of predicates providing supporting functionality, included in the geometric category, is used to verify special characteristics of geometries. For instance, to verify whether a geometry has anomalous geometric points (e.g., self intersection, or self tangency).

GeoXACML, being an extension of XACML, has the same policy syntax of XACML. A GeoXACML policy is then composed of a set of `Rule` elements, each one leading to a binary effect (i.e., *Permit* or *Deny*). An authorization decision is derived by first determining all the rules applicable to a given request. All matching rules are then combined according to a predefined algorithm to obtain the resulting effect of the policy. When more policies are applicable, all resulting policy effects produced for a given request must be combined to produce the final authorization decision. The main difference between XACML and GeoXACML is that the latter supports the declaration of spatial restrictions, which are expressed through the predicates above-mentioned. Figure 2 shows an example of GeoXACML rule whose `Effect` is Permit. For simplicity, namespaces in the rule element are omitted. The rule's target (i.e., element `Target`) has three main elements: `Subjects`, which defines the rule's subjects, that is, *John Brown*; `Resources`, which identifies the rule's objects, that is, *Building*; and `Actions`, which specifies the actions that can be performed, that is, *Read*. Element `Condition` introduces further matching conditions; in

Table 3. Examples of GeoXACML spatial functions

Type	Function	Description
Topological	Contains(*g1:Geometry*, *g2:Geometry*) : Boolean	Returns a true value if and only if geometry g2 lies in the closure (boundary union interior) of geometry g1.
	Crosses(*g1:Geometry*, *g2:Geometry*) : Boolean	Returns a true value if and only if geometries g1 and g2 have some but not all interior points in common, and the dimension of the intersection is less than that of both of the geometries.
	Disjoint(*g1:Geometry*, *g2:Geometry*) : Boolean	Returns a true value if and only if the geometries g1 and g2 have no points in common.
	Equals(*g1:Geometry*, *g2:Geometry*) : Boolean	Returns a true value if and only if geometries g1 and g2 are equal (geometrically contain exactly the same points).
	Overlaps(*g1:Geometry*, *g2:Geometry*) : Boolean	Returns a true value if and only if geometries g1 and g2 have some but not all points in common, and the intersection has the same dimension as each geometry.
	Within(*g1:Geometry*, *g2:Geometry*) : Boolean	Returns a true value if and only if geometry g1 is spatially within geometry g2, that is, if every point on g1 is also on g2.
Geometric	Boundary(*g:Geometry*) : Bag	Returns a bag of geometry values representing the combinatorial boundary of geometry g.
	Centroid(*g:Geometry*) : Geometry	Returns the point that is the geometric center of gravity of the geometry g.
	Intersection(*g1:Geometry*, *g2:Geometry*) : Bag	Returns a bag of geometry values representing the Point set intersection of geometry g1 and geometry g2.
	Union(*g1:Geometry*, *g2:Geometry*) : Bag	Returns a bag of geometry values representing the Point set union of geometry g1 with geometry g2.
	Area(*g:Geometry*) : Double	Returns a value representing the area of geometry g.
	Distance(*g1:Geometry*, *g2:Geometry*) : Double	Returns a value representing the shortest distance in meter between any two points in the two geometries g1 and g2.

