An Access Control Model for Location based Services

Cameron Ross Dunne, Thibault Candebat and David Gray

School of Computing, Dublin City University, Dublin 9, Ireland

Abstract. In this paper we propose an access control model for use by a trusted middleware infrastructure, which is part of an architecture that supports the operation of Location Based Services (LBSs) over the Internet. This access control model provides users with increased security, and particularly privacy, by enabling them to create two different types of permissions based on how their location information is being used. These permissions specify which users and LBSs are entitled to obtain location information about which other users, under what circumstances the location information is released to the users and LBSs, and the accuracy of any location information that is released to the users and LBSs.

1 Introduction

The use of mobile devices, such as mobile phones, which can be located, is becoming increasingly popular. These mobile devices enable a new range of services, known as Location Based Services (LBSs), which take the mobile device's location into consideration when providing the service. However, these LBSs raise some security, and in particular privacy, concerns. Users tend to be reluctant to provide personal location information to third party LBSs [4]. Therefore, users must be able to trust LBSs not to misuse their location details.

There are three important security features that enable users to reduce the amount of trust that they must place in LBSs. Firstly, users can specify who is entitled to obtain their location information [7]. Secondly, users can specify the circumstances in which their location information is released. These circumstances can be based on factors such as location [1] or current activity [7]. Thirdly, users can specify the accuracy of any location information that is released to other users and LBSs. Typically, an increased trust is rewarded with an increased accuracy. It is likely that users will require a range of security requirements from very simple security specifications to very complex and personalised security specifications [1].

In this paper we focus on providing users with increased security by proposing an access control model. This access control model is based on a system where users create permissions that specify who is entitled to obtain their location information, under what circumstances this location information is obtainable, and the accuracy of this location information. Our access control model is then responsible for releasing location information about users in accordance with their permissions. Since users create permissions regarding their own location information, our access control model provides a form of *Discretionary Access Control* (DAC).

Ross Dunne C., Candebat T. and Gray D. (2008).
An Access Control Model for Location based Services.
In *Proceedings of the 6th International Workshop on Security in Information Systems*, pages 49-58
DOI: 10.5220/0001737200490058

Our architecture assumes that a user, who is requesting location information about another user, communicates directly with the LBS that provides the relevant service. This LBS then communicates with an independent entity, which we refer to as the infrastructure, in order to obtain the necessary location information. This is shown in Figure 1. Therefore, both the user and the LBS are seeking location information from the point-of-view of the infrastructure. This in effect creates two different *subjects* in the context of an access control model.

The main novelty of our access control model is that it enables users to specify two different types of permission. The first type of permission is used to specify which users are trusted, and therefore can obtain location information, and the second type of permission is used to specify which LBSs are trusted, and therefore can obtain location information. This has the effect of creating a *whitelist* for users, and a separate whitelist for LBSs. The access control model will only allow location information to be released if it is presented with both a valid permission specified in terms of users, and a valid permission specified in terms of LBSs.

The remainder of this paper is structured as follows: Section 2 introduces the concepts on which we build our access control model. Section 3 describes our access control model as a mathematical model. Section 4 describes an abstract syntax for expressing permissions. Section 5 describes the implementation of our access control model. Section 6 compares our access control model to other related work. Finally, Section 7 presents our conclusions.

2 Background

Our architecture consists of five entities. A *locatable* entity is a mobile device that can be located. A *network operator* manages and maintains a network of locatables, and therefore it is capable of producing location information for each locatable. An *LBS* is a service that is operated by a party who is normally independent of the network operator. The *infrastructure* entity is responsible for providing all of the common functionality that is required by LBSs to manage the users' identity and location information. In particular, the infrastructure is responsible for hosting an implementation of the access control model. A *user* is an entity that subscribes to a trusted infrastructure. This enables him/her to invoke LBSs using either his/her mobile device or a desktop computer. Users own locatables, and therefore we refer to locating users rather than locating locatables.

