Protecting Access to People Location Information

Urs Hengartner[1] and Peter Steenkiste[1,2]

[1] Department of Computer Science
[2] Department of Electrical and Computer Engineering
Carnegie Mellon University
{uhengart,prs}@cs.cmu.edu

Abstract. Ubiquitous computing provides new types of information for which access needs to be controlled. For instance, a person's current location is a sensitive piece of information and only authorized entities should be able to learn it. There are several challenges that arise for the specification and implementation of policies controlling access to location information. For example, there can be more than one source of location information, policies need to be flexible, conflicts between policies might occur, and privacy issues need to be taken into account. Different environments handle these challenges in a different way. We present the design of an access control mechanism for a system providing location information that addresses the challenges above and that can be deployed in different environments. Finally, we demonstrate feasibility of our design with an example implementation based on digital certificates.

1 Introduction

Ubiquitous computing environments, such as the ones examined in CMU's Aura project [6], rely on the availability of people location information to provide location-specific services. However, location is a sensitive piece of information. Very often, it is possible to gather information about the current activities of a person from her location. Therefore, only authorized entities should have access to her location information.

Location information has received increased attention while its access control requirements have not been studied thoroughly. Location information is inherently different from information such as files stored in a file system, whose access control requirements have been studied widely. Location information is different since there is no single point to which access needs to be controlled. Instead, a variety of sources (e.g., a personal calendar or a GPS device) can provide location information. In addition, different types of queries can provide the same information (see Section 3). Therefore, a system providing location information has to perform access control in a distributed way, considering different services and interactions between queries.

The contribution of our work is threefold. First, we discuss challenges that arise when specifying access control policies for location information in different environments. Second, we provide the design of an access control mechanism that is flexible

enough to be deployed in different environments, having multiple sources of location information. Third, we present an implementation of the proposed mechanism, which is based on digital certificates.

The outline of the rest of this paper is as follows: Section 2 introduces the architecture of a location system. In Section 3, we discuss several challenges that arise when specifying location policies. We elaborate on dealing with multiple sources of location information in Section 4. In Section 5, we present the design of our access control mechanism. We discuss our prototype implementation in Section 6. We elaborate on related work in Section 7 and on our conclusions and future work in Section 8.

2 People Location System

In this section, we introduce the architecture of a people location system that exploits different sources of location information.

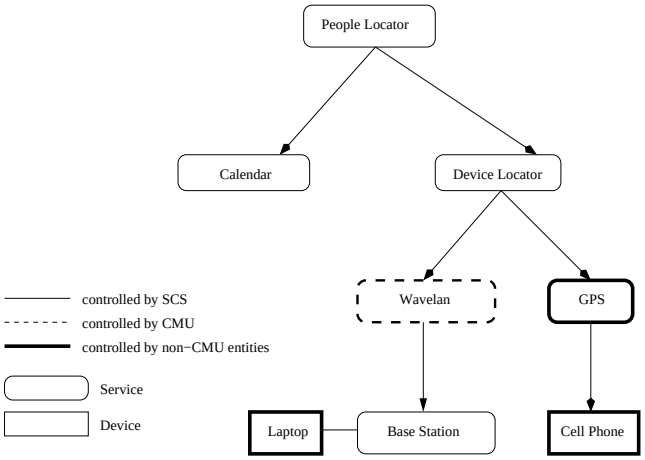

Fig. 1. *Example location system.*

We assume a hierarchical structure for the location system. Figure 1 shows an example of such a system, as it could be deployed in CMU's School of Computer Science (SCS). The nodes in the graph are either services or devices, the arrows denote which service contacts which other service. The *location system* is a composition of multiple *location services*. Each location service either exploits a particular technology for gathering location information or processes location information received from other location services. A client contacts the People Locator service at the root of the system. This service then contacts other services which themselves may also contact other services. Location information flows in the reverse direction of a request (not shown in the figure). A location service can be implemented either on a single node or on multiple nodes to improve scalability and robustness.