our example, the fact that address has to be *Wincott Street*. The semantic of the rule is that "the user *John Brown* can *Read* the information object of class *Building*, if the address is *Wincott Street*" [27].

```
<Rule ... Effect=‘‘Permit’’ RuleId=‘‘Example’’>
  <Target>
    <Subjects>
      <Subject>
        <SubjectMatch MatchId=‘‘urn:oasis:names:tc:xacml:1.0:function:string-equal’’>
          <AttributeValue DataType=‘‘http://www.w3.org/2001/XMLSchema#string’’>
            John Brown
          </AttributeValue>
          <SubjectAttributeDesignator
           DataType=‘‘http://www.w3.org/2001/XMLSchema#string’’
           AttributeId=‘‘urn:oasis:names:tc:xacml:1.0:subject:subject-id’’/>
        </SubjectMatch>
      </Subject>
    </Subjects>
    <Resources>
      <Resource>
        <ResourceMatch MatchId=‘‘urn:oasis:names:tc:xacml:1.0:function:string-equal’’>
          <AttributeValue DataType=‘‘http://www.w3.org/2001/XMLSchema#string’’>
            Building
          </AttributeValue>
          <AttributeSelector
           RequestContextPath=‘‘name(//ca:CityModel/gml:featureMember/ca:Building[1]’’
           DataType=‘‘http://www.w3.org/2001/XMLSchema#string’’/>
        </ResourceMatch>
      </Resource>
    </Resources>
    <Actions>
      <Action>
        <ActionMatch MatchId=‘‘urn:oasis:names:tc:xacml:1.0:function:string-equal’’>
          <AttributeValue DataType=‘‘http://www.w3.org/2001/XMLSchema#string’’>
            Read
          </AttributeValue>
          <ActionAttributeDesignator
           AttributeId=‘‘urn:oasis:names:tc:xacml:1.0:action:action-id’’
           DataType=‘‘http://www.w3.org/2001/XMLSchema#string’’/>
        </ActionMatch>
      </Action>
    </Actions>
  </Target>
  <Condition>
    <Apply FunctionId=‘‘urn:oasis:names:tc:xacml:1.0:function:all-of’’>
      <Function
       FunctionId=‘‘urn:oasis:names:tc:xacml:1.0:function:string-equal’’/>
        <AttributeValue
         DataType=‘‘http://www.w3.org/2001/XMLSchema#string’’>Wincott Street</AttributeValue>
        <AttributeSelector ‘‘
         RequestContextPath=‘‘//ca:CityModel/gml:featureMember/ca:Building/ca:address’’
         DataType=‘‘http://www.w3.org/2001/XMLSchema#string’’>
    </Apply>
  </Condition>
</Rule>
```

Fig. 2. An example of GeoXACML rule

4 Protecting Location Information in Ubiquitous Computing

Today ubiquitous technologies give the basis for accessing, mining, and managing large amount of location information. Such information, however, can be extremely sensitive, and lack of its protection can result in several attacks to the user's personal sphere. Research has been approaching the problem of protecting access to location information from different perspectives, such as the development of enhanced access control architectures or the definition of new languages for protecting location information. In the following, we illustrate some of these proposals.

4.1 Geopriv

IETF *Geopriv* working group [28] proposes a solution for protecting privacy of location information, when it is transmitted and shared over the Internet. Geopriv's main principles and considered threats have been formalized in the IETF RFC 3693 and RFC 3694 [29, 30]. Geopriv considers a scenario in which a requester asks for location information of a target to a location server. An architecture to manage such a scenario has been introduced and includes four main parties.

- *Location Generator* (LG) gathers location information of users and makes it available to the Location Server.
- *Location Server* (LS) provides location services to Location Recipients, and stores the location information of the users.
- *Location Recipient* (LR) subscribes for a location-based service provided by the LS, and requests access to the location information stored by the LS.
- *Rule Holder/Maker* (RH/M) defines the privacy policies which regulate the disclosure of location information to the LR. The policies are enforced by the LS.

Based on these logical components, different architectural layouts are possible. For instance, LG and LS may coexist on the same mobile device (e.g., a GPS receiver) or could be distributed components communicating remotely. The RH/M could be a centralized component managing privacy rules and communicating them to the corresponding LS, or it could be co-located with the LS.

The location information of users is part of a container, called *Location Object* [31]. In addition to the location information, a location

object includes privacy preferences (i.e., usage-rules), that remain attached to the location information for its entire life-cycle. In particular, usage-rules allow the definition of conditions that can: *i)* limit retransmission (e.g., "retransmission-allowed"), *ii)* limit retention (e.g., "retention-expires" date), and *iii)* contain a reference to external rulesets.

Geopriv IETF RFC 4745 [32] defines the framework for creating privacy policies that regulate the release of location information. A Geopriv privacy policy, encoded in XML, is composed of a `ruleset` element that contains an unordered list of `rules` elements corresponding to positive authorizations. Each rule has an element `conditions`, `actions`, and `transformations`. The `condition` element is a set of expressions, each of which evaluates to either true or false. A limited set of conditions can be specified in the `conditions` element: `identity`, `sphere`, and `validity`. The `identity` element restricts the rule matching either to a single identity, using the `one` element, or a group of identities, using the `many` element. In particular, the `one` element identifies exactly one authenticated entity or user, while the `many` element represents a generic number of users in a domain (i.e., it matches the domain part of an authenticated identity). Moreover, the `identity` element can exclude individual users or users belonging to a specific domain through the `except` element. The `sphere` element can be used to match the state (e.g., work, home) a target holds at the time of the access request evaluation. Finally, the `validity` element is used to restrict the time validity of each rule. Additional condition elements can be added by proposing extensions to the privacy policy specification in RFC 4745. The `actions` element specifies actions to be applied before the release of location information. The `transformations` element specifies how the location information should be modified when a permission is granted; for instance, it can state that the original location should be made less precise. While conditions can be considered as the 'if'-part of the rules, which states whether the rule is applicable, actions and transformations form the 'then'-part, which determines the operations to be performed before disclosing information.