Within our architecture users communicate directly with LBSs, and LBSs communicate directly with the infrastructure. These communications occur using secure channels. The infrastructure is designed to operate as a *middleware* entity between the LBSs and the network operators. An overview of this architecture is a shown in Figure 1.

Every user has one or more unique *public names*, and these are used to identify him/her to other users and to the LBSs. Public names identify users either by real names or by pseudonyms [9]. Every LBS has a single public name that is unique. Every user knows his/her complete mapping, and the infrastructure knows the complete mapping for every user. The infrastructure provides no external services that enable users or LBSs to link users with public names. The only way that this linkage can be discovered is if the user who is being linked describes the relationship using a different medium.

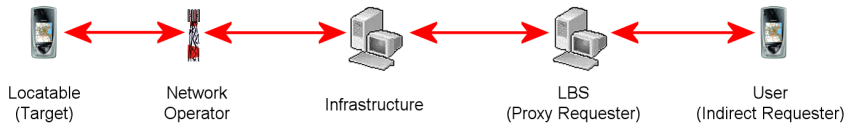

Fig. 1. Architectural Diagram Showing Entities and Roles (in Parentheses).

There are two different roles that are identified within our access control model. *Targets* are users who need to be located by the infrastructure as part of the delivery of LBSs. Therefore, it is their privacy that the access control model is protecting. *Requesters* are users and LBSs who request location information about targets from the infrastructure. An *indirect requester* is a user who requests location information from an LBS, which then accesses the infrastructure. Such an LBS is known as a *proxy requester*. Both targets and requesters are identified using their public names, and a single mobile device can be both a target and an indirect requester simultaneously. These roles are shown in Figure 1.

A sighting for a target contains a location component that describes where the target was sighted, and a time interval that describes when the target was sighted. The *sighting accuracy* of a sighting is a measure of the quality of the sighting such that a more accurate sighting has a smaller location and/or time interval. *Sighting blurring* is the process of taking an existing sighting and a desired sighting accuracy, and creating a new sighting that has a sighting accuracy that is less than or equal to the desired sighting accuracy. A metric for describing sighting accuracy, as well as several sighting blurring techniques, are presented in [2].

Typically, our access control model will be used in the following scenario: a user, as an indirect requester, invokes an LBS, as a proxy requester, by sending a request for sighting information about users, as targets. The proxy requester contacts the infrastructure and for each target it requests a sighting. The infrastructure invokes the access control model for each target in order to determine if the target is willing to allow the release of its sightings, and at which sighting accuracy. The infrastructure obtains a sighting from the network operator for each allowed target, and then blurs this sighting using a sighting blurring algorithm and the sighting accuracy output by the access control model. The infrastructure returns the targets' blurred sightings to the proxy requester. The proxy requester normally processes the sightings further, and combines them with additional information. For example, the proxy requester might generate maps containing the sightings, or provide directions to a location. Finally, the proxy requester sends this processed sighting information to the indirect requester.

3 Access Control Model

3.1 Types

There are three different types defined within the mathematical model of our access control model. We define $\mathcal{N}$ as the set of public names for users and LBSs. These public names are strings that have a consistent syntax. We define $\mathbb{P}(\mathcal{N})$ as the power set of $\mathcal{N}$. We define $\mathbb{B}$ as the set of boolean values. Therefore, $\mathbb{B} = \{\texttt{True}, \texttt{False}\}$. We define $\mathbb{A}$ as the totally ordered set of sighting accuracies with a least element $\alpha_\perp$, which is used to mean no sighting accuracy.

