Noname manuscript No.
(will be inserted by the editor)

Vibhor Rastogi · Evan Welbourne
Nodira Khoussainova · Travis Kriplean · Magdalena Balazinska
Gaetano Borriello · Tadayoshi Kohno · Dan Suciu

Expressing Privacy Policies Using Authorization Views

Received: date / Accepted: date

Abstract In this paper, we design a rule-based privacy policy for the RFID Ecosystem, an RFID-based ubiquitous computing system. We start from the *physical access control* (PAC) rule (Kriplean *et al.*, *IEEE Pervasive Computing* 2007) that provides a default level of privacy but constrains the possible set of applications. We extend it by using principled ways of defining other access control rules that retain the simplicity of PAC and yet provide increased flexibility for users and applications. We then propose authorization views as a database technique to implent rule-based privacy policies and demonstrate the use of authorization views over the privacy policy designed for the RFID Ecosystem.

1 Introduction

Privacy in ubiquitous computing systems depends on a complex set of dynamic and interdependent factors. Key factors include the control that users have over what, when, where, how, and to whom their personal information is disclosed [14]. The privacy problem for ubiquitous systems, as Bellotti and Sellen note [3], is related to the way "technology can attenuate natural mechanisms for feedback and control". For location aware systems, the lack of control is particularly severe when sensor data is captured and stored by infrastructure that is outside the user's direct control [15]. In this case, users' concerns over control of personal data and trust in a centralized authority are eminent barriers to success. At the same time, centralized systems provide users with definite advantages such as: administration and maintenance services, expertise and vigilance in security, amortized cost of shared infrastructure, and higher (backed up) storage capacity.

V. Rastogi
185 Stevens Way, Seattle, WA 98195-2350
Tel.: (206)-543-1695
Fax: (206)-543-2969
E-mail: vibhor@cs.washington.edu

The RFID Ecosystem project at the University of Washington is an RFID-based pervasive computing system with a ubiquitous building-wide deployment of RFID readers and antennas. It is a capture-and-access system: data is collected at the readers and streamed into a trusted central database. In this paper, we address privacy issues for the collected RFID data. Our goal is to develop policies that offer users an intuitive sense of privacy and provide a control over their data that matches everyday life expectations, while maintaining the advantages of a centralized approach. Furthermore, we intend to implement and deploy these methods in our building-wide test bed so that we can inform and evaluate our research with longitudinal, in-situ user studies.

Any implementation of policies on privacy and information disclosure in a centralized system must be tightly integrated with the data management subsystem. As such, it seems natural to investigate and possibly adapt existing techniques for database security and privacy. Indeed, recent work on security and privacy from the database community is encouraging. For example, there has been research on "sanitization" to remove private information from personal data [8], efficient fine-grained access control [17], even Hippocratic databases, which include the privacy of data they manage as a founding tenet [1]. A key goal of our work is to apply or adapt database security and privacy techniques to ubiquitous computing.

As a first step for achieving privacy, we provide control over information disclosure through context-aware, rule based access control. A key challenge [15,3] is that privacy rules are difficult for users to specify, understand, and manage. Moreover, the physical, social, and historical context of the querier, the subject of the query, the query itself, and the accuracy of the returned result must all be considered [14]. Incorporating such contextual information is difficult especially when it is inferred or uncertain in nature. We propose to address these issues in two ways. First, we tightly constrain the space of privacy rules to those which model common modes of information disclosure in everyday life [13] or which reference a

specific higher-level event (e.g., a meeting next Tuesday). In this way, we minimize the difficulty and need for user specification and management of privacy rules while also making them intuitively understandable. Second, we introduce authorization views [17] as a flexible framework for translating privacy rules into fine-grained access control for databases. Authorization views also allow us to leverage a rich set of additional database techniques such as those that deal with uncertainty [16].

2 Related Work

There are a number of approaches for controlling access to personal data in ubiquitous computing systems including location privacy techniques such as [4, 11]. In this paper, we concentrate on fine-grained rule-based access control policies.

Specification of Rule-Based Policies Interfaces for specifying rule-based privacy policies have ranged from simple web forms [5, 9], to elaborate GUIs and social metaphors [15], to complex and expressive languages [5, 9]. As noted in the introduction, the lack of initiative and ability on the part of users to set, understand, and manage privacy rules has been a serious impediment to these rule-based approaches. We posit that a crucial problem with past approaches is that the space of allowable rules is too broad. Furthermore, we argue that the problem can be mitigated to some extent by constraining ourselves to a small, rigid set of pre-defined rules that model specific, commonly acceptable (or unavoidable) scenarios for information disclosure in everyday life. In general, rule-based policies may have to be supplemented with explicit user-level interventions when applications require information disclosure beyond what may be commonly acceptable to users. Such mechanisms can be built on top of our rule-based framework, but our aim is to minimize the need for such interventions by developing simple, intuitive sets of rules that require minimal management.