There are two groups of location services. The first group consists of services that are aware of people. The second group of services is aware of devices. These services

locate a user indirectly by locating the device(s) she is carrying with her. The People Locator service, the calendar service, and the Device Locator service belong to the first group. The People Locator service aggregates information received from other services. The calendar service looks at people's appointments to determine their current location. The Device Locator service maps a query for a person to potentially several queries for her devices and contacts the corresponding services. In our example, this group of services consists of the Wavelan service and the GPS service. The Wavelan service keeps track of the location of wireless devices by identifying the base station(s) they are connecting to. The GPS service calls GPS-enhanced mobile phones and retrieves their geographical location. We believe that our location system can easily incorporate other location services (e.g., Microsoft's Radar [1] or MIT's Cricket [13]).

A basic assumption in our work is that different entities may administrate the various services. In our example, SCS's computing facilities control the calendar service, CMU's computing facilities administrate the Wavelan service, and a phone company runs the GPS service.

3 Location Policies

Location queries can originate both from people and from services. In the rest of this paper, we are going to call an entity that issues a query "location seeker". A query can ask either for the location of a user ("user query") or for the people in or at a geographical location, such as a room in a building ("room query"). Based on these two basic queries, it becomes possible to build more sophisticated queries or services that provide location-specific information.

To prevent location information from leaking to unauthorized entities, we employ location policies. An entity can access location information about a person or about the people in a room only if permitted by that person's and that room's location policy, respectively. In this section, we examine location policies and present requirements that need to be provided by the access control mechanism of a people location system.

3.1 User and Room Location Policies

Corresponding to the two types of queries, there are two types of location policies: user policies and room polices. A user policy states who is allowed to get location information about a user. For example, "Bob is allowed to find out about Alice's location". Similarly, a room policy specifies who is permitted to find out about the people currently in a room. For example, "Bob is allowed to find out about the people in Alice's office".

In addition, both user and room policies should be able to limit information flow in some more ways. Namely, we believe that at least the following properties should be controllable:

Granularity. A policy could restrict the granularity of the returned location information. For example, a user policy can state that the building in which a queried

user is staying is returned instead of the actual room (e.g., "CMU Wean Hall" vs. "CMU Wean Hall 8220"). A room policy can require that the number of people in a room is returned instead of the identity of the people in the room (e.g., "two people" vs. "Alice and Bob").

Locations/users. User policies can contain a set of locations (e.g., buildings or rooms). The location system will return location information only if the queried user is at one of the listed locations. For example, "Bob is allowed to find out about Alice's location only if she is in her office". Similarly, room policies can include a set of users. The answer to a room query will include only users listed in the policy (provided they are in the room).

Time intervals. Location policies can limit time intervals during which access should be granted. For example, access can be restricted to working hours.

Earlier work (e.g., by Spreitzer and Theimer [14]) lets users have influence on room policies. As part of their user policy, users can specify whether room queries for a room will include them in their answer. While this approach is appropriate for some scenarios, it is not for others. We argue that in most cases, the owner of a room should always be able to find out who is in her room, regardless of the user policies of the users in her room.

3.2 User vs. Institutional Policies

Depending on the environment, different entities specify location policies. For some environments, a central authority defines policies, whereas for others, users set them. In addition, some environments might give both users and a central authority the option to specify policies.

In general, governments and companies probably do not want the location of their employees or the people in their buildings to be known to outsiders, whereas this information can be delivered to (some) entities within the organization. In such cases, a central authority would establish the location policies such that no information is leaked. For other environments, such as a university or a shopping mall, the institution behind the environment cares less about where an individual is or who is in a room. For these cases, it should be up to an individual to specify her user policy. We examine some example environments in more detail in Section 3.6.

In the rest of the paper, we are going to call the entity that specifies location policies "policy maker". A location system should be flexible enough to support different policy makers, depending on the environment.

3.3 Transitivity of Access Rights

If Bob is granted access to Alice's location information, should he be allowed to forward this access right to Carol? If Ed is given the right to find out about the people in his office, should he be allowed to grant this privilege also to Fred? In short, should access rights to location information be transitive?