3.2 Permissions

An *Indirect Access Permission* (IAP) is used by a target to specify which indirect requesters can indirectly access the infrastructure via a proxy requester to obtain its sightings, and at what accuracy these sightings can be obtained. We define the set of IAP permissions as $\mathcal{P}_{\texttt{IAP}} = \mathcal{N} \times \mathbb{P}(\mathcal{N}) \times \mathbb{P}(\mathcal{N}) \times \mathbb{B} \times \mathbb{A}$. Given the IAP $\langle t, I, P, c, \alpha \rangle \in \mathcal{P}_{\texttt{IAP}}$ we have:

- t is the public name of the target that created this permission.
- I specifies which indirect requesters are allowed to use this permission to obtain sightings of t indirectly via a proxy requester in P.
- P specifies which proxy requesters are allowed to use this permission to act as a proxy in order to obtain sightings of t directly from the infrastructure on behalf of an indirect requester in I.
- c is used to determine if a sighting will be released based on parameters that are outside the scope of both the access control model and the sighting blurring algorithm. The access control model will never release a sighting accuracy if this value is `False`. Therefore, if t does not want to specify any condition, then it simply sets this value to `True`.
- α is the accuracy at which t can be sighted.

A *Proxy Access Permission* (PAP) is used by a target to specify which proxy requesters can directly access the infrastructure when operating as proxies for indirect requesters to obtain sightings of the target, and at what accuracy these sightings can be obtained. We define the set of PAP permissions as $\mathcal{P}_{\texttt{PAP}} = \mathcal{N} \times \mathbb{P}(\mathcal{N}) \times \mathbb{P}(\mathcal{N}) \times \mathbb{B} \times \mathbb{A} \times \mathbb{B}$. Given the PAP $\langle t, P, I, c, \alpha, o \rangle \in \mathcal{P}_{\texttt{PAP}}$ we have:

- t, P, I, c, and α all have the same meaning as they do in the IAP.
- o is used to specify which of the sighting accuracies used in the IAP and the PAP has priority. If this value is `True`, then the access control model should use the sighting accuracy specified in this PAP. Otherwise the sighting accuracy specified in the IAP is used. This priority is assigned to the appropriate sighting accuracy regardless of which sighting is more accurate. In other words, the override parameter can be used to increase or decrease the sighting accuracy specified in the IAP. t sets this parameter based on its preferences, and t's reasoning for using this parameter is external to the access control model.

The access control model will only allow a sighting to be released if it is presented with both a valid IAP and a valid PAP.

3.3 Boolean Expressions

Our access control model is based on permissions that contain sets of requesters. There are two significant disadvantages associated with this approach where the sets of requesters are stated explicitly. Firstly, targets must revoke old permissions and create new permissions every time that they want to add or remove a requester from an existing permission. Secondly, every time that the access control algorithm is invoked it must enumerate entire sets of requesters. As the sets of requesters in permissions grow larger, this approach becomes more cumbersome. The disadvantages of this approach can be overcome by specifying these sets of requesters using set comprehension. However, calculating the set of requesters satisfying some predicate would be inefficient.

Fortunately, our access control algorithm only needs to perform a membership test in order to determine if a given requester is a member of a given set of requesters. This enables us to represent sets as predicates over requester names. Therefore, we will use boolean expressions, which contain a free variable representing the name of the requester to be tested, in our notation instead of predicates.

4 Notation

4.1 Abstract Syntax

The use of boolean expressions allows us to define the abstract syntax of IAPs and PAPs as follows:

$$IAP ::= \langle Name, B_{\texttt{Exp}}, B_{\texttt{Exp}}, B_{\texttt{Exp}}, Accuracy \rangle$$
$$PAP ::= \langle Name, B_{\texttt{Exp}}, B_{\texttt{Exp}}, B_{\texttt{Exp}}, Accuracy, B_{\texttt{Exp}} \rangle$$

A $Name$ is a syntactic representation of a value from $\mathcal{N}$, and $Accuracy$ is a valid accuracy from $\mathbb{A}$. We do not allow boolean expressions to be used in permissions in place of the target's name. This is because each permission is created by a single user, and each permission is only applicable to its creator.