Figure 3 shows an example of Geopriv rule. The rule states that, during February 2009, the authenticated entity *sip:bob@example.com* or *mailto:dave@example.net* can access the location information, protected by the rule, if target's sphere is equal to "work".

4.2 Protecting Location Information in Mobile Applications

Different works have addressed the problem of protecting location information in mobile applications.

```
<rule id=‘‘a7k55r7’’>
  <conditions>
    <identity>
      <one id=‘‘sip:bob@example.com’’/>
      <one id=‘‘mailto:dave@example.net’’/>
    </identity>
    <sphere value=‘‘work’’/>
    <validity>
      <from>2009-02-01T00:00:00.000-03:00</from>
      <until>2009-02-28T23:59:59.000-03:00</until>
    </validity>
  </conditions>
  <actions/>
  <transformations/>
</rule>
```

Fig. 3. An example of Geopriv rule

A first line of research focuses on extending Platform for Privacy Preferences (P3P) for protecting the secondary uses of location information [33–35]. P3P [36, 37] has been originally designed by the World Wide Web Consortium (W3C) to address the need of users to assess whether the privacy practices defined by a Web site comply with their privacy requirements, before the release of personal information. Privacy requirements are expressed through *A P3P Preference Exchange Language* (APPEL) [38]. Starting from the work done in P3P, Langheinrich [34] proposes a privacy awareness system (pawS) for ubiquitous and pervasive environments, where services collect users data. The main goal of pawS is to provide an infrastructure that allows users to protect their privacy and to keep track of all data released and of their subsequent management at the service side. pawS uses P3P to encode data usage policies of the service and users define their preferences through APPEL. In pawS, a mobile user carries a mobile device with a *privacy assistant*. When a user enters a geographical area in which a number of services are available (e.g., location tracking service using video-camera), the privacy assistant is prompted with the data collection practices of the service. This communication happens on wireless channels. To save the battery of the portable devices and make the system appealing also for mobile users, data usage practices are delegated by the user to a *personal privacy proxy* residing on the Internet, which is responsible for managing all negotiations with the service. In particular, the personal privacy proxy asks the *service proxy* for service policies and then matches them with the users' preferences. If the matching is successful, the service is used and data released, otherwise the service is disabled. Access control poli-

cies exploiting the location of the requesters are evaluated and enforced. Myles et al. [35] discuss a location-based scenario where applications require location information of the users for service release. The main goal is, on the one side, to balance the need of mechanisms to protect users' privacy limiting service intrusiveness, and, on the other side, to minimize the overhead given to the users. The proposed system architecture is composed of three main entities: *i)* a location server, that manages positioning systems (e.g., GPS, cellular technologies) and answers to requests for location information; *ii)* several validators, that are responsible for evaluating the requests and determining whether the location information can be released, based on preferences of the users; *iii)* client applications, that submit requests for location information. The authors assume trust relationships between users, validators, and location servers. Users are registered with at least one location server and store their requirements within it. These requirements are implemented by the validators. When a client application needs to access the location of a user, it first selects the relevant location server, and then submits the request. Such a request also includes the privacy policies that specify how the client application will manage the data after their collection. The privacy policies can be expressed through an extension of P3P that allows the modeling of requests initiated by the application. After receiving the request, the privacy policies are matched with the privacy preferences stored by the validators. Such preferences can contain restrictions based on the time of the request and on geographical areas. Validators can implement a variety of mechanisms for privacy preferences specifications (e.g., APPEL). Hong et al. [33] provide an extension to P3P for representing user privacy preferences in context-aware applications. The authors add features to the P3P language to express the identifiers of the users whose locations are collected, the time period in which the data can be accessed, and the location from which the data can be managed. They propose a centralized architecture that includes a middleware responsible for matching preferences and policies. The middleware is enriched with a plug-in service to support context-aware applications, called *privacy database mediator*. The privacy database mediator provides functionality to automatically generate privacy policies and user preferences according to the context.

