Modeling Privacy Control in Context-Aware Systems

Significant complexity issues challenge designers of context-aware systems with privacy control. Information spaces provide a way to organize information, resources, and services around important privacy-relevant contextual factors.

Xiaodong Jiang and James A. Landay
University of California, Berkeley

Many worry that existing privacy problems will only worsen in context-aware pervasive computing environments.[1,2] Ubiquitous sensing and the invisible form factor of embedded computing devices have made it easier than ever to collect and use information about individuals without their knowledge. Sensitive private information might live indefinitely and appear anywhere at anytime. Moreover, the ability of context-aware systems to infer revealing information from loosely related personal data has even more troubling implications for individual privacy. The risks are high: even a few privacy violations could lead to user distrust and abandonment of context-aware systems and to lost opportunities for great enhancements.

In this article, we describe a theoretical model for privacy control in context-aware systems based on a core abstraction of *information spaces*. We have previously focused on deriving socially based privacy objectives in pervasive computing environments.[3] Building on Ravi Sandhu's four-layer OM-AM (objectives, models, architectures, and mechanisms) idea,[4] we aim to use information spaces to construct a model for privacy control that supports our socially based privacy objectives.[3] We also discuss how we can introduce *decentralization*, a desirable property for many pervasive computing systems, into our information space model, using unified privacy tagging.

An example

We use a hypothetical example to illustrate how you can use decentralized information spaces to model privacy control in a smart office environment. Imagine that Bob, a sales representative from company A, visits Carol, company B's senior manager, at B's headquarters to discuss a potential deal. Bob brings his own laptop, on which a trusted privacy runtime system has been preinstalled. On entering the building, Bob was given a visitor badge and an ID tag for his laptop, both enabled by radio frequency technologies, so that RF readers in the building constantly track his laptop's location.

Bob first meets Carol in her office. As part of the discussion, Carol sends Bob's laptop some internal documents to review and specifies that these documents should only persist for the period of their meeting. The trusted privacy runtime system on Bob's laptop can enforce Carol's preference over data persistence if all documents were properly tagged. Although these documents reside on Bob's laptop, these "privacy tags" dictate that Carol controls them. In effect, such tags define an information space that Carol owns.

After the meeting, Bob and Carol head toward a meeting where employees in Carol's department will discuss the deal with Bob's company. Bob almost forgets to take his laptop when a voice alert sounds a reminder before he leaves Carol's office. The alert is triggered because the privacy runtime system detects a possible unwanted *boundary crossing*—that is, information stored on Bob's machine is left unat-

1536-1268/02/$17.00 © 2002 IEEE

tended in a *physical* space that Carol owns.

Everyone present at the meeting can download the slides on individual machines during Carol's talk. Privacy tags assigned to these slides specify that regular employees can store these slides in thumbnails (a specific level of accuracy) or print them to a printer inside the departmental offices.

Even a few privacy violations could lead to user distrust and abandonment of context-aware systems.

Privacy runtime systems on employees' laptops can enforce these requirements. Bob, as a visitor, cannot print the slides in any form to any printer. Neither could he email them to anyone outside Carols company because the privacy system checks the privacy tags of all outgoing traffic. Thus, unwanted boundary crossings across different *social* spaces can be prevented.

The meeting room's automated audiovisual capture system records the entire session. The privacy runtime system assigns privacy tags to this record to indicate joint control by both Carol and Bob. Some activity inference programs provide high-level activity information using vision-based analysis of raw audiovisual records. Carol likes to make this activity information available to all employees in her department. Bob, on the other hand, only wants it to be available to Carol. A compromise that satisfies both preferences is made based on privacy tags for the inferred data: Carol is the only person in the department who can access this activity record.

When Bob finishes work, RF scanners in the building automatically detect him leaving. As the physical boundary crossing occurs, the privacy system on Bob's laptop garbage-collects all data owned by Carol that reside on the laptop.

Information space model

Next, we describe the theoretical model of information spaces that makes privacy control in the previous example and many others possible. Properly viewed, an information space is a semantic construct around which you can formulate a privacy control policy.

