Designing for Privacy in Ubiquitous Computing Environments

Yitao Duan and John Canny

Computer Science Division
University of California, Berkeley
Berkeley, CA 94720, USA
{duan, jfc}@cs.berkeley.edu

Abstract. In an Ubiquitous Computing environment, sensors are actively collecting data, much of which can be very sensitive. Data will often be streaming at high rates (video and audio) and it must be dealt with in real-time. Protecting the privacy of users is of central importance. Effective solutions for controlling access to data in ubicomp settings remain to be developed. Dealing with these issues will be a central challenge for ubicomp for some time to come. Here we propose some simple design principles which address several of these issues. We illustrate them through the design of a smart room capture system we are building at Berkeley. The main design principle is "data discretion": users should have access and control of data about them, and should be able to determine how it is used. In our implementation, the data discretion principle is enforced with cryptographic techniques. We show how data discretion supports both personal *and* collaborative uses. Unlike traditional ACL based access control systems, our scheme essentially embeds access rights of legitimate users in the data. We have implemented a prototype system in a Smart Room at Berkeley equipped with several cameras, and we give data throughput rates under various degrees of protection. Finally we describe ongoing work toward a trustworthy ubicomp environment whose discretion is realistically checkable.

1 Introduction

An ubicomp environment is typically envisioned as a space populated with large number of invisible, collaborating computers, sensors and actuators interacting with user-worn gadgets. Data about individuals who are in the environment is constantly being generated, transmitted and stored. Much of the data can be quite sensitive. Protecting private data is a major concern for users. There are a few challenges that make data security in Ubicomp settings different from other system protection:

1. The environment is often unfamiliar to the users. They will not have a trust relationship with the owners of the environment as they might with their local system administrator appropriate for handling their private information.

2. Data are often generated dynamically and streaming at high rates (video and audio) and must be processed in real-time.
3. Users' access rights change dynamically with respect to their relationship with the mechanisms by which data are generated.
4. Data usage often involves sharing among a group of people [1]. Any protection scheme must allow efficient sharing among legitimate users.

Unless there is a clearly stated protection policy and scheme in place, together with provisions for convincing verification, users will not be comfortable trusting the infrastructure with their data. And this will hinder the acceptance and limit the usefulness of such systems.

For users' digital data, privacy protection is relatively easy. Clients can encrypt files by themselves and upload the encrypted files to the server. In ubicomp settings, the data are often generated by another party. In this paper we are concerned with protection of user data that are generated by the infrastructure. This case is tricky because the users have to trust the system to some degree (e.g., the fidelity of the sensor data) since they rely on the system to perform certain operations (e.g., data generation, transmission, and encryption). We do not want to encourage "blind trust" from the user. Rather, we would like to design the system in such a way that it promotes user trust and confidence. We consider a typical ubicomp environment, in this case a smart room augmented with a variety of sensors. The issues we address in this paper are:

1. Protection of the user data generated and maintained by the environment.
2. Privacy of individuals who use the environment.
3. Ability of legitimate users to make use of data recorded in the environment.
4. Dealing with high-speed streams of data.
5. Trustworthiness of the environment and users who have access to the data it captures. (this is work in progress)

We propose an approach to protect user data in a dynamic and ad hoc environment. Our scheme makes use of both secret key and public key cryptography and can achieve efficiency high enough to be implemented using today's commodity hardware and software, and deal with steams of audio and video data. It is by no means a complete solution to data security in ubicomp environments, rather it offers a simpler and more efficient alternative to ACLs and represents a first step toward making a ubicomp environment trustworthy.

The rest of the paper is organized as follows. In Section 2 we survey some recent books and papers on privacy that have guided our design. Section 3 presents related work which motivates our privacy principle. In section 4 we describe personal and collaborative applications of smart spaces which leverage our privacy system. In section 5 we describe our testbed smart room. In section 6 we present the data discretion principle and describe the protection scheme that enforce this principle and how it is applied to our smart room. Section 7 gives performance evaluation of our prototype system. Section 8 presents extensions and possible techniques for improving and validating privacy compliance in ubicomp environments. Finally section 9 summarizes and describes directions for future research.

2 Privacy Principles

Privacy is an enormous subject, but there are three perspectives that we feel are essential to designs that target a broad range of ubicomp applications. They are legal, economic, and social/critical perspectives. Our summaries are extremely brief, but nevertheless cover principles that are extremely important for ubicomp design.

2.1 Legal Perspectives

The legal profession has of course a great deal to say on privacy. An influential recent work is Alderman and Kennedy's "The Right to Privacy" [2]. The legal perspective contributes many key insights:

1. Privacy issues normally arise as a tension between two parties: one party who wants information about another, and the second party who faces some risk from the disclosure of this information.
2. The identity of the *seeker* of information leads to very different situations and legal precedents, e.g. individual vs. the press, vs. law enforcement, vs. employer...
3. Privacy problems are most acute when there is an imbalance of power between the party seeking the information and the other party.
4. Loss of privacy often leads to real harm: loss of job, loss of rights, loss of quality of life.