There is no simple answer to this question. Again the answer depends on the environment. The location system should thus let policy makers explicitly state whether they want access rights to be transitive. However, note that even though a

user might not be allowed to forward his access rights to other users, he could still proxy for them.

3.4 Conflicting Policies

User and room location policies can conflict. For example, assume that Alice does not allow Bob to locate her, but Carol allows Bob to locate people in her office. If Alice is in Carol's office, should the location system tell Bob about it? There are multiple ways for dealing with this issue:

- The system ignores the room policy when answering user queries. Similarly, it ignores the user policy for room queries. In our example, Bob would thus see Alice being in Carol's office.
- The system looks at both policies for any request and returns information only if it is approved by both of them. Bob would thus not see Alice being in Carol's office.
- The user and room policies are established in a synchronized fashion so that no conflicts arise. For example, Leonhardt and Magee [12] suggest authorizing user/room pairs. Alice and Carol's location policies would thus have to be rewritten.

The approach that fits best depends on the environment in which the location system is deployed.

3.5 Privacy Issues

Location policies can contain personal data that users might want to have kept private. For example, a user might grant access rights to his friends and thus define a set of friends. The decision about who is (not) in this set can be delicate and the user thus wants to keep the set secret. A location system should allow users to keep their policies secret.

3.6 Example Environments

In this section, we discuss how location policies are specified and applied in two different environments; a hospital and a university environment.

Hospital Medical data such as patient information is typically protected based on a multilateral security model, which protects information flow between compartments. For example, only doctors taking care of a patient have access to her medical data, but not every doctor in the hospital. A similar model is required for location information. Only the doctors of a patient should be able to locate her. In addition, a patient should be able to allow other people (e.g., her husband) to locate her. To fulfill these requirements, the hospital has a central authority establish policies. It can give the patient the right to include additional people in her location policy.

The central authority should also establish room policies to protect the patients' privacy. User and room location policies do not need to be synchronized for the hospital scenario.

User policies that allow patients to give other people access might be transitive, whereas room policies are not.

University In a university setting, students and faculty members specify their user policies. However, room policies are established by a central authority.

For offices, the authority is likely to delegate this right to to the occupant of the office. For lecture rooms and hallways, the authority typically would set the room policy such that room and user policies become synchronized. That is, upon a room query, the location system consults the user policies of users in the room/hallway before returning their identity. For offices, user and room policies are typically not synchronized.

User policies might be transitive. For room policies, the institution might decide to not let the occupant of an office re-delegate his rights to other people.

4 Service Trust

As explained in Section 2, a location system can consist of multiple location services. Some of these services, such as the People Locator service shown in Figure 1, do not generate their own location information. Instead, they process location information received from other services. To avoid information leaks, the location system must ensure that only services that implement access control checks are given location information.

A possible way to implement this condition would be to require that the user or room policy grants a service access to location information before it is given this information. This option is appropriate for services that must be able to issue requests for location information. For example, the Device Locator service shown in Figure 1 has to create queries for devices upon receiving a user query.

However, for services such as the People Locator service, this option gives the services more privileges than they really need to have. The People Locator service only forwards requests received from clients. By granting it the right to issue requests, we increase its exposure in case of a break in. If an intruder breaks into the service and issues requests on its behalf, these requests will be granted access by the location system.

Due to these reasons, we introduce the concept of service trust. If a service such as the People Locator service is trusted, it is given location information, even if it is not allowed to issue requests. For a particular query, a service needs to be trusted by the policy maker that defined the corresponding user or room policy. The trust assumption is that the service implements access control as follows:

In the first step, the service checks whether the policy maker has granted access to the entity issuing the request. Only if this check is successfull, the service proceeds to the second step, else access is denied. In the second step, the service checks whether the entity from which it received the request corresponds to the entity that issued the request. If it does, access is granted. If it does not, the service has

to verify whether the policy maker trusts the entity before access is granted. Else access is denied.