Another line of research has addressed the definition of authorization architectures, based on certificates and encryption, to protect location information. Hauser and Kabatnik [39] address the problem of protecting the location information of the users by providing a privacy-aware architecture that allows users to define rules regulating the access to

their location information. The proposed solution relies on asymmetric encryption and authorization certificates. The requester asks the location server for the position of a given target (position query), by sending the authorization certificate released by the target. The certificate is a chipertext encrypted with the public key of the location server and contains the pseudonym of the target. The location server, after decrypting the chipertext, retrieves the target's pseudonym, and satisfies the subject request by releasing the target's position. Note that the location server is not aware of the real identity of the targets. A more complex solution is also provided for queries that ask for a list of targets in a given area. In this case, a certificate specifying the privilege to query a specific area is not enough, but rather the requester has to send the authorizations of all the users relevant for the query. Hengartner and Steenkiste [40, 41] use digital certificates combined with rule-based policies to protect location information. They consider an environment in which users submit requests to a "people locator", which in turn collects the relevant location information through multiple positioning systems. The authors propose an access control mechanism where policies are encoded as digital certificates using SPKI/SDSI. Location policies can specify the entities that can access the location information, the granularity of the information returned to the requester, the location of the requester, and the time allowed for each access. In case of forwarded requests, *trust policies* are used to verify whether the intermediate service is trusted or not to forward a request and receive a response. Finally, *delegation* of right is allowed to grant access to other entities. Atluri and Chun [42] present Geo-Spatial Data Authorization Model (GSAM), an authorization model that protects access to geospatial data. GSAM provides policies evaluating geospatial and temporal characteristics of user's credentials and data objects, and introduces different types of actions (e.g., zoom-in, view, and download). For instance, GSAM defines security and privacy policies that allow access to low resolution images regardless of location coordinates of users, whereas restrict access to high resolution images only for those users located in a particular region.

5 Open Issues

We briefly describe some open issues that need to be taken into consideration in the future development of access control systems for location-based services.

– *Reliable enforcement based on fine-grained context information.* As discussed, a key aspect to the success of location-based access control systems is the definition of a reliable enforcement solution, able to verify information which is approximate and time-variant. In the near future, location servers will provide a wealth of additional environment-related knowledge (e.g., is the user sitting at her desk or walking toward the door? Is she alone or together with others?), that may give the opportunity of defining and evaluating new classes of location-based conditions in the context of LBAC systems. LBAC systems however may be flawed by the intrinsic errors of location measurements, in calculating such fine-grained knowledge. Future access control mechanisms should then try to enhance current approaches to the management of uncertain information, thus providing policy evaluation mechanisms able to support fine-grained location information.

– *Privacy-aware LBAC.* An important aspect to consider in today access control systems is the protection of the user privacy. Some solutions have been presented in the past (e.g., [9]) which provide, on the one side, access control functionality and, on the other side, privacy protection. However, LBAC systems introduce new threats that should be carefully considered. In particular, a fundamental issue to be addressed considers the conflicting requirements of preserving users privacy and of providing high quality LBAC. A suitable protocol should in fact balance the tradeoff between the level of location accuracy requested by LBAC providers and the protection of the location information requested by the users. A possible approach in developing a privacy-aware LBAC may integrate access control with location privacy solutions (e.g., obfuscation [6, 7] and anonymity [43–46] techniques).

– *Integration of different location sources.* An important issue in the development of LBAC systems is represented by the availability of several location servers, which support different positioning systems for measuring location of the users. In this context, a solution which implements communication and negotiation protocols between the LBAC system and multiple, functionally equivalent, location servers is needed. These protocols should provide an approach based on service level agreement attributes which maximize the QoS and/or cost/benefit functions.

6 Conclusions

In this chapter, we discussed how the advent of location-based services and the availability of precise location information are changing traditional access control systems. We considered two different scenarios: *i)* the definition of a location-based access control system, which integrates, evaluates, and enforces traditional access control policies enriched with conditions based on the physical position of users; *ii)* the development of enhanced access control systems for protecting the location information. For both of them, we investigated recent proposals and ongoing work. Finally, we presented open issues that need further investigation.

Acknowledgments

This work was supported in part by the EU, within the 7th Framework Programme (FP7/2007-2013) under grant agreement no. 216483 "PrimeLife".