In the legal arena, the tension between the two parties is played out in the courts as litigation. Ubiquitous computing is creating a new set of situations where privacy difficulties arise, but the set of key actors is the same. Both technology and law will contribute to the development of privacy-respecting environments in future. Neither can "solve" the privacy problem alone. The law creates the framework to which technology design must comply. There are many gaps in laws applicable to ubicomp environments and electronic environments generally. But these gaps are being filled. The new situations that ubicomp creates are less novel than might appear at first sight. In particular, all the traditional tensions are present such individual vs. employer, individual vs. press etc. We believe that the best strategy for technical design is to look to the existing case law for the appropriate set of situations. This is the most likely path that the courts will follow, at least in the long run. We also believe that it is important to study broad legal perspectives on privacy, not just emerging cyberlaw principles which apply primarily to the internet. Ubicomp applies to many everyday situations in the home and workplace and reconstructs many traditional privacy conflicts.

2.2 Economic Perspectives

A recent workshop held at Berkeley [3] studied the economic aspects of security and privacy. Economic analysis has mostly focusses on markets and online

commerce. Another of the major issues discussed was incentives for or against privacy technologies. Unfortunately, market forces work against customer privacy. A very thoughtful paper by Feigenbaum et al. [4] discussed this problem in some detail. In spite of the existence of technical solutions to protect privacy in electronic transactions, such solutions have not been adopted. At least three influences work against adoption of privacy technologies: (i) companies benefit from more customer information for both marketing and price-setting, and so are incentivized to minimize privacy (ii) there is almost always a power imbalance between online vendors and individual customers, so customers have no countervailing influence and (iii) network effects work against privacy technologies until they are widely adopted. The adoption problem for privacy technologies is particularly acute. Privacy is not guaranteed until *all the agents* involved in a transaction, banks, vendors, shippers or intermediaries, support the technology.

Odlyzko [5] goes further to argue that vendors often seek customer information to support *price discrimination*. The availability of cheap customer information in electronic settings means that vendors are incentivized to both collect this information, and to expand their price discrimination. Price discrimination is desirable from an economic perspective, because it increases efficiency. But it is often disfavored by consumers, who may regard it as unfair. This tension remains difficult to resolve, once again because of the power imbalance between vendors and consumers.

In [6], Acquisti discusses ways that some of these problems might still be addressed using economic methods. He proposes incentive-compatible contracts as a way to induce *groups* of agents to deploy privacy-enhancing technologies when no single agent has such incentive. Secondly, he proposes economic analysis of information exchange to discover favorable flows, followed by legal and technical means to support those flows.

2.3 Social Science Perspectives

A recent Ubicomp workshop co-organized by one of the authors [7] brought together technologists, social scientists and lawyers doing research on privacy. David Philips' paper at the workshop [8] rejected the of notion of privacy as a fixed and one-dimensional aspect of the individual and situation. Rather, he underlined the importance of *management of identity* in privacy discussions. Identity is not a fixed property of the individual either. All of us manage distinct identities; as worker, as homemaker, friend, parent, etc. Privacy involves the management of information boundaries between these social spaces. But it should not be a defensive posture biased toward minimum disclosure. Full participation in social life requires free exchange of information among individuals in various social spheres. Nor are the boundaries fixed. We negotiate and renegotiate them as part of social exchange and trust-building.

This perspective argues for system designs that support pseudonymity - users can maintain several distinct identities corresponding to the social spheres they participate in. It also argues for tools to support the creation and maintenance of communities.

Recently, Dourish and Palen gave a treatise on privacy and HCI that built on a long history of social psychology and critical perspectives [9], in particular the work of the social psychologist Irwin Altman [10, 11]. They argue that privacy is a boundary regulation problem, conditioned by experience and expectation. The boundary has three aspects: disclosure, identity and temporality. They also argue for "genres of disclosure" which are patterns of disclosure combined with social conventions for use and expected responses. Both Dourish/Palen and Philips emphasize the dynamic negotiation of privacy boundaries. They also both endorse the notion of desirable disclosure/publicity for full participation in social life, so that naively "maximizing privacy" cannot be a design goal.

2.4 Summary of the Three Perspectives

The social, legal and economic perspectives are strongly complementary. Economic perspectives typically consider markets or other situations where some form of exchange is evident, e.g. e-commerce, services, and some well-defined social exchanges. Incentives are readily evident, and agent behavior is assumed to be rational and predictable. The situations may be either symmetric or asymmetric, caused by imbalance in either information or power. Some situations lead to inefficient markets, others to socially undesirable outcomes. Those latter cases are where the law steps in. Most of the legal situations we listed involve power imbalance and without legal restrictions, the weaker actor easily becomes a victim. These situations are well-represented in the online world: employer/employee monitoring, surveillance by law enforcement, tracking by online vendors... The social perspectives cover much of what remains: privacy in social situations at home and in the workplace. While not explicitly stated in the social sciences works, the situations are generally symmetric which favors rich and complex negotiation between the actors. The incentives and risks are less acute (certainly less obvious), so economics and law generally do not intervene. Outcomes are not easily predictable, and the "design problem" cannot be about fostering economically or socially desirable outcomes. Rather, the designer should support the rich privacy management practices and community memberships that social actors seek to pursue.