Principals and objects

In context-aware systems, an information space consists of basic objects and is owned by and released to *principals*: *users* and *user agents*. The user is a person or group of people interacting with a context-aware system. The user agent is a software system that serves and protects the user. This distinction affects trust relations in context-aware systems. For example, interactions between multiple levels of user agents have stimulated research on trust modeling in pervasive computing environments.[5]

An *object* refers to any entity to which you can apply authorizations. To keep our model general and applicable to authorization for data with different semantics, we do not make assumptions about the underlying data semantics for which the model controls access. As such, our model can specify access controls for a wide range of information, resources, and services. For example, you could model access to a conference room projector by the write privilege to a Boolean control object that represents the projector's on and off states.

You can capture (often by sensors) many data objects in context-aware systems with different levels of confidence and represent them at different levels of accuracy. Capturing confidence is a number between 0 and 1 that denotes the probability that a sensor's measurement reflects the actual object value.

An object's representational accuracy describes the ease of distinguishing it from other similar objects. We model an object's representational accuracy by using a lattice defined over subset relations. Suppose data a, b are encodings of the same object on different levels of representational accuracy. We say $a = b$ iff a is a less accurate representation than b. Assuming that a, b are elements of a lattice, least upper bound $\text{lub}(a, b)$ denotes the lattice element that is no less accurate than the elements a, b, but no more accurate than absolutely necessary. Likewise, we let greatest lower bound $\text{glb}(a, b)$ denote the lattice element that is no more accurate than the elements a, b, but no less accurate than absolutely necessary. T denotes the most accurate element, and $\perp$ denotes the least accurate element.

For example, we can use a lattice to model a principal's identity. We define p's precise identification as a singleton set $\{p\}$. We define complete anonymity as the empty set $\{\}$ and pseudonyms as a set PN. Thus, we define the accuracy of p's identity as

$$\begin{aligned}
&\text{ID}(p) = \{x \mid p \in x \wedge x \subseteq PN\}\\
&T = \{p\}\\
&\perp = \{PN\}\\
&x \leq y \Leftrightarrow y \subseteq x\\
&\text{glb}(x, y) = x \cup y\\
&\text{lub}(x, y) = x \cap y
\end{aligned}$$

For example, radio frequency identification (RFID) sensors can measure Bob's identity information only with a certain level of confidence because of technical limitations such as noise and signal interference. You can also represent identity data at different levels of accuracy: a person, a male person, either Bob or Doug, Bob, and so on. The change in representational accuracy of Bob's identity helps to achieve *intentional ambiguity*, the ability to blur identity to the extent that Bob's privacy is preserved, or *plausible deniability*, the ability to say "unknown" to a query about Bob's identity.

Capturing confidence is often a property of a particular sensor, whereas accuracy corresponds to the way data are being represented and used. Together they determine an object's *sensitivity* with respect to privacy—that is, how identifiable an object is by a given sensor, captured with a particular level of confidence and represented at a given level of accuracy. For the same type of objects, the higher their sensitivity, the more identifiable they are.

Assume object o has a representational accuracy that corresponds to a discrete

subset of size n and its capturing confidence is p. We can define object o's sensitivity as the reduction in uncertainty (entropy) after we know object o's capturing confidence for a given level of o's representational accuracy:

$$Sensitivity(o) = -\sum_{1}^{n} \frac{1}{n} \log \frac{1}{n} - \left(-p \log p - \sum_{1}^{n-1} \frac{1-p}{n-1} \log \frac{1-p}{n-1} \right).$$

In Bob's example, an RFID sensor in the meeting room identifies him as speaker Bob with a confidence of 70 percent, and we know there are three other speakers also scheduled for the meeting that afternoon). Suppose we treat Bob's identity as a random variable. Before we know the sensor measurement, we can only randomly guess one from the four possible candidates, with an entropy measurement of 2. After we know the measurement, we can compute the probability of the identified person being among the other three speakers as (1 – 70 percent)/3 = 0.1. Thus, the new uncertainty measurement is 1.36. The reduction in uncertainty thus measures how easy it is to identify Bob's identity as one of the four speakers at the meeting.

Information space

Intuitively, an information space provides a way to organize information, resources, and services around important privacy-relevant contextual factors in context-aware systems. Each information space has a group of owners, who determine permissions (for example, readers and writers for objects in the space).