How can we verify whether a service fulfills its trust assumption? We require services to sign whatever location information they return to achieve non-repudiation. Therefore, an entity trusting a service can reactively identify misbehaving services and revoke the trust in them.

Whoever specifies the location policy for a user or a room also identifies the corresponding trusted services. Therefore, for a hospital, the set of trusted services should be defined by the central authority. For a university, each user and each owner of a room define their own set of trusted services.

5 System Design

Based on our discussion in Sections 3 and 4, we now present the design of our location system. We build on three main concepts. First, services respond to a location request only after performing a location policy check that verifies that the location seeker is given access. Second, services verify that the service from which they receive a forwarded request is trusted before returning an answer to it. Third, services can delegate both location policy and service trust checks to other services; delegation can be used to eliminate redundant checks. In this section, we motivate and discuss these concepts.

5.1 Digital Certificates

For implementation purposes, we assume that policy and trust decisions are stated in digital certificates. A digital certificate is a signed data structure in which the signer states a decision concerning some other entity. There are various kinds of digital certificates. Two examples are KeyNote [2] and SPKI/SDSI [5] certificates. Our implementation is based on SPKI/SDSI certificates, which we discuss in Section 6.1.

We introduce the following notation for expressing a policy or trust decision:

$$A \xrightarrow[scope]{type} B.$$

A is the entity making a decision concerning B. Above the arrow, the type of decision is given ("*policy*" or "*trust*"), below the arrow, the scope of the decision can be given (e.g., whose policy is specified).

For a request, a service first needs to check whether the issuer of the request is granted access in the location policy. The service thus tries to build a chain of certificates from itself to the issuer of the request. Each certificate in the chain needs to give access to the requested location information to the next entity further down the chain. In addition, any constraints listed in the certificate (e.g., time or location based) need to be fulfilled to have the policy check succeed. We elaborate on location policy certificates in Section 5.2.

If a service receives a forwarded request, it also has to check whether the service from which it got the request is trusted. Similar to the location policy check, the

service tries to build a chain of trust certificates from itself to the forwarding service. We discuss trust certificates in Section 5.3.

A certificate can state that a right is not transitive. In such a case, no certificates can follow this certificate in the certificate chain.

5.2 Location Policy Check

For each entity that is permitted to retrieve location information about someone, there has to be a certificate stating what kind of information the entity is allowed to get. (We limit the description of our design to user location policies. Room location policies are dealt with in a similar way.) For example, a certificate could state that Bob can locate Alice at a coarse-grained level. If Bob then tries to locate Alice, a service receiving his location request needs to check the existence and validity of a certificate permitting this access.

A location service does not have to be aware of the identity of the entities that are given access rights. If Bob can prove that he was given access (by presenting his digital certificate), he will be granted access. This solution thus makes dealing with unknown users easy. Solutions proposed earlier (e.g., by Leonhardt and Magee [12]) rely on policy matrices consisting of querying users and users whose location is being queried for and thus rely on the system being aware of the identity of querying users.

In addition, some digital certificates (such as SPKI/SDSI [5]) can not only give rights to single entities, but also to groups of entities. That is, with a single certificate, Alice can, for example, give all her friends access.

Depending on the environment, different entities issue certificates. For the two environments introduced in Section 3, certificates are specified as follows:

Hospital We show some of the certificates issued in the hospital environment: (PL denotes the People Locator service, C the central authority, S the surgery ward, and D_S^C all doctors that are classified by C as belonging to S.)

$$(1) \quad (a) \quad PL \xrightarrow{policy} C \qquad (b) \quad C \xrightarrow[A]{policy} D_S^C \qquad (c) \quad D_S^C \longmapsto B$$

First, the administrator of the People Locator service enables the central authority to decide about all the patients' location policies. Second, the authority gives all doctors in patient A's ward access to her location. Third, the authority certifies that doctor B belongs to the surgery ward. Note that this certificate is a membership certificate (denoted by the special arrow) and does not grant any rights on its own.