All of these perspectives reiterate negotiation of privacy boundaries. Such negotation is dynamic but not arbitrary. In market situations, it will be constrained by pareto-optimal strategies for each actor. In many asymmetric situations, it will be constrained by applicable law and sometimes by contract. In social situations, it will follow "genres of disclosure" with their associated conventions and patterns of behavior. So privacy-aware technologies should reify these constraints and conventions as constraints and/or templates, while providing users with appropriate choices to fully support the negotiation that fits the situation.

2.5 Other Formulations

In [12], Marc Langheinrich laid out some principles for design of privacy-respecting ubicomp systems. Langheinrich stresses the importance of including privacy con-

siderations in the early stage of system design process. He develops six principles for guiding privacy-aware ubiquitous system design. The six principles are:

1. Notice: users should always be aware of what data is being collected.
2. Choice and Consent: users should be able to choose whether it is used.
3. Anonymity, Pseudonymity: should apply when identity is not needed.
4. Meeting Expectations: systems should mimic real-world norms.
5. Security: different amounts of protection depending on the situation.
6. Access and Recourse: users should have access to data about them.

Langheinrich's principles provide very specific guidance for our work. Later, we describe our design approach that supports several of the principles directly, and makes it easy to build systems which support all of them.

3 Related Work

Several recent projects seek to protect user anonymity in communication and data capturing [13, 14] They basically follow the rules and practice established in Internet Privacy [15, 16], which focus on obscuring user's IP address, and extend them into ubiquitous computing context. The Mist Project at UIUC strives to anonymize users' communication in ubicomp environments [14]. It utilizes a hierarchy of "Mist Routers" that perform "handle-based routing" to preserve privacy and hide information about the source and destination. Users will expose their identities to part of the system, but their locations will be concealed. The EuroPARC's RAVE [17] presents a networked node model where interactions are defined by connections established between nodes. It emphasizes two principles in preserving privacy: control and feedback. The former empowers users to stipulate what information they project and who can access it while the latter allows the users to be informed of the capturing and usage of their information. Privacy preserving is achieved by controlling connection capabilities. While this is a reasonable model for protecting transient information, it is not clear how permanent data is protected. Also it represents an "untrustworthy" environment for visitors who have no knowledge of how it works. Based on recent economic market theories (Ackerloff's asymmetric information theory), Jiang and others proposed a principle of minimum information asymmetry for ubicomp environments [18].

4 Applications

4.1 Personal History

A number of researchers have explored the idea of lifetime personal histories collected electronically. Gelernter's "Mirror Worlds" is an early example [19]. More recently, wearable computing researchers Steve Mann [20] and Bradley Rhodes [21] (the "remembrance agent") built systems to record and retrieve

their daily histories. Other current explorations of lifetime history include "Stuff I've seen" and Cyberall (resp. Susan Dumais, Gordon Bell, Microsoft), and the "Personal Server" (Roy Want, Intel).

A wearable computer provides one source of lifetime history. Wearables may or may not be adopted by most users. Sensor-equipped spaces provide an alternative. Data recorded in the space can be sent to all the users who are in the space at the time. In this way, the room becomes a "virtual personal recorder". As the user moves, they carry an ID tag, which might include their encryption key and the URI of a server to which the space should send their recorded data. If they enter another sensor-equipped space, it takes over the task of recording and shipping data to their server. Advantages of this approach are that the room may include more sensors, e.g. cameras in many positions, that the data may be streamed over a high-bandwidth fixed link to the users server, and that this server can have higher storage capacity (and backup) than a wearable.

We are implementing a simple video recording service along these lines in a smart room. We use RFID tags to determine who is in the room, although because of memory limits we use a lookup table to map from the tag number to an encryption key and target URI.

4.2 Collaborative Applications

Surprisingly, individual private data collection also allows some important *collaborative* applications that support privacy. In other words, users can share their information without losing privacy. We explain how shortly.

Collaboration is a major challenge to privacy. By its nature, collaboration involves exchange of information between collaborators. It may also involve automation - the best example of which is collaborative filtering [22]. We also have several ongoing projects which make use of history data from several individuals to support other types of collaboration. In particular, we are using personal history data to compute collaborative "activities". Activities are shared patterns of communication, document and web access which are mined using a clustering algorithm from histories of the participants.

Normally, collaborative computation would require user data to be stored and processed on a single server computer. We became very concerned about the privacy risks involved with doing this. In ubicomp settings, users generally do not have strong trust relationships with the individuals who manage the hardware. This lack of strong trust applies also between individuals and the owners of community-based services. The problem exists already on the web - many sites monitor users' progress around their sites, and their electronic purchases. The problem is at least limited to the scope of a particular web site. But ubicomp provides much richer and more invasive data.