The concept of a boolean expression is well understood, and therefore we only provide a partial definition of its abstract syntax:

$$\begin{aligned} B_{\texttt{Exp}} ::= {} & \texttt{True} \mid \texttt{False} \mid \neg B_{\texttt{Exp}} \\ & \mid B_{\texttt{Exp}} \wedge B_{\texttt{Exp}} \mid B_{\texttt{Exp}} \vee B_{\texttt{Exp}} \\ & \mid User \in \{Userlist\} \mid User \in \{User.Attribute\} \\ & \mid User.Attribute \mid User.Attribute = Value \\ & \mid \ldots \end{aligned}$$

A $User$ is either an explicitly named user, or a variable that represents a user. There are three different variables that targets can use in the boolean expressions within the permissions that they create. The target itself is represented using `#t`, the indirect requester is represented using `#i`, and the proxy requester is represented using `#p`. `System` is a special user that represents the system that is hosting the access control

model. These variables are needed because the target does not necessarily know who these entities are when it is creating the permission. They are replaced with the actual values by the access control model immediately before the access control algorithm is invoked. The partial definition of the abstract syntax of a $User$ is:

$$User ::= Name \mid \#\texttt{t} \mid \#\texttt{i} \mid \#\texttt{p} \mid \texttt{System} \mid User.Attribute \mid \ldots$$

An $Attribute$ is the name of an attribute of a $User$. There are many different attribute names, and these will depend on the underlying implementation of the access control model. $Userlist$ and $Value$ both have implicit definitions of their abstract syntax, and therefore we have not included them in this paper.

4.2 Access Control Algorithm

Our access control model uses the access control algorithm described in Algorithm 1 when an indirect requester accesses the infrastructure via a proxy requester. There are four inputs to the access control algorithm. These are the public name of the indirect requester, the public name of the proxy requester, an IAP, and a PAP. The output of the access control algorithm is always a sighting accuracy. However, in circumstances where the access control algorithm determines that the target is unwilling to allow the requesters to obtain a sighting then the sighting accuracy will be $\alpha_\perp$.

Algorithm 1: The Access Control Algorithm.

Input: i is the public name of the indirect requester.
Input: p is the public name of the proxy requester.
Input: The IAP $\langle t_1, i_1, p_1, c_1, \alpha_1 \rangle$.
Input: The PAP $\langle t_2, p_2, i_2, c_2, \alpha_2, o \rangle$.
Output: The allowed sighting accuracy.
if $t_1 \neq t_2$ **then return** $\alpha_\perp$
if $(\neg i_1) \vee (\neg i_2)$ **then return** $\alpha_\perp$
if $(\neg p_2) \vee (\neg p_1)$ **then return** $\alpha_\perp$
if $(\neg c_1) \vee (\neg c_2)$ **then return** $\alpha_\perp$
if o **then return** α_2
return α_1

4.3 Examples

In order to demonstrate how our access control algorithm operates on IAPs and PAPs that contain boolean expressions we will present some examples that are based on the following example definitions of $Name$ and $Accuracy$:

$$Name ::= \texttt{Stefano} \mid \texttt{Ilaria} \mid \texttt{Maria} \mid \texttt{Alexia} \mid \texttt{FriendFinder}$$
$$Accuracy ::= \alpha_\perp \mid \alpha_4 \mid \alpha_3 \mid \alpha_2 \mid \alpha_1$$

The boolean expressions in our examples also use the following attributes:

- `isUser` is a boolean attribute that is `True` if a $User$ is a user, and `False` if a $User$ is an LBS.

- `IMStatus` is a string attribute that represents a $User$'s instant messenger status.
- `Day` is a time attribute of the `System` user that represents the current day.

The following examples are based on two requesters indirectly accessing the infrastructure with an IAP and a PAP.