A *boundary*—physical, social, or activity-based—delimits an information space. A physical boundary demarks an information space using physical limits. For example, principal Bob might create an information space that contains all information and resources in his office. A social boundary delimits an information space through social groups (space owners do not have to be part of the group). For example, Bob might create an information space for his family members. An activity-based boundary delimits an information space by including only information about an ongoing activity, such as a meeting Bob is attending.

So, in much the same way physical places provide structural constraints for organizing complex daily activities,[6] information spaces with different types of boundaries also represent different perspectives for organizing information and resources and, thereby, authorizations in context-aware systems. Many context-aware systems already manage information, resources, and services from at least one of these three perspectives. For example, smart office applications[7] typically organize information and resources around physical locations (for example, offices) and activities (for example, meetings). Mobile healthcare applications[8] often organize resources around social groups (for example, the sick).

Our formulation of boundaries of information spaces also coincides with what MIT professor Gary. T. Marx calls *border crossings* when he discusses the social effects of surveillance.[9] He defines four types of border crossings that form the basis of perceived privacy violations: natural, social, spatial or temporal borders and borders due to ephemeral or transitory effects. Central to our many privacy concerns is the prospect for intentional or inadvertent personal border crossings, which could become more frequent in pervasive computing environments. On the other hand, many context-aware technologies exist precisely to identify such borders in our daily life (for example, location tracking to identify natural borders). By formally capturing borders using the boundary abstraction, our information space model provides the basis for leveraging existing context-aware technologies to minimize undesirable border crossings.

Central to our many privacy concerns is the prospect for intentional or inadvertent personal border crossings.

An information space is a 5-tuple (O, P, B, Op, $Perm$), where O is a set of objects representing information or resources, P is a set of principals who own the space, B is a boundary predicate such that all objects in O (and only the objects in O) satisfy B. Op is a set of allowable operations on objects, and $Perm$ is a set of permissions that define principals allowed to perform individual operation in Op.

The three types of boundary predicates essentially define the membership of objects in an information space. Their exact operational semantics rely on application logic and usage contexts and, therefore, you can only define them by individual applications.

We define three operations that you can apply to objects in an information space:

- *Read* and *write*: These operations are what their names suggest.
- *Promotion* and *demotion*: Promotion increases privacy risks by making objects live longer, be captured with a higher level of confidence, or be represented at a higher level of accuracy. Demotion does exactly the opposite.
- *Aggregation*: You can aggregate different objects to give more information. This might be a simple composition of objects, or it might involve inference on information from different sources.

For each object in an information space, owners of the space can assign permissions for each type of operation.

Privacy control

From a privacy-control standpoint, you can consider an information space boundary as a contextual trigger to enforce permissions that owners of that space define.

Both read and write accesses raise privacy concerns because they can alter an information space's boundary. Reading from an information space creates an identical copy of an object in the space and transfers the copy to an outside entity. The result is that a copied object, albeit

identical to the original, might no longer satisfy the information space's boundary predicate. Similarly, writing to an information space tries to replace an existing object in the space with a foreign object that itself might not satisfy the boundary predicate.

Many existing context-aware technologies can help identify the boundaries of information places by

- Demarking physical boundaries through location awareness
- Demarking social and activity-based boundaries through identity and activity awareness

Once such boundaries are identified, a system can help decide the appropriate form of data to be released upon occurrences of boundary crossings. Imagine a context-aware medical alert system for seniors that would minimize the activity details sent daily to an attending doctor but would provide a full account should a medical emergency occur. Such a system decides what to do with data in the patient's information space depending on whether an activity-based boundary has been crossed, such as occurrence of a medical emergency. Similarly, a system can alter capturing confidence (by using alternative sensors) and representational accuracy to provide an appropriate level of data sensitivity for a given purpose of use. This is one way of changing the flow of information between information spaces.[3]

Unified privacy tagging

In and of itself, our information space model is neutral with respect to its possible style of implementation architecture. However, many context-aware systems favor a decentralized architecture for scalability and robustness. So, we extend our basic model of information spaces to support decentralization using *unified privacy tagging*.

Background

Unified privacy tagging uses a form of metadata to identify which information spaces an object belongs to and what permissions it's been assigned. In this way, you can distribute both data and privacy controls in an information space in a context-aware system.

Usage control models in digital rights management systems have used similar ideas.[10] More recently, the IBM Enterprise Privacy Architecture (www.zurich.ibm.com/security/enterprise-privacy) adopted the *sticky policy paradigm*, which mandates that a privacy policy sticks to the data, travels with it, and can decide how you can use the data. These approaches often limit the computations that you can perform on metadata to reads and writes.