To deal with this problem we explored encrypted computation. The idea is that raw user data should remain accessible only to the user in question. When collaborative computation is needed, it should be done only on encrypted data, thereby protecting individual privacy to the maximum extent possible. This

approach is indeed feasible, and in [1] we showed that is practical for interesting collaborative tasks. That paper described a new collaborative filtering (CF) algorithm based on encrypted computation that matched the accuracy of the best-performing algorithms at that time. We also showed that basic clustering (based on SVD) is possible on encrypted data.

We continued to work on collaborative filtering, and in [23] we showed that encrypted computation need not penalize either performance or accuracy. A second collaborative filtering algorithm, based on sparse factor analysis (SFA), was introduced in [23]. Not only does that algorithm support encrypted computation, but through several experiments we showed that it is the most accurate CF algorithm to date and one of the most efficient. This method also supports *community creation and maintenance* of collaborative groups, and so addresses the needs for social participation emphasized by the social sciences perspectives we gave earlier.

A very interesting application of CF in ubicomp is to location data, and we recently received an NSF grant to support this exploration. For instance, by tracking their location and aggregating with others, users can obtain recommendations about restaurants, shops, places to see and things to do. But gathering such information creates great risks to privacy. The general thread of our work is to explore cryptographic and AI techniques to compute from user data only the information needed for a particular task, and to protect the rest of the data.

5 Smart Room Testbed

We have begun building a smart room equipped with sensing and able to record user activity. The room is a small meeting room with a conventional whiteboard, a Smart Technologies Smartboard^{TM} with projector, four video cameras, and a Philips I-CODE^{TM} RFID tag reader. There is a single entrance to the room, and the tag reader antenna is mounted around this entrance so that it can record entry and exit by any user carrying an I-CODE tag (it also provides security and can record any piece of tagged equipment that might be removed from the room). The tags provide reliable recognition of users entering the room, although users without tags will not be noticed by the tag reader. It can of course be fooled by tags which move through the doorway when not attached to their users. The cameras capture images from the room continuously, and send them to a data server. Another computer drives the Smart-board, and this computer runs a logging program that records all significant user activity using the board. The smart room is representative of typical ubicomp environments and serves as a good testbed for our privacy principles. The images and log data represent dynamically generated user data that we seek to protect. We will provide a simple playback application that explores the privacy protections. Fig. 1 shows a phote of the smart room testbed. Section 7 will give more details about the prototype system and the experiments we conducted on the smart room testbed.

Fig. 1. Smart room testbed

6 System Design: the Data Discretion Principle

If we take Langheinrich's principles 4 and 6 together, meeting expectations and access and recourse, we derive a principle we call data discretion:

Data Discretion: Users should always have access to, and control of (recorded or live) information that would be available to them in "real-world" situations. They should not have direct access in other situations.

So for instance, users should have access to information recorded in a smart room that while they were in the room. They should not have access to information recorded in that room while they were not present. They should also have control of this information, and be able to use it however they please.

By making access to the user a requirement, we can make it easier to satisfy the other privacy principles 1, 2, 3 and 5. That is because we can require all accesses to user data to be routed to the user, and therefore we involve the user in all decisions about use of their data. In our current smart room, we are developing applications to personal recording and to (collaborative) activity monitoring.

There is some subtlety in making the discretion principle work. Particularly in a smart room setting, we would like to keep the identity of the people who were in the room a secret from those who were not in the room at that time. This means that users who were not in the room at the time should not have access to information recorded in the room at that time, nor should they be able to find out who does have access, because that is equivalent to knowing who was in the room. This rules out many access control methods, which expose the set of users who have access to files to system administrators or others who can penetrate the operating system. We prefer methods based on cryptography which make it impossible to gain access to sensitive information even with control of the machine and operating system.

6.1 Basics

Our method is based on hybrid secret-key and public-key cryptography. We assume that all data is stored in files that represent short time intervals. For instance, each video file contains a short sequences of images of one second or so, or logs from the Smartboard in files that span about a second, etc. Every user file is encrypted with a unique, randomly generated secret key. This secret key is then encrypted by the public keys of the file owners. We will describe the scheme in details in section 6.3.

A user joins our system by registering his or her public key [1] and choosing a recognition tag that the system can detect once it is within the vicinity of a sensor. The recognition tag is used by the system to detect user's presence so that it can retrieve related information (e.g., user's public key). We define recognition tag in an abstract sense in that it does not have to be a physical tag and can be implemented with any technology (e.g., RFID tag or biometric). The user is assumed to wear the recognition tag.

Note this approach is different from access control lists (ACL) that are typically used to protect user data whose owner is predeterminable. Our system only needs to know the association between user data and its access key(s) and can function perfectly well without knowing identification of the holder of a particular tag. No attempt is made to connect a tag with a user's real-world identity, or even his or her system-wide electronic identity. User anonymity is therefore protected and users can maintain multiple pseudo-identities with multiple tags. This of course is limited by their visible presence to other users in the space who may know them by name.

