Automated Privacy Audits to Complement the Notion of Control for Identity Management

Rafael Accorsi

Department of Telematics
Albert-Ludwigs-Universität Freiburg, Germany
accorsi@iig.uni-freiburg.de

Abstract. Identity management systems are indispensable in modern networked computing, as they equip data providers with key techniques to avoid the imminent privacy threats intrinsic to such environments. Their rationale is to convey data providers with a sense of *control* over the disclosure and usage of personal data to varying degree, so that they can take an active role in protecting their privacy. However, we purport the thesis that a holistic sense of control includes not only the *regulation* of disclosure, as identity management techniques currently do, but must equivalently comprise the *supervision* of compliance, i.e. credible evidence that data consumers behave according to the policies previously agreed upon. Despite its relevance, supervision has so far not been possible. We introduce the concept of *privacy evidence* and present the necessary technical building blocks to realise it in dynamic systems.

1 Introduction

In a technological setting where some even prophesy the death of privacy [5], the need for approaches to mediate and legislate for the collection of personal attributes and their usage is increasingly gaining in momentum and relevance. While such an investigation involves interdisciplinary efforts, we focus on the technical aspects. In this context, identity management systems (henceforth IMS) play an essential role in circumventing the privacy threats inherent to the deployment of information technology. They allow data providers to selectively disclose attributes to data consumers, possibly enabling data providers to formulate policies under which collected attributes can or cannot be employed.

The rationale of IMS is to convey a sense of control to data providers, where the "control" stands for the *regulation* of attribute disclosure. However, data providers today obtain no indication as to whether data consumers actually behave according to the policies agreed upon. Put other way, data providers are left with a number of privacy promises or expectations, but obtain no creditable evidence that their policies have been adhered to. Thus, this setting clearly fails to reproduce the established understanding of control individuals have in mind, in which control comprises not only the regulation of a set of activities, but also the *supervision* that this set of activities indeed takes place as expected. As a result of lacking supervision, data consumers often fear that their personal attributes could be (illegally) shared with third parties or used for purposes other than those stated [12].

Please use the following format when citing this chapter:

Accorsi, R., 2008, in IFIP International Federation for Information Processing, Volume 261; *Policies and Research in Identity Management*; Eds. E. de Leeuw, Fischer-Hübner, S., Tseng, J., Borking, J.; (Boston: Springer), pp. 39–48.

We close this gap by investigating the technical building blocks necessary to realise supervision in dynamic systems, i.e. open and adaptive systems based on ubiquitous computing technologies [9]. Addressing supervision requires a conceptional change, though. Traditional IMS build on observability, unlinkability and unidentifiability and therefore use a number of techniques, such as pseudonyms and partial identities over anonymous communication channels. In addition to this, recent IMS allow data providers to formulate policies and stick them to data, a concept called to as "sticky policies" [4]. (We refer to [10] for a comprehensive survey on techniques for IMS.) Thus, current techniques aim at an *a priori*, preventive protection of privacy. In contrast, when investigating supervision we found ourselves in an *a posteriori* setting where techniques to verify the compliance with privacy policies are needed.

To realise this, we employ the concept of *privacy evidence* [12]. Its rationale is to make the behaviour of the data consumer regarding data collection and enforcement of privacy policies evident to data providers. Intuitively, a privacy evidence is a record consisting, on the one hand, of all the information collected from and related to a particular data provider – a so-called *log view* – and, on the other hand, the result of an automated audit of this log view based on the policies of the data provider. Together, these pieces of information build the basis for supervision and thereby pave the way for a holistic realisation of control.

The thesis we purport is that investigation towards a holistic realisation of control for informational self-determination in IMS is indispensable. Due to the improved transparency inherent to privacy evidence, such realisation of control has the chance to increase the confidence placed on the data consumers and even foster the willingness to disclose personal attributes, which is an essential factor for the acceptance of dynamic system in general and for the deployment of personalised services [13] in particular. Eventually, both data providers and data consumers could equally profit from such an extended notion of control.

This paper is structured as follows. In §2, we present the technical setting underlying our approach and the main building blocks necessary to realise the concept of privacy evidence. These building blocks are then described sequentially: in §3, we introduce a language for the expression of privacy policies; in §4, log views based on a secure logging service are presented; and in §5, we describe our approach to auditing log views based on the stated privacy policies. We discuss our work and provide perspectives for further work in §6.