If B inquires about A's location, the People Locator service deduces that B has access to A's location since it can combine certificates 1(a), (b) and (c) in a chain of certificates and conclude

$$\text{PL} \xrightarrow[A]{policy} \text{B}.$$

In addition to certificate 1(b), the authority also issues a certificate that gives patient A the right to let additional people locate her. A can do so by issuing corresponding certificates.

University In the university environment, the administrator of the People Locator service gives student or faculty member A access to her location information:

$$PL \xrightarrow[A]{policy} A.$$

A can define her own location policy by issuing additional certificates.

5.3 Service Trust Check

Trust decisions are also stated in digital certificates. For the two environments, we show how the Device Locator service handles trust decisions. Typically, this service is not directly contacted by users, but by other services (e.g., the People Locator service).

Hospital The administrator of the Device Locator service (DL) lets the central authority specify the trusted services used for answering patient queries. The authority states that all services run by the hospital (T_C) are trusted. The authority also states that the People Locator service is in this set. The certificates look as follows:

$$DL \xrightarrow{trust} C \qquad C \xrightarrow{trust} T_C \qquad T_C \longmapsto PL$$

Upon receiving a forwarded user query from the People Locator service, the Device Locator service combines these certificates and concludes that the People Locator service is trusted.

University In the university scenario, users specify their trusted services. The Device Locator service thus gives student A the right to define this set. Assuming A trusts the People Locator service, the certificates look as follows: (T_A denotes the set of services trusted by A.)

$$(2) \quad (a) \quad DL \xrightarrow[A]{trust} A \qquad (b) \quad A \xrightarrow[A]{trust} T_A \qquad (c) \quad T_A \longmapsto PL$$

5.4 Delegation

Entities in our system grant other entities particular kinds of rights. For example, Alice grants Bob access. It is up to Alice to also grant Bob the right to forward the access right to a third entity. If she does so, Alice effectively delegates the right to decide about her location policy to Bob. In the remainder of this paper, we are going to use the term "delegation" when an entity grants access rights to a second entity and it also permits the second entity to grant these rights to a third entity.

Not only the decision about a user's location policy can be delegated to some other entity, but also the location policy (or trust) check and the decision about which organization an entity belongs to.

Location Policy and Trust Checks Each trusted location service has to implement location policy and trust checks. However, to reduce overhead or in case of low processing power, not every service is required to build the potentially long certificate chains itself. It can delegate this task to some other service. For example, the Device Locator service is likely to delegate location policy checking to the People Locator service since the People Locator service needs to build a certificate chain for each request anyway. After validating the chain, the People Locator service issues a new certificate that directly authorizes the location seeker. It gives this certificate to the Device Locator service, which thus does not have to validate the entire chain again.

Organizations If there are lots of services available, it is cumbersome for a user to issue a certificate for each service that she trusts. We assume that trust in a service is closely tied to the entity that administrates this service. For example, a user might trust all the services run by her company. Therefore, we give users the possibility to state in a certificate that they trust all the services in a particular organization. The organization would then generate a certificate for each of the services that it runs. In this situation, a user effectively delegates the decision about which services she trusts to the organization and she relies on the organization to do the right thing. For the university environment, certificate 2(b) could thus look as follows:

$$A \xrightarrow[A]{trust} T_C.$$

5.5 Privacy

Location policy and trust certificates can be stored in a centralized database from which location services or location seekers retrieve them upon a request.

However, location policies contain personal information. Policy makers thus might want to keep these policies private and restrict access to the corresponding certificates. We now discuss two methods to limit exposure of these certificates. The first method keeps policies secret from location seekers, the second one from the location system. Note that a location seeker can obtain some policy information implicitly by issuing queries to the location system. For example, if the location seeker is denied access, it concludes that the policy forbids access. Similarly, by analyzing the returned location information, it might be able to deduce that it is given coarse-grained access only. We elaborate on this problem in Section 7.