References

1. Varshney, U.: Location management for mobile commerce applications in wireless internet environment. ACM Transactions on Internet Technology (TOIT) **3**(3) (December 2003) 236–255
2. Ardagna, C., Cremonini, M., Damiani, E., De Capitani di Vimercati, S., Samarati, P.: Privacy-enhanced location services information. In Acquisti, De Capitani di Vimercati, Gritzalis, Lambrinoudakis, eds.: Digital Privacy: Theory, Technologies and Practices. Auerbach Publications (2007)
3. Enhanced 911: Wireless Services. http://www.fcc.gov/911/enhanced/.
4. Chicago Tribune: Rental firm uses GPS in speeding fine. July 2nd, p.9. Associated Press: Chicago, IL, 2001.
5. Duckham, M., Kulik, L.: Location privacy and location-aware computing. In Drummond, J., Billen, R., Forrest, D., Joao, D., eds.: Dynamic & Mobile GIS: Investigating Change in Space and Time. CRC Press (2006) 34–51
6. Ardagna, C., Cremonini, M., Damiani, E., De Capitani di Vimercati, S., Samarati, P.: Location privacy protection through obfuscation-based techniques. In: Proc. of the 21st Annual IFIP WG 11.3 Working Conference on Data and Applications Security, Redondo Beach, CA, USA (July 2007)
7. Duckham, M., Kulik, L.: A formal model of obfuscation and negotiation for location privacy. In: Proc. of the 3rd International Conference on Pervasive Computing (PERVASIVE 2005), Munich, Germany (May 2005)
8. Samarati, P., De Capitani di Vimercati, S.: Access control: Policies, models, and mechanisms. In Focardi, R., Gorrieri, R., eds.: Foundations of Security Analysis and Design. LNCS 2171. Springer-Verlag (2001)
9. Ardagna, C., Cremonini, M., De Capitani di Vimercati, S., Samarati, P.: A privacy-aware access control system. Journal of Computer Security **16**(4) (2008) 369–392

10. De Capitani di Vimercati, S., Foresti, S., Samarati, P.: Recent advances in access control. In Gertz, M., Jajodia, S., eds.: Handbook of Database Security: Applications and Trends. Springer-Verlag (2008)
11. eXtensible Access Control Markup Language (XACML): Version 2.0. (February 2005) http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf.
12. Bonatti, P., Samarati, P.: A unified framework for regulating access and information release on the web. Journal of Computer Security **10**(3) (2002) 241–272
13. Akyildiz, I., Ho, J., eds.: Dynamic mobile user location update for wireless PCS networks. Volume 1. Wireless Networks (1995)
14. Faria, D., Cheriton, D.: No long-term secrets: Location-based security in overprovisioned wireless lans. In: Proc. of the 3rd ACM Workshop on Hot Topics in Networks (HotNets-III), San Diego, CA, USA (November 2004)
15. Garg, S., Kappes, M., Mani, M.: Wireless access server for quality of service and location based access control in 802.11 networks. In: Proc. of the 7th IEEE Symposium on Computers and Communications (ISCC 2002), Taormina/Giardini Naxos, Italy (July 2002)
16. Myllymaki, J., Edlund, S.: Location aggregation from multiple sources. In: Proc. of the 3rd IEEE International Conference on Mobile Data Management (MDM 2002), Singapore (January 2002)
17. Cho, Y., Bao, L., Goodrich, M.: Secure access control for location-based applications in WLAN systems. In: Proc. of the 3rd IEEE International Conference on Mobile Adhoc and Sensor Systems, Vancouver, Canada (October 2006)
18. Nord, J., Synnes, K., Parnes, P.: An architecture for location aware applications. In: Proc. of the 35th Hawaii International Conference on System Sciences, Hawaii, USA (2002)
19. Sastry, N., Shankar, U., Wagner, S.: Secure verification of location claims. In: Proc. of the ACM Workshop on Wireless Security (WiSe 2003), San Diego, CA, USA (September 2003)
20. Zhang, G., Parashar, M.: Dynamic context-aware access control for grid applications. In: Proc. of the 4th International Workshop on Grid Computing (Grid 2003), Phoenix, AZ, USA (November 2003)
21. Atallah, M., Blanton, M., Frikken, K.: Efficient techniques for realizing geo-spatial access control. In: Proc. of the 2nd ACM Symposium on InformAtion, Computer and Communications Security (ASIACCS 2007), Singapore (March 2007)
22. Atluri, V., Shin, H., Vaidya, J.: Efficient security policy enforcement for the mobile environment. Journal of Computer Security **16**(4) (2008) 439–475
23. Ardagna, C., Cremonini, M., Damiani, E., De Capitani di Vimercati, S., Samarati, P.: Supporting location-based conditions in access control policies. In: Proc. of the ACM Symposium on InformAtion, Computer and Communications Security (ASIACCS 2006), Taipei, Taiwan (March 2006)
24. Open Geospatial Consortium: Geospatial eXtensible Access Control Markup Language (GeoXACML) Version 1.0. (February 2008) http://portal.opengeospatial.org/.
25. Marsit, N., Hameurlain, A., Mammeri, Z., Morvan, F.: Query processing in mobile environments: a survey and open problems. In: Proc. of the 1st International Conference on Distributed Framework for Multimedia Applications (DFMA'05), Besancon, France (February 2005)
26. Ardagna, C., Cremonini, M., De Capitani di Vimercati, S., Samarati, P.: Privacy-enhanced location-based access control. In Gertz, M., Jajodia, S., eds.: The Handbook of Database Security: Applications and Trends. Springer-Verlag (2007)