2 Technical Setting and Building Blocks

The realisation of privacy evidence anticipates the steps depicted in Fig. 1. In (1), a data provider A formulates a policy P_A and communicates it to the data consumer. Since we consider dynamic systems with implicit interactions, we assume that policies are communicated before joining the system. (Implicit interactions take place without the awareness of the data provider.) When interacting with the system, a number of events are recorded as entries in log files (2). In fact, we assume that *every* event is recorded, so that log files offer a complete digital representation of the activity in a dynamic system. At some point in time the data consumer may retrieve the log view S_A containing all

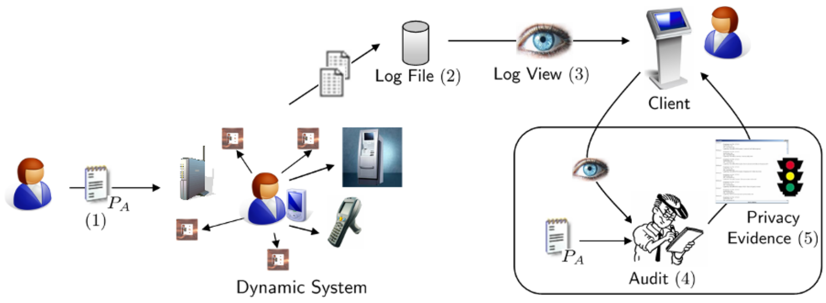


Fig. 1. The workflow for privacy evidence.

the log entries related to A (3). A can then visualise the collected data and start a third-party automated audit process (4) to check whether the policies P_A have been adhered to, thereby generating the corresponding privacy evidence (5).

To realise privacy evidence, the following central technical building blocks are essential: a *policy language* for the expression of privacy preferences in dynamic systems; *log views* to allow the visualisation of recording activity; a *secure logging* to ensure the authenticity of recorded data, in particular to improve the credibility of log views; and an *automated audit* process for checking the adherence to policies. In the forthcoming sections, we describe the work towards the realisation of privacy evidence.

Assumptions. In our work, we consider the following assumptions. First, *every* event happening in the system, as well as every access to collected data is recorded as an event in a log file. Second, on interacting with the system, data providers are identified while the events they are involved in are recorded. That is, the entries in the log file are always related to a data provider. Third, while the system is dynamic in that it adapts itself to the data providers' preferences, it is static regarding the data collection possibilities. Technically, this means that the ontology describing the system does not change over time and, hence, the policies of data providers do not become obsolete. Although these assumptions do not hold in general, they hold for some scenarios, as the one we consider in §6.

3 A Policy Language for Dynamic Systems

A policy language allows data providers to specify a set of rules, i.e. a policy to regulate the access to their attributes, whereas execution monitors on the data consumers' side enforce these rules and record the authorisation decisions for further inspection. However, in dynamic systems the sole expression of access rights is not enough. Policies for dynamic systems should also allow data providers to express which attributes may or may not be collected. The policy language we propose therefore builds on two notions: *access* and *collection*. In contexts where the distinction between these notions is irrelevant, we simply refer to them as an *act*.

We enrich atomic acts with conditions for *usage control*. Usage control extends traditional access control techniques by allowing data providers to specify *provisions*

```
 1.  <Policy>     := (<Rule>) | (<Rule>), <Policy>
 2.  <Rule>       := <Col_Ctrl> | <Col_Ctrl>, if (<Cond>) |
 3.                  <Acc_Ctrl> | <Acc_Ctrl>, if (<Cond>)
 4.  <Col_Ctrl>   := <Perm>, <Subj>, <Obj>, <Event>
 5.  <Acc_Ctrl>   := <Perm>, <Subj>, <Obj>, <Right>
 6.  <Cond>       := <Atom_Cond> | <Atom_Cond> && <Cond>
 7.  <Atom_Cond>  := <Provision> | <Obligation>
 8.  <Provision>  := role <Op> <Role> | purpose <Op> <Purpose>  |
 9.                  <DataField> <Op> <Value>
10.  <Obligation> := delete <DataField>  <Temp_mod> [<Sanction>] |
11.                  notify <DataProvider> <Temp_mod> [<Sanction>]
12.  <Perm>       := allow | deny
13.  <Right>      := read | write | exec <Cmd>
14.  <Temp_mod>   := immediately | within <Nat_Number> days
15.  <Sanction>   := otherwise <String>
16.  <Op>         := > | < | >= | <= | == | !=
```

Fig. 2. Policy language for dynamic systems.

and *obligations* [11]. Intuitively, provisions express the conditions that must be fulfilled in order to grant or deny an act [7]. For example, access to the profile of data provider A is granted only for accounting purposes. Obligations express events that must occur once an act is granted or denied. For example, data provider A wants to be notified whenever the collection of attributes via RFID readers take place.