6.2 Establishing Information Ownership

The first step in protecting user data is to ascertain the natural owners of the information. In a dynamic environment such as ubiquitous computing context, this has to be done while the information is being generated. We assume that the system has the ability to detect a user's recognition tag when they are present in the environment. Other identification technologies (e.g., face and fingerprint recognition) are expanding rapidly, and there already exist several that are mature enough to offer non-intrusive detection at reasonable cost. However, some of these technologies rely on recognition of information that are unique to a user (e.g., fingerprint) and are not suitable for privacy. RFID tags are good technologies from a privacy perspective because they have no unique attributes of particular users. The decoupling of user's identity and their recognition tag effectively enables us to guarantee anonymity. A user can also have several pseudonyms via multiple tags. Biometric data is more sensitive because if it were extracted from the system, it would provide outsiders with cues to this individual's identity.

[1] We hope to eliminate this step in future. We would prefer the user's public key to be carried in their tag, along with (possible) a URL to which their data should be sent. Our current tags do not have enough writeable bits

RFID tags also allow users who are not interested in using the system a way to opt out by not carrying a tag.

6.3 Encryption Scheme

The basic output of the system is a sequence of files and key-tuples. File F_i is the data logged at time i. Associated with F_i is a tuple of user keys $(k_{i1}, ..., k_{in})$, denoted as *key set*, which determine who has access to this file. The length of the tuple of user keys, n, is always fixed for that environment. The value of n is larger than the maximum number of users who could plausibly be in the environment at the same time. Using a fixed number of keys is necessary if we would like to hide the number of users who are actually in the room at a given time. This many keys may seem extravagant, but in our smart room application, the key set is much smaller than its data file.

Encryption on user data is performed in the following steps:

1. The system performs appropriate key generation algorithm such as ANSI X9.17 [24] and obtains a sequence of pseudorandom keys $d_1, d_2, ...$ One key, denoted d_i, will be selected from the sequence to be used to encrypt file Private key encryption is used for efficiency. Each key will be used only once and the system will perform key generation periodically when the pool of keys is exhausted.
2. d_i is then encrypted with the public keys of the m people in the room (determined by recognition tag reader). These encrypted keys are placed in m locations among the n key positions in the key set, in pseudo-random fashion. The other $n - m$ key positions are filled with random numbers.
3. Users who were in the room can recover the keys and review the video while they were in the room.

Although there are n keys associated with each file, it is not necessary to search through them all in steps 2 and 3. We use a family of n hash functions $h_1, ..., h_n$ to perform a pseudo-random search. At step 2, the system places user j's key in the position specified by one of the hash functions applied to the encrypted image and the user's public key. The first hash function which hashes to a free key location is used. If we assume that n is at least twice the maximum number of people in the room at one time, then at least half these locations will be free, and on average only $\log_2 m$ such steps will be needed to find a free location. The same method is used to retrieve the key at step 3. This scheme is illustrated in Fig. 2.

6.4 Master Key Sharing

It is sometimes necessary for a few privileged parties, e.g., police, to access data stored in the system. However, it is not desirable that a single party be granted full access right to the data since there is a danger of a malicious power party misusing his privilege and compromising users' privacy. Our solution to this

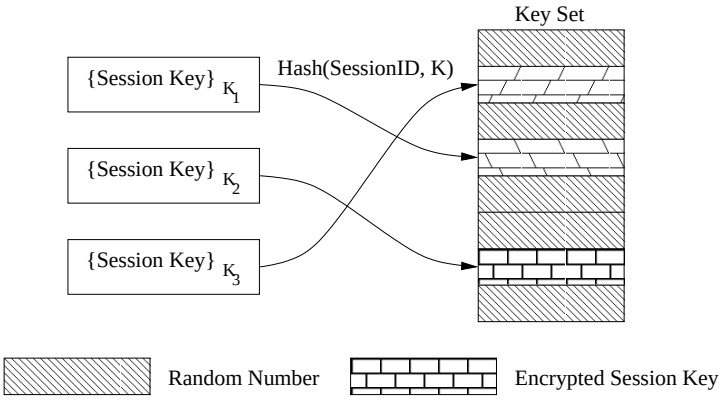

Fig. 2. Key set construction

dilemma is a shared master key and threshold decryption scheme. The master key consists of a globally known El-Gamal public key and a matching private key that is not held by any single party but instead secrete shared among a number of authorized "masters". Masters would normally be distributed to people like the local police department, the building security manager, the corporate president or safety officer, etc. Threshold decryption allows a subset of those individuals (say any two of them) to retrieve data for safety or law enforcement reasons. But it avoids the risks of single individuals accessing data inappropriately.

Each file's encryption/decryption key d_i is encrypted with the public master key. A group of masters whose number must exceed a pre-specified threshold can collaborate to retrieve d_i and access the data[2]. Pedersen's key generation protocol [25] or its variants/enhancements [26, 27] can be used to securely generate the public key and distribute the secret shares of the private key among participants

6.5 System Architecture

Putting together all the functionalities described above, our prototype system is organized as shown in Fig. 3 and consists of three basic components: *Data Generation*, *Data Storage*, and *Data Access*. Data Generation is the data source that is administrated by the system (as opposite to the users). Besides generating data, it is also responsible for user detection which is essential for establishing the ownership of the data being produced. Data Storage is charged with data encryption and associating users access rights. Data Access is the part that services encrypted data to clients. Since our scheme embeds users' access rights in the encrypted data, this part will do minimal access control and only services users who have registered their public keys with the system.