- The inputs to the access control algorithm are the indirect requester `Ilaria`, the proxy requester `FriendFinder`, and the following IAP and PAP:

$$\langle \texttt{Maria}, \#\texttt{i} \in \{\texttt{Ilaria}, \texttt{Alexia}\}, \neg\#\texttt{p.isUser}, \texttt{True}, \alpha_3 \rangle$$
$$\langle \texttt{Maria}, \#\texttt{p} \in \{\texttt{FriendFinder}\}, \#\texttt{i.isUser}, \texttt{True}, \alpha_\perp, \texttt{False} \rangle$$

  The access control algorithm will return a sighting accuracy of α_3. This IAP effectively allows `Ilaria` to obtain sightings for `Maria`, with a sighting accuracy of α_3, using any LBS that `Maria` trusts. However, if `Maria` has a higher trust in sighting requests jointly from `Ilaria` and `FriendFinder`, then `Maria` creates the following IAP that can be used with the previous PAP:

$$\langle \texttt{Maria}, \#\texttt{i} \in \{\texttt{Ilaria}, \texttt{Alexia}\}, \#\texttt{p} \in \{\texttt{FriendFinder}\}, \texttt{True}, \alpha_2 \rangle$$

- The inputs to the access control algorithm are the indirect requester `Ilaria`, the proxy requester `FriendFinder`, and the following IAP and PAP:

$$\langle \texttt{Stefano}, \#\texttt{i} \in \{\texttt{Ilaria}, \texttt{Maria}, \texttt{Alexia}\}, \neg\#\texttt{p.isUser}, \texttt{True}, \alpha_1 \rangle$$
$$\langle \texttt{Stefano}, \#\texttt{p} \in \{\texttt{FriendFinder}\}, \#\texttt{i.isUser}, \neg(\#\texttt{System.Day} = \text{“}\texttt{Sunday}\text{”}), \alpha_4, \texttt{True} \rangle$$

  The access control algorithm will return a sighting accuracy of $\alpha_\perp$ if it is invoked on a Sunday, or α_4 if it is invoked on any other day.
- The inputs to the access control algorithm are the indirect requester `Maria`, the proxy requester `FriendFinder`, and the following IAP and PAP:

$$\langle \texttt{Stefano}, \#\texttt{i} \in \{\texttt{Ilaria}, \texttt{Maria}, \texttt{Alexia}\}, \neg\#\texttt{p.isUser}, \texttt{True}, \alpha_1 \rangle$$
$$\langle \texttt{Stefano}, \neg\#\texttt{p.isUser}, \#\texttt{i} \in \{\texttt{Ilaria}, \texttt{Maria}, \texttt{Alexia}\}, \texttt{True}, \alpha_\perp, \texttt{False} \rangle$$

  The access control algorithm will return a sighting accuracy of α_1. These permissions have the effect of allowing `Ilaria, Maria` and `Alexia` to indirectly retrieve `Stefano`'s sightings with an accuracy of α_1 using any LBS as the proxy requester. Therefore, this PAP enables `Stefano` to allow `Ilaria, Maria` and `Alexia` to delegate the sighting request rights that he gave them to any LBS.

4.4 Semantics

The access control model must enforce certain semantic rules at run-time, and if these rules are violated then it will return a sighting accuracy of $\alpha_\perp$. Consider an example where the inputs to the access control algorithm are the indirect requester `Ilaria`, the proxy requester `FriendFinder`, and the following IAP and PAP:

$$\langle \texttt{Maria}, \#\texttt{i} \in \{\texttt{Ilaria}, \texttt{Alexia}\}, \neg\#\texttt{p.isUser}, \texttt{Alexia.IMStatus} = \text{“}\texttt{Online}\text{”}, \alpha_2 \rangle$$
$$\langle \texttt{Maria}, \#\texttt{p} \in \{\texttt{FriendFinder}\}, \#\texttt{i.isUser}, \texttt{True}, \alpha_\perp, \texttt{False} \rangle$$

The access control algorithm will return a sighting accuracy of α_2 if Alexia's instant messenger status is "Online", and otherwise it will return a sighting accuracy of $\alpha_\perp$. If Ilaria successfully uses this IAP to obtain a sighting accuracy of α_2 for Maria, then Ilaria can determine that Alexia's instant messenger status was "Online". If Maria can determine that Ilaria successfully uses this IAP to obtain a sighting accuracy of α_2, then Maria also knows that Alexia's instant messenger status was "Online". However, Alexia has not been part of this invocation of the access control model, and she might be unwilling to share her instant messenger status with Ilaria and Maria.