Figure 2 depicts the core definition of our policy language in BNF-notation. Intuitively, the policy of a data provider A is a finite set of rules $P_A = \{r_1, \ldots, r_n\}$ (Line 1), each of which can stand for a (conditional) act, i.e. collection or access regulation (Lines 2 and 3). When formulating a collection rule, A stipulates whether a certain subject is able to collect an attribute and/or event (Line 4). The same applies for the formulation of access rules (Line 5). In both cases, the wildcard $\star$ can be used to represent a whole class of, e.g., subjects or attributes. Conditions can include provisions and obligations (Line 7): provisions regard the role a subject takes, as well as the purpose of the access or collection and the value of collected data fields serving as guards (Lines 8 and 9); obligations encompass the deletion of some attribute within a certain timeframe and the notification of individuals (Lines 10 and 11). Obligations may or may well not include sanctions that hold in case a particular obligation is not fulfilled (Line 15).

The actual value of terminals, such as `Obj` and `Subj` are application-dependent and omitted here for simplicity. (To this end, we have defined data models corresponding to our scenario.) To illustrate how formulated rules appear and exemplify their expressive power, in Fig. 3 we consider two rules for the data provider A. Rule r_1, stipulates that A grants read access to his attributes provided the accessing subject adopts the role "Marketing", the purpose is personalised service and the accessed attribute is deleted within 30 days. In the case of non-adherence, a compensation of \$100 is due. Rule r_2 prohibits the collection of data by RFID-readers.

```
r1 := ( allow, *, *, read,
      if ( role == Marketing &&
           purpose == PersService &&
           delete * within 30 days otherwise Fine=$100$ ))
r2 := ( deny, RFID-Reader, *, * )
```

Fig. 3. Examples of policy rules.

4 Secure Logging and Log Views

Log data is a central source of information in computer systems. In contrast to other rather "static" files, such as text documents or spreadsheets, log files allow one to reconstruct the dynamics of a system, i.e. the course of events that led to some particular state. Hence, log files are a central source of information for audits. However, to be useful and credible, log data must be authentic, i.e. it must fulfil the following properties [1]:

- *Integrity* states that log data faithfully reflects the state of the devices, i.e., the log data is accurate (entries have not been modified), complete (entries have not been deleted), and compact (entries have not been illegally added to the log file). Thus, log data is not modified, deleted, or appended during the transmission to and storage at the collector.
- *Confidentiality* states that log entries cannot be stored in clear-text, for such log data can be easily accessed to duplicated.

The authenticity properties of log data must be realised with cryptographic techniques which account for *tamper evidence*, i.e., attempts to illicitly manipulate log data are detectable to a verifier, and *forward integrity*, i.e. log data contains sufficient information to confirm or rebuke allegations of log data modification before the moment of the compromise. (Put another way, forward integrity states that if an attacker succeeds in breaking in at time t, log data stored before t cannot be compromised.)

Based on [1], we present below the basis of a secure logging service. There are two kinds of actors in a logging setting: the *devices* sense the environment and communicate changes therein in the form of events to a *collector*, whose responsibility is to sequentially record these events. Assuming that the communication between devices and collectors cannot be manipulated, here we focus on the collector and the corresponding mechanisms to ensure the authenticity of recorded data.

In our approach, log data is secured when recording the entry associated to an event and not as a separate process. Each log entry E_j is (symmetrically) encrypted with an evolving cryptographic key K_j obtained from a secret master key A_j and an index field W_j. (The latter is used to describe the data provider to which the entry refers.) A hash chain Y associates the previous entry E_{j-1} and the current. This procedure is depicted in Fig. 4, where the numbers correspond to:

1. $A_j = Hash(A_{j-1})$ denotes the authentication key of the jth log entry. The confidentiality of this information is essential as it is used to encrypt log entries. Thus, we

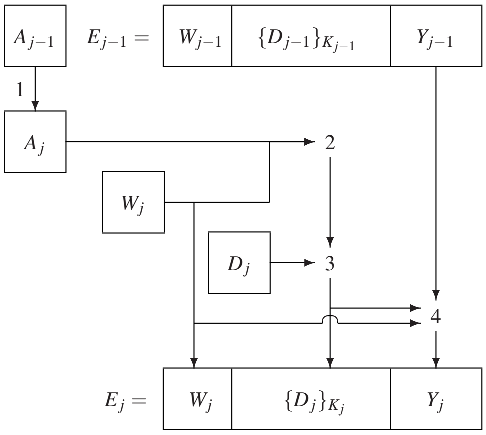


Fig. 4. Adding an entry to the log file.

assume that the computation of the new value irretrievably overwrites the previous value.