The closest work to our unified privacy tagging is Andrew Myers' decentralized label model for a secure programming language called JFlow.[11] In his model, the programmer specifying permissions assigns a label to a program's individual variables and methods. The programmer can then apply static analysis techniques to help eliminate problems such as covert channels. However, this model is intended to address an entirely different class of problems. Additionally, three important differences separate our work and Myers' model:

- As an integral part of our information space model, privacy tags assigned to objects also identify the information spaces to which an object belongs.
- The privacy tags are more expressive than Myers' labels in that you can specify permissions for operations such as promotion and demotion, and the privacy tags include privacy-relevant properties such as capturing confidence, representational accuracy, and data lifetime.
- Our privacy tags are unified in that they represent both virtual tags of data objects and physical tags of physical objects to enable similar privacy control for physical resources.

Unified privacy tagging model

Throughout the following discussion, we use *read* as an example operation for which we define permissions. We can give similar definitions to other types of operations such as *writes* and *promotions*.

Every object in an information space is associated with a *privacy tag*. A privacy tag consists of three parts:

- A *space handle* that specifies which information spaces the object belongs to
- A *privacy policy* that represents permissions specified by space owners for different types of operations (for example, promotion and demotion)
- A *privacy property list* describing an object's lifetime, representational accuracy, and capturing confidence

In particular, a privacy policy is a set of *privacy controls* that express an object's privacy requirements. Each privacy control has two parts: an owner of the data item and a set of readers. Each control is written in the form *owner: readers*. A privacy control's owner is a principal whose data was tagged by this control. A privacy control's readers are a set of principals who the owner permits to access the data item. It is implicitly understood that a privacy control's owner can also read the data item. An example of an expression that denotes a privacy policy is $P = \{o_1: r_1, r_2; o_2: r_2, r_3\}$, where o_1, o_2, r_1, r_2, r_3 denote owners (o_1, o_2) and allowed readers (r_1, r_2, r_3). Privacy controls specified by both o_1 and o_2 should be enforced, so only r_2 can access the data.

A complete privacy tag of an object that belongs to information space 1 is captured with 80 percent confidence, transferred at the first level of accuracy, and allowed to live for five hours might look like $T = \{space1, \{o_1: r_1, r_2; o_2: r_2, r_3\}, \{5hrs, level\ 1, 80\%\}\}$. The system automatically computes and checks these privacy tags for all data against end-user privacy policy specifications.

Privacy tags can also prevent undesirable object aggregations. For example, if you infer object o_1 from o_2 and o_3, the privacy policy in o_1's privacy tag should be the union of o_2's and o_3's privacy policies (that is, inferred data readable by the maximum subset of readers that the privacy policies of o_2's and o_3's owners specify). We can also define rules for computing a privacy property list for a set of aggregated objects from privacy property lists of input objects. For example, you can conservatively define any aggregation operations' lifetime of output

objects as the shortest among lifetimes of all input objects. You can define an output object's capturing confidence as the product of the input objects' captured confidence, if the input objects are statistically independent from each other.

Privacy involves both transferring electronic data across networks and physically transporting data into an insecure place. For example, principal Alice stores sensitive data on her laptop. You can compromise the privacy of Alice's data by taking her laptop, on which sensitive data resides, into Bob's office. So, we also need to control physical object movement.

Our privacy-tagging model is unified in that you can use it to tag both physical and data objects. We can imagine attaching an RFID tag to the laptop, which encodes the principals (for example, only Alice) who can move the laptop. We can define Alice's information space to include all devices in her office and do the same for Bob. Whenever the RF scanner detects that the laptop is about to leave Alice's office and enter Bob's office without explicit authorization, it alerts Alice. This preserves the privacy of Alice's laptop using the same privacy-tagging model for data objects.

The trusted computing base problem

A criticism of all metadata-based approaches is that not only the metadata but also the software component that processes the metadata must be trustworthy. This trust assumption can be problematic for large-scale decentralized systems.