For both methods, we assume that each policy maker has a trusted host where it keeps its issued certificates. If the policy maker is a central authority, the trusted host can be the authority itself. If the policy maker is an individual, the trusted host is a designated host. This host has to be available all the time so that it can provide certificates upon a request. An individual has to make a trade-off between its privacy concerns and the convenience provided by its chosen solution. If the individual is willing to trust an organization, it can have the organization run a centralized repository for it. If it is not willing to trust an organization, it needs to set up its own repository.

To prevent policy information from leaking to location seekers, we have the trusted host control access to the certificates stored with it. Only location services entitled by the issuer of a certificate are allowed to access a certificate. This method increases the load on a location service since the service can no longer require location seekers to provide any certificates required for answering the request. Instead it has to retrieve them itself. However, delegation could reduce some of the load on a location service.

To prevent policy information from leaking to the location system, a policy maker has all queries go through its trusted host and it has the trusted host run the access control check. In this solution, certificate chains expressing the location policy are rooted at the trusted host. If the access control check succeeds, the trusted host issues a request to the People Locator service. It also forwards the answer from the service back to the location seeker. Note that the trusted host is the only entity that is given access to the location information offered by a location service. In this solution, a location service does not become aware of the actual location seeker and of membership associations.

A variant of the second method is to implement the People Locator service in a completely distributed way. That is, each policy maker runs its own People Locator service on its trusted computer. (This solution is similar to the one introduced by Spreitzer and Theimer [14].) However, running and administrating a complete People Locator service is a potentially heavyweight operation; it involves contacting other services and processing received location information.

Finally, note that all these methods do not require any changes to the access control mechanism of the location system. They differ only in how the proposed access control mechanism is deployed. Our access control mechanism based on digital certificates thus provides a great degree of flexibility.

5.6 Discussion

Some advantages of our location system are:

No bottleneck. The services in the system run the location policy and trust checks by building chains of certificates. Certificates are signed data structures and do not need to be kept at a centralized node. Therefore, there is no bottleneck through which each request has to go and that has to approve each request. All the services perform access control independently of each other and approve a request only if it is supported by digital certificates.

Support of unknown users. The system does not need to know the identity of location seekers. All the system requires is a digital certificate that grants access to the entity.

Support of group access. With a single certificate, an entire group of entities can be given access to someone's location information.

Transitivity control. SPKI/SDSI certificates allow handing out non-transitive access rights. In a valid certificate chain, all but the last certificate in the chain need to allow transitivity of access rights.

We have been able to use digital certificates for expressing location policy and trust decisions in both of our example environments. Therefore, the mechanism

for controlling access to location information is identical. Similarly, we can use the same tools for building and proving certificate chains in both environments. However, setting up these mechanisms and tools is environment-specific. We now discuss some of the differences.

User interfaces. Most probably, there are going to be different user interfaces in different environments. In the hospital case, there is a strong emphasis on being able to define groups and assigning them access. In the university case, users are more likely to authorize other people directly. Though the actual interfaces might differ, they would both create corresponding location policy (and trust) certificates in the background.

Types of chains. The types of chains checked upon a query also depend on the environment. A service either checks only the user or room policy chain or it checks both chains. For example, when processing a room query about the people in a lecture room in a university, the location system first checks the room policy of the room. If this check succeeds, the system will check the user policy for each of the people in the room. The system returns only location information about people for whom this latter check succeeds.

6 Implementation

6.1 Digital Certificates

To implement the policy and trust decisions, as introduced in Section 5, we use digital certificates and chains of certificates. In particular, we rely on SPKI/SDSI certificates [5]. Authentication and authorization based on these certificates does not rely on the existence of a global naming structure as, for example, the one introduced for ISO's X.509 certificates. Instead, the authentication and authorization step are merged by directly giving a public key access rights in a certificate.

SPKI/SDSI defines a language for giving access rights to public keys in certificates. There are also tools that, based on these certificates, build and prove chain of certificates and decide whether a request should be granted access.