2. $K_j = Hash(W_j, A_j)$ is the cryptographic key with which the jth log entry is encrypted. This key is based on the index W_j, so that only corresponding data providers gain access to the entry.
3. $\{D_j\}_{K_j}$ is the encrypted log entry D_j.
4. $Y_j = Hash(Y_{j-1}, \{D_j\}_{K_j}, W_j)$ is the jth value of the hash chain. Each link of the hash chain is based on the corresponding encrypted value of the log data.

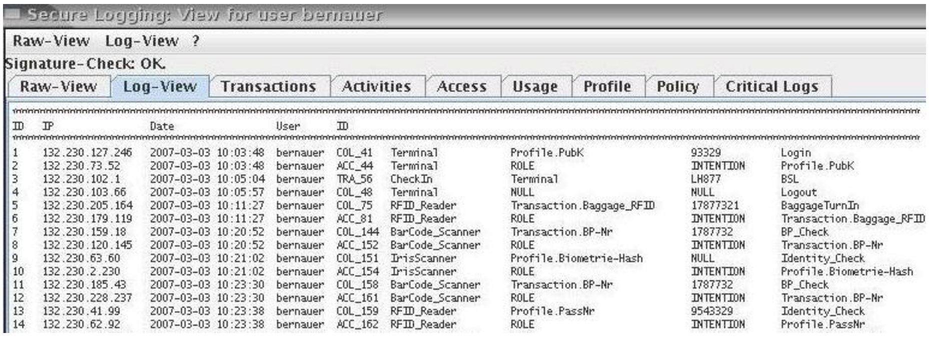
The generated log entry, denoted $E_j = W_j, \{D_j\}_{K_j}, Y_j$, consists of the index W_j, the encrypted log entry $\{D_j\}_{K_j}$, and the hash chain value Y_j.

4.1 Log Views and their Generation

A central concept to allow supervision is to furnish data providers with timestamped information regarding *which* attributes have been collected, *who* has had access to them and *how* collected attributes have been used. In our approach, these pieces of information are compiled into a *log view* [12], a concept bearing similarity with its homonymous counterpart in the field of databases.

Log views are individualised audit trails consisting of factual data (performed transactions, collected attributes, etc.) and monitored data (access and usage information) about a particular data provider, as well as meta data – in the form of a digital signature – about the generating data consumer and the integrity of a view. Figure 5 illustrates a part of log view of a data provider referred to as “bernauer”.

As for the generation of log views, to retrieve a log view S_A the data provider A employs a trusted device (e.g. a home computer or a terminal dedicated to this purpose) to authenticate himself to the data consumer, who then starts a query over (possibly distributed) log files. Intuitively, the index of each entry is checked against the authenticated data provider. If they match and the entry passes an integrity check (based on



```
ID  IP               Date                 User      ID
1   132.230.127.246  2007-03-03 10:03:48  bernauer  COL_41   Terminal          Profile.PubK              93329      Login
2   132.230.73.52    2007-03-03 10:03:48  bernauer  ACC_44   Terminal          ROLE                      INTENTION  Profile.PubK
3   132.230.102.1    2007-03-03 10:05:04  bernauer  TRA_56   CheckIn           Terminal                  LH877      BSL
4   132.230.103.66   2007-03-03 10:05:57  bernauer  COL_48   Terminal          NULL                      NULL       Logout
5   132.230.205.164  2007-03-03 10:11:27  bernauer  COL_75   RFID_Reader       Transaction.Baggage_RFID  17877321   BaggageTurnIn
6   132.230.179.119  2007-03-03 10:11:27  bernauer  ACC_81   RFID_Reader       ROLE                      INTENTION  Transaction.Baggage_RFID
7   132.230.159.18   2007-03-03 10:20:52  bernauer  COL_144  BarCode_Scanner   Transaction.BP-Nr         1787732    BP_Check
8   132.230.120.145  2007-03-03 10:20:52  bernauer  ACC_152  BarCode_Scanner   ROLE                      INTENTION  Transaction.BP-Nr
9   132.230.63.60    2007-03-03 10:21:02  bernauer  COL_151  IrisScanner       Profile.Biometrie-Hash    NULL       Identity_Check
10  132.230.2.230    2007-03-03 10:21:02  bernauer  ACC_154  IrisScanner       ROLE                      INTENTION  Profile.Biometrie-Hash
11  132.230.185.43   2007-03-03 10:23:30  bernauer  COL_158  BarCode_Scanner   Transaction.BP-Nr         1787732    BP_Check
12  132.230.228.237  2007-03-03 10:23:30  bernauer  ACC_161  BarCode_Scanner   ROLE                      INTENTION  Transaction.BP-Nr
13  132.230.41.99    2007-03-03 10:23:38  bernauer  COL_159  RFID_Reader       Profile.PassNr            9543329    Identity_Check
14  132.230.62.92    2007-03-03 10:23:38  bernauer  ACC_162  RFID_Reader       ROLE                      INTENTION  Profile.PassNr
```