[2] Note that since the encryption keys are one-time only and only valid for one session, it is safe to reveal it to the collaborating "masters".

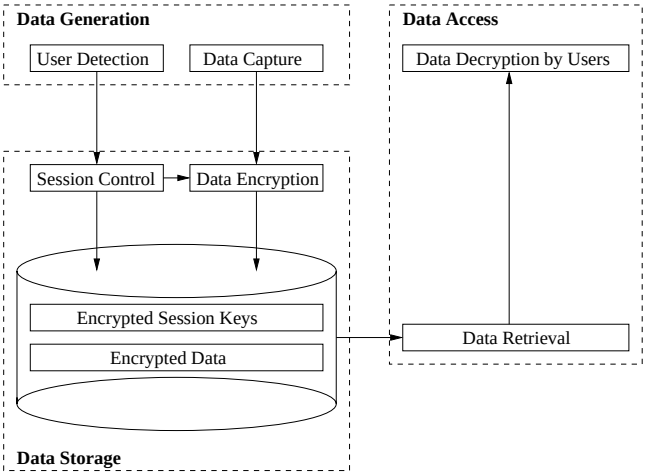

Fig. 3. System architecture

7 Performance Evaluation

We have implemented a prototype system on the smart room testbed to carry out our protection scheme. Our smart room has four cameras as well as one tag reader. Images are typically captured at several images per second.

Our prototype system consists of the following four pieces of software:

1. An FTP server that receives image data from the four video cameras and saves them to disk.
2. A data encryption process that constantly checks for new data on disk and encrypts it.
3. A user data server that serves the encrypted data to users upon request.
4. A proof-of-concept client program that continuously requests the most update-to-date data from the user data server, tries to decrypt them on the fly, and displays the recovered images on the screen (if the decryption is successful); thus if the user has legitimate access to the data, he should be able to see a smooth video replay of what's being captured in the room.

Conceptually the first two programs should be merged into one; we did not implement this due to time constraint. Instead, we simulated the effects of one single integrated data receiving/encryption server by deleting the plaintext files after we have generated the encrypted version. Nevertheless, we believe that an integrated server is more in line with our security principles and will be implemented in future versions of our system. We used Crypto++ Library^{TM} 5.0 [28], an open source crypto library written in C++, for our encryption and decryption functions.

The encryption server is the crucial part of the system and will be discussed in detail below. Logically it consists of two parallel modules: 1) session control

and 2) data encryption. The session control module, a.k.a. the session manager, monitors the users (and their public keys) in the current session and generates session keys. It is notified of user arrivals and departures by the RFID tag reader. A session is defined as a short duration of activities with a fixed set of users. Users' access rights to the data generated at that time remain invariant during one session but would change across sessions. Thus whenever a user enters or exits the room, or whenever a periodic timer expires, the session manager destroys the old session and its associated state, and creates a new session and a new random session key.

The data encryption module monitors the arrival of new image data from the cameras (by periodically checking for new files in a particular file system directory in our case), and encrypts those data with the current session key. We chose Triple-DES with 192-bit keys as our data encryption algorithm. In addition, the module encrypts the session key, which is first prepended with a watchword, with the public key of each user in the current session. We used the RSA public key encryption algorithm for this purpose. The encrypted keys are hidden in the fixed-size key set as described in section 6 and stored together with data.

We have run several tests on our system. One involves two users simultaneously requesting the latest video captures while they are moving in and out of the room randomly. The results are encouraging. Our system can detect the changes of their presence on time and reflect them with the changes of access rights. To determine the throughput of our system, we feed the system with a stream of files of fixed sizes arriving at high speed and measure the time it takes to process them. The experiments were run on a PIII 900MHz machine running Linux 2.4.18 Kernel. The code was complied with gcc3.2. We did not account for the cost of generating and encrypting session keys in our experiments because these operations are performed only once per session and are dominated by the cost of encrypting the data files. Fig. 4 shows the system throughput with different image file size. As our processing time includes two disk I/Os (one read and one write) as well as the encryption, the throughput improves as the file size increases. With a 1MB file size, the system can achieve a throughput of 2.07MBps while with 8KB file size the throughput is about 1.66MBps. Assuming a capture rate of 20 files per second, our system can support up to 10 cameras. These are not optimal figures, although more than enough for our current needs, but there is plenty of room for improvement. First, the encryption algorithm we use, Triple-DES, is very strong and expensive. Other secret key encryption algorithms exist that can yield much higher throughput. As documented in the Crypto++ 5.0 Benchmarks (http://www.eskimo.com/ weidai/benchmarks.html), DES yields almost 3 times throughput as Triple-DES, Blowfish 4 times, Twofish 6 times, and ARC4 more than 13 times. In cases where the strong property of Triple-DES is not required, these algorithms can be used for higher throughput. Second, we expect the throughput to go up with an integrated receiving/encryption server, as we can eliminate one disk I/O from the critical path. Third, code optimization has yet to be done.