In the example below, we show a SPKI/SDSI certificate in which Alice (`issuer`) gives Bob (`subject`) access to her location information (i.e., $A \xrightarrow[A]{policy} B$ using our notation). The type of access is described after the keyword `tag`. Note that the certificate has to be accompanied by Alice's signature (not shown here).

```
(cert
  (issuer (pub_key:alice))
  (subject (pub_key:bob))
  (propagate)
  (tag (policy alice
          (* set (* prefix world.cmu.wean) world.cmu.doherty.room1234)
          (* set (monday (* range numeric ge #8000# le #1200#))
                 (tuesday (* range numeric ge #1300# le #1400#)))
          coarse-grained)))
```

Alice gives access to Bob's public key (`pub_key:bob`). (`pub_key:bob` is replaced by the actual public key of Bob in the real implementation.) The keyword `propagate` states that Bob is allowed to give the granted right to some other entity by issuing another certificate. If Alice decided not to let Bob forward his access right, she would omit the `propagate` entry. Bob can locate Alice only if she is either in Wean Hall or in Room 1234 in Doherty Hall and on Monday between 8am and 12pm and on Tuesday between 1pm and 2pm. Also, Bob can locate Alice only at coarse-grained granularity.

6.2 Location System

We have implemented a subset of the location system shown in Figure 1. The system consists of the People and Device Locator services and location services that proxy to several calendar systems, that locate devices connecting to CMU's wireless network, and that exploit login information.

We use SSH at the transport layer to achieve mutual authentication of entities and confidentiality and integrity of information. Access control is entirely based on SPKI/SDSI.

Location information and SPKI/SDSI certificates are transmitted between services using the Aura Contextual Service Interface [10], which is a protocol running over HTTP and which exchanges messages encoded in XML. Our SPKI/SDSI implementation has been implemented in Java and is based on previous work [9]. There is an HTML-based front end to the service that lets users specify location policies and conduct searches. The front end makes dealing with certificates transparent to users. Currently, the certificates are stored in a centralized database.

In our system, if Alice directly gives Bob access to her location information in a certificate, it takes about 0.6 sec on a Pentium III/500 to add this certificate to the certificate chain for Alice and to verify the entire chain upon a request.

7 Related Work

Several location systems, all of them based on only one location technology, implemented only within one administrative entity, and/or not addressing the various security issues mentioned in this paper have been proposed [1, 8, 13, 15]. We discuss two notable exceptions:

Spreitzer and Theimer's location system [14] is based on multiple technologies. Each user has her personal agent that gathers location information about her and that implements access control for this information. The authors design the system to work in an environment with different administrative entities, although the actual implementation runs only within a single entity and the authors do not mention how users specify services they trust. A key difference from our system is that the amount of information and the users granted access to room queries is determined by located users, not the "owner" of a room.

Leonhardt and Magee [12] also address security concerns. Since user queries can be implemented by a series of room queries and vice versa, the authors argue that access control for both types of queries needs to be consistent. The authors

propose an extension to the matrix-based access control scheme. We believe that having consistent user and location policies is difficult to achieve when policies are established independently of each other. The authors do not discuss how policies are established in their system.

There has been some previous work about authorizing intermediate services, for example, Howell's quoting gateways [9]. This related work focuses on intermediate services that create new requests upon receiving a request and thus need to be authorized to issue requests. However, in scenarios like our location system where some services only forward requests, this model would give too many capabilities to intermediate services. Using our model of trust, we can avoid this risk and clearly define which services should be given location information and which services should also be able to issue requests.

Kagal et al. [11] propose an extended role-based access control model for ubiquitous computing. The model also supports delegation of access rights. For access control, the proposed system relies on a centralized trusted entity running a Prolog knowledge base. The authors do not examine trust in services that are run by different administrative entities.

The Location Interoperability Forum [7] and the Instant Messaging / Presence Protocol working group [4] discuss privacy requirement for location and presence information, respectively. This work deals only with user queries. It suggests that location seekers should not be able to learn about the content of location policies. The working group proposes that the location system returns false location information instead of denying access to location information. We refrain from implementing such a solution in our system since it erodes trustworthiness into the system. Also, if a location seeker tried to validate the wrong location information, he could still conclude that he was actually denied access. As an alternative to the proposed method, we suggest that policy makers build wrapper services that access the location system and translate "access denied" messages. For example, the wrapper service could return a busy signal when a user temporarily does not want to be disturbed and thus decides not to be locatable.