Fig. 5. Part of a log view for data provider `bernauer`.

the hash chain), then the content of the entry is decrypted and added to the log view of A. When all the entries are queried, the resultant view is signed and sent back to the inquiring data provider.

5 Automated Audits and Digital Privacy Evidence

Log views would, at least in theory, suffice to realise the holistic sense of control we argue for in this manuscript: data providers could browse through their log views and check whether their privacy policies have been adhered to or not. However, this is more intricate than it seems. Log views can easily include thousands of entries and their interrelationships are often hard to comprehend and reconstruct, regardless of how much effort we put into improving their readability.

We develop an approach to audit log views parameterised by the policies of data providers. Intuitively, given a policy $P := \{r_1, \ldots, r_n\}$ and a log view S, we define a transformation ν that takes P and returns the set of rules $V_P = \{v_1, \ldots, v_n\}$ such that each $v_i \in V$ denotes the violation of the corresponding rule r_i. To illustrate this, consider the rule r_2 in Fig. 3. By applying the transformation ν, the following violation is generated:

$v_2 :=$ `( allow, RFID-Reader, *, * ).`

This denotes that the collection of attributes through RFID readers is allowed, thereby contradicting the original desire of the data provider.

With V_P at hand, we then search for violations in the log view of the corresponding data provider. To this end, we define the pinpoint relation $\triangleright$ between views and the set of violations V_P such that $S \triangleright v_i$ if v_i can be pinpointed, i.e. detected, in S. If there is a $v_i \in V_P$ such that $S \triangleright v_i$, then there is an execution of the system that violates r_i and, in consequence, the policy P. In contrast, if there is no such v_i, such that $S \triangleright v_i$, then a violation of P can be ruled out. Technical details are found in [2].

We employ a semaphore notation to make the result of audit evident to the pertinent data provider. In this case, red obviously stands for a violation of some rule, while green denotes the compliance with a policy. An amber semaphore indicates that some obligation-based rule could not be pinpointed and therefore stands for a warning. Such

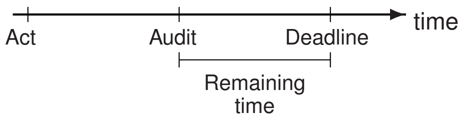


Fig. 6. Condition leading to an amber semaphore

a warning is triggered whenever a log view S is audited before the deadline of a pending obligation, as illustrated in Fig. 6.

A log view, together with the corresponding audit analysis, constitutes a privacy evidence. In the case of a violation, an individual may click over the semaphore and obtain details on which rules have been violated as well as the entries that led to this result. A similar procedure can be carried out when the semaphore shows amber.

6 Discussion and Perspectives

Taking stock, in this manuscript we purport the thesis that a holistic notion of control for IMS encompasses not only the regulation of communicated (respectively, collected) attributes, but also the supervision of adherence to stated policies. While this understanding of control as regulation and (at least the option of) supervision is prevalent in the common language, to our knowledge it has not been considered in the context of IMS. We firmly believe that the investigation of approaches to realise such forms of control is the next milestone towards the development of IMS to cope with the privacy challenges of dynamic systems.

We see various advantages arising from a holistic notion of control. Today, individuals tend to have doubt than confidence that computing systems behave according to privacy policies. In consequence, individuals aware of the imminent privacy threats are reluctant to use technologies, even in cases where this would be advantageous. The use of tools supporting the regulation and supervision introduced in this manuscript offer a unique chance to revert these figures. For individuals can only feel confident when a certain level of transparency and assurance is at hand; this is achieved by means of privacy evidence.