Therefore, only the target's attributes and the requesters' attributes can be accessed within a permission, and the access control model returns a sighting accuracy of $\alpha_\perp$ if it is presented with a permission that contains attributes for any other user. [1]

The use of the targets' and requesters' attributes within permissions can reveal additional information. For example, consider that the inputs to the access control algorithm are the indirect requester Ilaria, the proxy requester FriendFinder, the previous PAP, and the following IAP:

$$\langle \texttt{Maria}, \#\texttt{i} \in \{\texttt{Ilaria}, \texttt{Alexia}\}, \neg\#\texttt{p.isUser}, \#\texttt{i.IMStatus} = \text{“}\texttt{Online}\text{”}, \alpha_2 \rangle$$

The access control algorithm will return a sighting accuracy of α_2 if Ilaria's instant messenger status is "Online", and otherwise it will return a sighting accuracy of $\alpha_\perp$. Again, if Ilaria successfully uses this IAP to obtain a sighting accuracy of α_2 for Maria, then Maria can determine that Ilaria's instant messenger status was "Online". Initially, this ability of a target to obtain information about the requesters appears to be a breach of the requesters' security. However, there is no loss of security associated with this ability. This is because each requester can inspect the permission to determine which of its attributes will be accessed, and hence shared with the target. If it is not willing to share these attributes with the target then it does not use the permission. Similarly, the target will not create any permissions containing attributes that reveal additional information to the requesters.

5 Implementation

Our access control model can be implemented as either a centralised system or a distributed system. In the centralised implementation the infrastructure is responsible for storing and selecting permissions. The main advantage of this implementation is that the requesters do not need to store and manage permissions. However, there are also significant disadvantages associated with this implementation. In particular, the infrastructure must determine which combinations of IAPs and PAPs to use. This could be difficult, because it is likely that there will be many suitable combinations for any indirect requester and proxy requester pair.

In the distributed implementation the requesters are responsible for storing and selecting permissions. When an indirect requester and a proxy requester pair require a sighting of a target, the indirect requester selects an IAP and the proxy requester selects

[1] The System user is an exception, because its attributes can be contained in any permission.

a PAP. Both of these permissions are then supplied to the infrastructure. The main advantage of this implementation is that there is no need to centrally store and manage permissions, and this facilitates scalability. Also, requesters can choose the permissions that they want to present to the infrastructure, which gives them complete control over permission selection. The most significant disadvantage of this implementation is that users must manage the permissions themselves, and these permissions may need to be stored on mobile devices that have limited resources. The distributed nature of these permissions raises some important security issues, and in particular, it must be possible to verify the authenticity and integrity of permissions. Therefore, we propose that users sign the permissions that they create, and users have the ability to revoke the permissions that they create. This requires the use of a *Public Key Infrastructure* (PKI).

6 Related Work

Leonhardt and Magee propose a centralised access control model in the context of LBSs [8]. If more than one subject is included in a permission, then it can only be used by all of the subjects simultaneously. The main difference between their access control model and ours is that theirs requires a separate permission for every indirect requester and proxy requester pair. In contrast, our access control model enables a single IAP to be used in combination with many PAPs, and a single PAP can be used with many IAPs.

Hauser and Kabatnik propose a distributed implementation of an access control model that is based on public key cryptography and certificates [5]. The distributed implementation of our access control model is similar to their implementation because both implementations enable targets to create certificates that contain the permissions. The main distinction is that their access control model assumes that there will only be one requester involved in each request for a target's sighting information. In contract to this, our access control model facilitates sighting requests that are jointly from indirect requesters and proxy requesters.