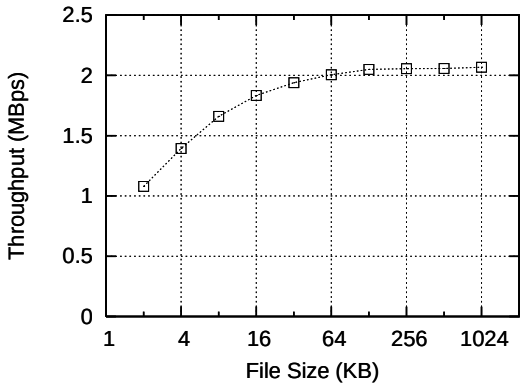

Fig. 4. System throughput

8 Toward Trustworthy Environments

Ubiquitous computing systems, ours and others, suffer from the trust problems we described at the begining of the paper. In future, users will move through a procession of sensor-equipped spaces, all of which can record their actions. Some of the owners of those spaces will be incentivized to collect and use that data. But the user about whom the data is recorded, who we argued is the rightful owner of the data, may not wish this to happen (recall Langheinrich's principle 1 against covert monitoring). The law will almost certainly have a say in this state of affairs. But laws have no potency unless they can be enforced, which means unauthorized monitoring must be detectable at reasonable cost. Bits flowing around the internet can be extremely difficult to trace, so this is certainly a challenge. It creates a technical problem: How do we monitor and verify that a smart room or space is transmitting only the data it is supposed to?

We are approaching this problem from the framework of trusted computing. That is, we assume that most of the hardware in the environment is untrusted, but that a small and inexpensive trusted device (a tamper-proof device like a smart card, but more powerful) is incorporated into the infrastructure in a way that it can verify that the system is satisfying particular constraints. This device would need to be inserted and inspected as part of certification of the space, much like a GFCI receptacle (ground-fault detectors required near water). In our case, the constraints are that the system should send only messages encrypted with keys of users who are in the room, and that the system should not leak information in the messages encrypted with authorized keys. The first guarantee is easily given with ZKP (Zero-Knowledge Proof) techniques [29]. The second (leakage) is quite subtle and is the subject of our ongoing work. In both cases though, the system should obey a key principle, described next.

8.1 Data Transparency Principle

Paradoxically, the first step in making an ubicomp environment trustworthy is to make sure all the data flowing out of it is encrypted, but can be seen by other users or by the inspection system (which is trusted infrastructure).

Data Transparency: Encrypted data recorded or transmitted by a ubicomp system should be easily observable. Where possible, the data itself should demonstrate compliance with stated principles. The information protections provided by well-implemented cryptography are much more reliable than access restrictions using the operating system or network routing. Once data *is* encrypted, ZKP techniques allow us to prove things about it without disclosing any new information about the data. In particular, we can show that it is encrypted with particular keys. And we can attempt to show the absence of leakage.

8.2 Verification Mechanisms

How verification works is the subject of our current research. We already noted the value of ZKPs for this step. ZKPs allow an agent A to prove to an agent B that A has information that would be very hard for B to compute, without disclosing that information ([29]). More concretely in [1], we used ZKPs to prove that private user data was validly generated without disclosing the data. The challenge with the leakage property is that there are many ways to leak information. Each leakage mechanism defines a property of the encrypted data. So proving non-leakage appears to be proving that the data does *not* have an open set of properties. This would not be feasible, but we believe there is another approach that is. The general approach is to force the system to first fix its data by bit-commitment, and then encrypt using a security parameter created by the (trusted) verifier. That means that the system has no way to anticipate what the encrypted data will look like, and it will appear highly random.

This approach has two drawbacks which we would like to remove. The first is that the verifier requires a private connection with the system (otherwise the system could leak information through its bit-commitment). Either a second network address must be used, or the verifier must be placed "inline" in the network path. Both of these have performance implications. The second problem is that the system and verifier communicate interactively. There is certainly a performance penalty with that configuration, and several extra points of failure. It would be better if verification could be non-interactive. In that case, the verifier could run at its own pace, recording data or even using recordings by the normal users of the system at some later time.

9 Conclusions

As many researchers in Ubiquitous Computing have noted, in order for Ubiquitous Computing to be really beneficial and socially acceptable, user privacy has

to be considered carefully at early stage of system design. In this paper, we argue that the essence of preserving user privacy is protecting user data and propose two design principles. The "data discretion" principle stipulates that access to information stored in a system should only be granted to individuals who would have access to the data in the "real-world". Explicit notion of ownership should be established as the information is generated to determine access right. The "data transparency" principle states that, rather than trying to enhance privacy by hiding the existence of information or communication, a system should rely on well-implemented cryptography for data protection and make the recording and transmitting of encrypted data observable. Only when the usage of data is made open can the system perform effective monitoring to enforce compliance with privacy policies. We consider this to be a very important step toward building trustworthy environment.