The Geopriv working group [3] also discusses privacy aspects of user queries. This working group has a strong emphasis on protecting identities, both of location seekers and located entities. We have presented a mechanism for hiding the identity of location seekers in Section 3.5. Protecting the identity of located entities strongly depends on the actual location service. For example, a user can buy a prepaid phone card when using a GPS-enhanced phone and thus hide his identity from the GPS location service.

8 Conclusions

In this paper, we have analyzed the access control requirements of a location system and presented the design of a solution. It relies on several key concepts: services implementing location policy checks, service trust for dealing with different services, and delegation for delegating various decisions to other entities in the system.

We have shown feasibility of our design with an example implementation of it. Our implementation is currently being deployed at Carnegie Mellon.

We have been able to formulate all of our policy and trust decisions using SPKI/SDSI certificates. These certificates provide a high degree of flexibility. A ubiquitous computing environment poses new challenges on access control that cannot be easily satisfied by conventional mechanisms. We believe that due to their flexibility, SPKI/SDSI certificates are a promising approach and deserve further investigation on their usability in such environments. Namely, these certificates could potentially not only protect access to people location information, but also to other kinds of information that is gathered in a similar way as people location information.

References

1. P. Bahl and V. Padmanabhan. RADAR: An In-Building RF-Based User Location and Tracking System. In *Proceedings of Infocom 2000*, pages 775–784, March 2000.
2. M. Blaze, J. Ioannidis, and A. Keromytis. The KeyNote Trust-Management System Version 2. RFC 2704, September 1999.
3. J. Cuellar, J. B. Morris, and D. Mulligan. Geopriv requirements. Internet Draft, June 2002.
4. M. Day, S. Aggarwal, G. Mohr, and J. Vincent. Instant Messaging / Presence Protocol Requirements. RFC 2779, February 2000.
5. C. Ellison, B. Frantz, B. Lampson, R. Rivest, B. Thomas, and T. Ylonen. SPKI Certificate Theory. RFC 2693, September 1999.
6. D. Garlan, D. Siewiorek, A. Smailagic, and P. Steenkiste. Project Aura: Towards Distraction-Free Pervasive Computing. *IEEE Pervasive Computing*, 1(2):22–31, April-June 2002.
7. D. Greening. Location Privacy. http://www.locationforum.org/About_LIF/Documents/Location-privacy.ppt.
8. A. Harter and A. Hopper. A Distributed Location System for the Active Office. *IEEE Network*, 8(1), January 1994.
9. J. Howell and D. Kotz. End-to-end authorization. In *Proceedings of OSDI 2000*, pages 151–164, October 2000.
10. G. Judd and P. Steenkiste. Providing Contextual Information to Ubiquitous Computing Applications. Under submission.
11. L. Kagal, J. Undercoffer, F. Perich, A. Joshi, and T. Finin. A Security Architecture Based On Trust Management for Pervasive Computing Systems. In *Proceedings of Grace Hopper Celebration of Women in Computing 2002*, October 2002.
12. U. Leonhardt and J. Magee. Security Considerations for a Distributed Location Service. *Journal of Network and Systems Management*, 6(1):51–70, March 1998.
13. N.B. Priyantha, A. Chakraborty, and H. Balakrishnan. The Cricket Location-Support System. In *Proceedings of Mobicom 2000*, August 2000.
14. M. Spreitzer and M. Theimer. Providing Location Information in a Ubiquitous Computing Environment. In *Proceedings of SIGOPS '93*, pages 270–283, Dec 1993.
15. A. Ward, A. Jones, and A. Hopper. A New Location Technique for the Active Office. *IEEE Personal Communications*, 4(5):42–47, October 1997.