27. Matheus, A.: Declaration and Enforcement of Access Restrictions for Distributed Geospatial Information Objects. PhD Thesis, 2005.
28. Geographic Location/Privacy (geopriv). http://www.ietf.org/html.charters/geopriv-charter.html.
29. Cuellar, J., Morris, J., Mulligan, D., Peterson, J., Polk, J.: Geopriv Requirements. IETF RFC 3693. (February 2004)
30. Danley, M., Mulligan, D., Morris, J., Peterson, J.: Threat Analysis of the Geopriv Protocol. IETF RFC 3694. (February 2004)
31. Cuellar, J.: A Presence-based GEOPRIV Location Object Format. IETF RFC 4119. (December 2005)
32. Schulzrinne, H., Tschofenig, H., Morris, J., Cuellar, J., Polk, J., Rosenberg, J.: Common Policy: A Document Format for Expressing Privacy Preferences. IETF RFC 4745. (February 2007)
33. Hong, D., Yuan, M., Shen, V.Y.: Dynamic privacy management: a plug-in service for the middleware in pervasive computing. In: Proc. of the 7th International Conference on Human Computer Interaction with Mobile Devices & Services (MobileHCI'05), Salzburg, Austria (2005)
34. Langheinrich, M.: Privacy by design-principles of privacy-aware ubiquitous systems. In: Proc. of the 3rd International Conference on Ubiquitous Computing (Ubicomp 2001), Atlanta, GA, USA (September-October 2001)
35. Myles, G., Friday, A., Davies, N.: Preserving privacy in environments with location-based applications. IEEE Pervasive Computing **2**(1) (2003) 56–64
36. Cranor, L.: Web Privacy with P3P. O'Reilly & Associates (2002)
37. World Wide Web Consortium (W3C): Platform for privacy preferences (P3P) project. (April 2002) http://www.w3.org/TR/P3P/.
38. World Wide Web Consortium (W3C): A P3P Preference Exchange Language 1.0 (APPEL1.0). (April 2002) http://www.w3.org/TR/P3P-preferences/.
39. Hauser, C., Kabatnik, M.: Towards Privacy Support in a Global Location Service. In: Proc. of the IFIP Workshop on IP and ATM Traffic Management (WATM/EUNICE 2001), Paris, France (September 2001)
40. Hengartner, U., Steenkiste, P.: Protecting access to people location information. Security in Pervasive Computing (March 2003)
41. Hengartner, U., Steenkiste, P.: Implementing access control to people location information. In: Proc. of the ACM Symposium on Access Control Models and Technologies 2004 (SACMAT 2004), Yorktown Heights, New York, USA (2004)
42. Atluri, V., Chun, S.: An authorization model for geospatial data. IEEE Transactions on Dependable and Secure Computing **1**(4) (2004) 238–254
43. Bettini, C., Wang, X., Jajodia, S.: Protecting privacy against location-based personal identification. In: Proc. of the 2nd VLDB Workshop on Secure Data Management (SDM'05), Trondheim, Norway (September 2005)
44. Ghinita, G., Kalnis, P., Skiadopoulos, S.: Privè: Anonymous location-based queries in distributed mobile systems. In: Proc. of the International World Wide Web Conference (WWW 2007), Banff, Canada (May 2007)
45. Gruteser, M., Grunwald, D.: Anonymous usage of location-based services through spatial and temporal cloaking. In: Proc. of the 1st International Conference on Mobile Systems, Applications, and Services, San Francisco, CA, USA (May 2003)
46. Mokbel, M., Chow, C.Y., Aref, W.: The new casper: Query processing for location services without compromising privacy. In: Proc. of the 32nd International Conference on Very Large Data Bases, Seoul, Korea (September 2006)