References

1. Canny, J.: Collaborative filtering with privacy. In: IEEE Symposium on Security and Privacy, Oakland, CA (2002) 45–57
2. Alderman, E., Kennedy, C.: The Right to Privacy. DIANE Publishing Co. (1995)
3. WEIS: Workshop on Economics and Information Security, Berkeley, CA, WEIS (2002)
4. Feigenbaum, J., Nisan, N., Ramachandran, V., Sami, R., Shenke, S.: Agents' privacy in distributed algorithmic mechanisms. In: Workshop on Economics and Information Securit, Berkeley, CA (2002)
5. Odlyzko, A.: Privacy, economics, and price discrimination on the internet. In: Workshop on Economics and Information Securit, Berkeley, CA (2002)
6. Acquisti, A.: Security of personal information and privacy: Economic incentives and technological solutions. In: Workshop on Economics and Information Securit, Berkeley, CA (2002)
7. Privacy In Ubicomp'2002: Workshop on Socially-informed Design of Privacy-enhancing Solutions in Ubiquitous Computing, Privacy In Ubicomp'2002 (2002)
8. Phillips, D.J.: Context, identity, and privacy in ubiquitous computing environments. In: Ubicomp 2002 Workshop on Socially-informed Design of Privacy-enhancing Solutions in Ubiquitous Computing, Goteborg, Sweden (2002)
9. Palen, L., Dourish, P.: Unpacking "privacy" for a networked world. In: Proceedings of the ACM Conference on Human Factors in Computing Systems CHI 2003, Fort Lauderdale, FL, ACM (2003)
10. Altman, E.: The Environment and Social Behavior. Brooks/Cole Pub. Co. In, California (1975)
11. Altman, E.: Privacy regulation: Culturally universal or culturally specific? Journal of Social Issues **33** (1977) 66–84
12. Langheinrich, M.: Privacy by design – principles of privacy-aware ubiquitous systems. In Abowd, G., Brumitt, B., Shafer, S., eds.: Proceedings of Ubicomp 2001. Volume 2201 of Lecture Notes in Computer Science., Springer (2001) 273–291
13. Abowd, G.D., Mynatt, E.D.: Charting past, present, and future research in ubiquitous computing. ACM Transactions on Computer-Human Interaction **7** (2000) 29–58

14. Al-Muhtadi, J., Campbell, R., Kapadia, A., Mickunas, D., Yi, S.: Routing through the mist: Privacy preserving communication in ubiquitous computing environments. In: International Conference of Distributed Computing Systems (ICDCS 2002), Vienna, Austria (2002)
15. Cranor, L., Langheinrich, M., Marchiori, M., Reagle, J.: The platform for privacy preferences 1.0 (p3p1.0) specification. W3C Recommendation (2002)
16. Anonymizer Inc.: Anonymizer. http://www.anonymizer.com (2003)
17. Bellotti, V., Sellen, A.: Design for Privacy in Ubiquitous Computing Environments. In: Proceedings of the Third European Conference on Computer Supported Cooperative Work (ECSCW'93), Kluwer (1993) 77–92
18. Jiang, X., Hong, J., Landay, J.: Approximate information flows: Socially-based modeling of privacy in ubiquitous computing. In: Proceedings of Fourth International Conference on Ubiquitous Computing (Ubicomp'2002), Göteborg, Sweden (2002)
19. Gelernter, D.H.: Mirror Worlds: Or the Day Software Puts the Universe in a Shoebox: How It Will Happen and What It Will Mean. Oxford University Press (1992)
20. Mann, S.: Smart clothing, turning the tables. In: ACM Multimedia Conf. (1996)
21. Rhodes, B.: The remembrance agent: A continuously running automated information retrieval system. In: The Proceedings of The First International Conference on The Practical Application of Intelligent Agents and Multi Agent Technology (PAAM '96), London, UK (1996) 487–495
22. Goldberg, D., Nichols, D., Oki, B., Terry, D.: Using collaborative filtering to weave an information tapestry. Comm. ACM **35** (1992) 51–60
23. Canny, J.: Collaborative filtering with privacy via factor analysis. In: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Tampere, Finland, ACM Press (2002) 238–245
24. Menezes, A.J., Oorschot, P.C.V., Vanstone, S.A.: Handbook of Applied Cryptography. CRC Press Series on Discrete Mathematics and Its Applications. CRC Press (1996)
25. Pedersen, T.: A threshold cryptosystem without a trusted party. In: Proceedings of EUROCRYPT '91. Volume 547 of Springer-Verlag LNCS., Springer (1991) 522–526
26. Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T.: Secure distributed key generation for discrete-log based cryptosystems. Lecture Notes in Computer Science **1592** (1999) 295–310
27. Fouque, P.A., Stern, J.: One round threshold discrete-log key generation without private channels. Public Key Cryptography (2001) 300–316
28. Wei, D.: Crypto++ Library^{TM} 5.0 (2002)
29. Goldreich, O., Oren, Y.: Definitions and properties of zero-knowledge proof systems. Journal of Cryptology **7** (1994) 1–32