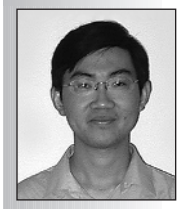
However, unified privacy tagging is part of a theoretical model that you can realize in many different ways. Although appealing, installing a trusted privacy runtime on every client used in a context-aware computing environment to process privacy tags is only one of many possibilities. For example, you don't need to physically collocate privacy tags with the data objects. You can alternatively store them on a centralized server that applications using tagged objects will have to contact. Therefore, we believe the best way to realize the model we've described will depend on different application contexts.

This work is part of our ongoing efforts in developing a four-layer privacy framework for pervasive computing environments. Elsewhere[3] we focused on deriving privacy objectives, drawing from research in social sciences. In this article, we focused primarily on models for privacy control to support achieving these socially compatible privacy objectives. Currently, we are developing a suite of new privacy mechanisms based on the information space model. We will integrate these mechanisms into a new architecture for privacy and security in pervasive computing. P

REFERENCES

1. V. Bellotti and A. Sellen, "Design for Privacy in Ubiquitous Computing Environments," *Proc. 3rd European Conf. Computer Supported Cooperative Work* (ECSCW 93), Kluwer, Dordrecht, the Netherlands, 1993, pp. 77–92.
2. S. Doheny-Farina, "The Last Link: Default = Offline Or Why Ubicomp Scares Me," *J. Computer-Mediated Communication*, vol. 1, no. 6, Oct. 1994, pp. 18–20.
3. X. Jiang, J. Hong, and J. Landay, "Approximate Information Flows: Socially Based Modeling of Privacy in Ubiquitous Computing," to be published in *Proc. 4th Int'l Conf. Ubiquitous Computing* (Ubicomp 2002), Springer-Verlag, Berlin, 2002.
4. R. Sandhu, "Engineering Authority and Trust in Cyberspace: The OM-AM and RBAC Way," *Proc. 5th ACM Workshop on RBAC,* ACM Press, New York, 2000, pp. 111–119.
5. L. Kagal, T. Finin, and A. Joshi, "Trust-Based Security in Pervasive Computing Environments," *Computer*, vol. 34, no. 12, Dec. 2001, pp. 154–157.
6. P. Arge, "Changing Places: Contexts of Awareness in Computing," *Human-Computer Interaction*, vol. 16, nos. 2–4, Mar. 2001, pp. 177–192.
7. A. Fox et al., "Integrating Information Appliances into an Interactive Workspace," *IEEE Computer Graphics & Applications,* vol. 20, no. 3, May/June 2000, pp. 54–65.
8. S. Kirn, "Ubiquitous Healthcare: The ONKONET Mobile Agents Architecture," *Proc. Workshop on Mobile Computing in Medicine* (MCM 02), BJHC, 2002.
9. G.T. Marx, "Murky Conceptual Waters: The Public and the Private," *Ethics and Information Technology*, vol. 3, no. 3, 2001, pp. 157–169.
10. J. Park and R. Sandhu, "Towards Usage Control Models: Beyond Traditional Access Control," *Proc. 7th ACM Symp. Access Control Models and Technologies* (SACMAT 02), ACM Press, New York, 2002, pp. 57–64.
11. A.C. Myers, "JFlow: Practical Mostly-Static Information Flow Control," *Proc. 26th ACM Symp. Principles of Programming Languages* (POPL 99), ACM Press, New York, 1999, pp. 228–241.

the AUTHORS



Xiaodong Jiang is a PhD student at the University of California, Berkeley. His research interests include pervasive and context-aware computing, privacy and security, and human-computer interaction. He has a BS in computer science from Nanjing University, China, and an MS in computer science from the University of Illinois, Urbana-Champaign. He is a student member of the IEEE and ACM. Contact him at 525 Soda Hall, Computer Science Division, Univ. of California, Berkeley, Berkeley, CA 94720-1776; xdjiang@eecs.berkeley.edu.

James A. Landay is an associate professor of computer science at the University of California, Berkeley. His research interests include UI design tools, pen-based UIs, and mobile and context-aware computing. He has a BS in electrical engineering and computer science from the University of California, Berkeley, and an MS and PhD in computer science from Carnegie Mellon University. He is a member of the ACM, ACM SIGCHI, and ACM SIGGRAPH. Contact him at 683 Soda Hall, Computer Science Division, Univ. of California, Berkeley, Berkeley, CA 94720-1776; landay@cs.berkeley.edu.