Hengartner and Steenkiste propose an access control model that allows users to specify permissions in terms of users, and trust in terms of services [6]. Trusted services are then allowed to receive location information on behalf of users who are the subjects of the permissions. Both users and services are capable of delegating their rights to other users and services to form delegation chains. Public key cryptography and certificates are used to implement these permissions and trusts. Our access control model treats both users and LBSs as equally important types of requesters, and therefore users can specify permissions in for each type of requester. In contrast, the services in this access control model always inherit the permissions allowed for the users. Additionally, our access control model does not support delegation chains.

Atluri and Shin present an access control model based on a data structure for combining users' sightings with their profiles and permissions [3]. The main distinction is that their access control model assumes that there will only be one requester involved in each request for a target's sighting information. However, our access control model facilitates sighting requests that are jointly from indirect requesters and proxy requesters.

7 Conclusions

In this paper we presented an access control model that is based on users who are targets creating permissions for users who are indirect requesters, and for LBSs that are proxy requesters. These permissions specify which requesters are entitled to obtain sightings of which targets, under what circumstances these sightings are released, and the accuracy of these sightings. Our access control model is based upon a mathematical model, and we have provided partial abstract definitions of the permissions. Our access control model can be implemented as either a centralised system where the infrastructure is responsible for storing and selecting permissions, or as a distributed system where the requesters are responsible for storing and selecting permissions.

Acknowledgements

The authors wish to thank Enterprise Ireland for its support with this research under grants IF/2002/336 and PC/2004/446. In addition, the authors wish to thank the anonymous reviewers for their suggestions and comments.

References

1. D. Anthony, T. Henderson, and D. Kotz. Privacy in Location Aware Computing Environments. *IEEE Pervasive*, 6(4):64–72, Oct–Dec 2007.
2. C. A. Ardagna, M. Cremonini, E. Damiani, S. D. C. di Vimercati, and P. Samarati. Location Privacy Protection Through Obfuscation-Based Techniques. In *DBSec*, volume 4602 of *Lecture Notes in Computer Science*, pages 47–60. Springer, 2007.
3. V. Atluri and H. Shin. Efficient Security Policy Enforcement in a Location Based Service Environment. In S. Barker and G.-J. Ahn, editors, *DBSec*, volume 4602 of *Lecture Notes in Computer Science*, pages 61–76. Springer, 2007.
4. L. Barkhuus and A. Dey. Location-Based Services for Mobile Telephony: a study of users' privacy concerns. In *Proceedings of IFIP INTERACT03: Human-Computer Interaction*, page 709. IFIP Technical Committee No 13 on Human-Computer Interaction, 2003.
5. C. Hauser and M. Kabatnik. Towards Privacy Support in a Global Location Service. In *IFIP Workshop on IP and ATM Traffic Management, Paris*, pages 81–89, 2001.
6. U. Hengartner and P. Steenkiste. Protecting Access to People Location Information. In *Security in Pervasive Computing: First International Conference, Boppard, Germany, March 12-14, 2003. Revised Papers*, volume 2802 / 2004, pages 25–38, 2004.
7. S. Lederer, J. Mankoff, and A. K. Dey. Who wants to know what when? privacy preference determinants in ubiquitous computing. In *CHI '03: CHI '03 extended abstracts on Human factors in computing systems*, pages 724–725, New York, NY, USA, 2003. ACM.
8. U. Leonhardt and J. Magee. Security Considerations for a Distributed Location Service. *Journal of Network and Systems Management*, 6(1):51–70, 1998.
9. A. Pfitzmann and M. Hansen. Anonymity, Unlinkability, Undetectability, Unobservability, Pseudonymity, and Identity Management - A Consolidated Proposal for Terminology. Technische Universität Dresden, Version v0.31, 15/2/2008.