We currently test this approach within an airport as those proposed by the IATA. The idea is to employ several ubiquitous computing technologies to automate the check-in of passengers. Basic technologies include the replacement of traditional boarding passes with 2D barcode boarding passes that could even be printed at home and the replacement of luggage tags with RFID tags. Several airlines plan to extend the vision further and include biometric identification and other communication media, e.g. mobile phones and PDAs. In such a setting, the assumptions we made in §2 are realistic.

The ideas presented here open up several interesting perspectives for further research into the subject. Below, we elaborate on some of them. First, by relaxing the assumptions made above, we are left with the fact that log entries may fail to refer to a data provider and the question is how to decide whether an "unknown" entry refers to a particular data provider or not. This is a relevant question, as data consumers could intentionally hide relevant entries from the log view and thereby influence the result of the audit. To tackle this problem, techniques for IT forensics, possibly in combination with the methods for data mining, may be needed.

Second, the term privacy evidence has an implicit legal connotation, one we knowingly do not explore in this manuscript. While in our approach we use trusted sandboxes [3, 6] to attest the existence of the corresponding logging and log view generation algorithms, we are aware that this is not enough for judicial evidence. (We recall that the audit is performed on a trusted device and, hence, does not pose a problem.) To transform privacy evidence in legally acceptable evidence with corresponding probative force, the notion of chain-of-custody [8] for evidence, for instance, should be investigated in greater detail.

Finally, the approach we propose does not exclude traditional IMS techniques. On the contrary, it *complements* them. It would thus be interesting to see more case studies using our techniques, as well as other developments, for supervision. We believe this will substantiate the importance of supervision as a distinguishing factor for future IMS and privacy-aware (dynamic) systems.

References

1. R. Accorsi. On the relationship of privacy and secure remote logging in dynamic systems. In S. Fischer-Hübner, K. Rannemberg, L. Yngström, and S. Lindskog, editors, *Proceedings of the 21st IFIP TC-11 International Security Conference: Security and Privacy in Dynamic Environments*, volume 201 of *International Federation for Information Processing*, pages 329–339. Springer-Verlag, 2006.
2. R. Accorsi and M. Bernauer. On privacy evidence for UbiComp environments – Broadening the notion of control to improve user acceptance. In A. Bajart, H. Muller, and T. Strang, editors, *Proceedings of the 5th Workshop on Privacy in UbiComp*, pages 433–438, 2007.
3. R. Accorsi and A. Hohl. Delegating secure logging in pervasive computing systems. In J. Clark, R. Paige, F. Pollack, and P. Brooke, editors, *Proceedings of the 3rd International Conference on Security in Pervasive Computing*, volume 3934 of *Lecture Notes in Computer Science*, pages 58–72. Springer Verlag, 2006.
4. M. Casassa-Mont, S. Pearson, and P. Bramhall. Towards accountable management of privacy and identity. In E. Snekkenes and D. Gollmann, editors, *Proceedings of the European Symposium on Research in Computer Security*, volume 2808 of *Lecture Notes in Computer Science*, pages 146–161. Springer-Verlag, 2003.
5. M. Froomkin. The death of privacy? *Stanford Law Review*, 52(5):1461–1543, May 2000.
6. A. Hohl. *Traceable Processing of Personal Data in Remote Services Using TCG*. PhD thesis, University of Freiburg, 2006.
7. S. Jajodia, M. Kudo, and V. Subrahmanian. Provisional authorizations. In A. Ghosh, editor, *E-Commerce Security and Privacy*, pages 133–159. Kluwer Academic Publishers, 2001.
8. E. Kenneally. Digital logs – Proof matters. *Digital Investigation*, 1(2):94–101, June 2004.
9. G. Müller. Privacy and security in highly dynamic systems. *Communications of the ACM*, 49(9):28–31, September 2006.
10. A. Pfitzmann. Multilateral security: Enabling technologies and their evaluation. In G. Müller, editor, *Proceedings of the International Conference on Emerging Trends in Information and Communication Security*, volume 3995 of *Lecture Notes in Computer Science*, pages 1–13. Springer-Verlag, 2006.
11. A. Pretschner, M. Hilty, and D. Basin. Distributed usage control. *Communications of the ACM*, 49(9):39–44, September 2006.

12. S. Sackmann, J. Strüker, and R. Accorsi. Personalization in privacy-aware highly dynamic systems. *Communications of the ACM*, 49(9):32–38, September 2006.
13. J. Strüker. Der gläserne Kunde im Supermarkt der Zukunft. *Wirtschaftsinformatik*, 49(1):59–62, January 2007.