Rule-Based Policy Enforcement Complementary to rule-specification interfaces, systems for efficient checking and enforcement of rule-based policies such as CPOL and Houdini have also been proposed [5, 9]. The focus of these works is on real-time checking of multiple rules defined in a flexible, expressive specification language. There are two key distinctions between these systems and our authorization view approach. First is that these systems apply only to current user context, while our database-centric approach applies to both current and historical (i.e. stored) context. Secondly, for these systems, extensions are limited to new rule sets and custom-coded filters [5, 9]. In contrast, authorization views are implemented as part of a database management system and so can leverage a rich variety of data management techniques such as those that deal with uncertainty. Authorization views is a well known database technique that provides fine-grained access control [17]. Our specific contribution is to use it for providing privacy in ubiquitous systems.

3 Physical Access Control

In this section, we describe Physical Access Control (PAC), an access control policy introduced in [13]. PAC restricts the information a user can obtain to that which the user could have observed in person. We argue that PAC provides a default level of policy that is commensurate with everyday life.

More specifically, PAC allows a user to access information about his objects and other users (but not the objects of others as they may be concealed from sight) when and where he was physically present. We also argue that this affords a level of privacy that is easily understood by users, although user studies are still necessary for verification. Formally, PAC is defined by the following two rules:

- **PAC rule 1:** If *user v is colocated with another user u at location l and time t* then the information "*user v is located at l at time t*" can be released to u
- **PAC rule 2:** If *user u owns an object o and u is colocated with o at location l and time t* then the information "*object o is located at l at time t*" can be released to u

Even though PAC is conservative in the information it reveals, the user-centric view of the data still allows some useful service primitives. For example, if a user is trying to locate a lost object, she can obtain the location where she last saw it, which is a likely answer. Similarly, queries over the history of a user's own activities are allowed. These queries enable applications such as reminder services that alert users when they forget to take an item with them as they go home for the day. Alternatively, a personal digital diary can record the places a user visits, whom she has contact with and what activities she is involved in so that she can later study trends in her use of time.

4 Extensions to PAC

As a default access control policy, PAC supports surprisingly many applications. However, PAC is just the first step. There are a number of applications that we wish to support in the RFID Ecosystem that cannot be done using PAC. To support such applications and exploit the true potential of RFID technology, we require a more relaxed and flexible privacy policy. We give below a list of scenarios that illustrate some of the desired applications.

1. Alice took Bob's stapler from his desk. Bob would like to use the RFID Ecosystem to find it. When Bob queries the location of his stapler, the system notifies Alice that Bob is looking for his stapler.

2. Bob has loaned a book to his friend Charlie. When Bob queries the location of his book, the system returns that it is with Charlie. Although Charlie has agreed to revealing this information, he does not want the system to disclose his actual location.
3. Alice and Bob have a scheduled meeting. Alice is waiting, but Bob does not show up, possibly, because he is running late. When Alice queries Bob's location, the system reveals it allowing Alice to figure out that Bob is on his way. Similarly, when Bob queries Alice's location, the system reveals it allowing Bob to know that Alice is ready and that he should hurry.
4. At lunch time, Alice would like the system to make her location public to all of her friends, so that they can join her for lunch.
5. Whenever Alice goes for a coffee break to the coffee room, she would like the system to disclose her location to all of her friends.
6. Alice has a button on her cell phone that when pressed allows her to make her location public to all of her friends. In addition, there is also a cloak button that allows Alice to make her location private.
7. The head of security, at night or in the event of a fire, is able to track the location of any person in the building.
8. Bob is organizing a colloquium scheduled at 3:30 PM. He sees that there are not many people in the room at 3:31 PM. Bob can query for the number of people in the hallways. This allows Bob to decide whether to wait for more people to arrive or not.
9. Alice would like to find Bob. Alice uses the RFID Ecosystem to "page" Bob with a message saying that she would like to know his location. Bob accepts this page by pressing a specific button on his cellphone, thereby revealing his current location to Alice.

In this section, we formulate a rule-based privacy policy that extends PAC and enables scenarios such as those mentioned above. Like PAC, the rules proposed regulate the release of information providing a level of privacy that is intuitive and simple to understand for the user. Users can not define new rules, but can choose the rules they wish to activate, thus configuring the amount of personal information disclosed to other users.

4.1 Classification of rules

Each of the above scenarios consists of three factors: a user that asks the query, a context that describes the scenario, and the information that is released. To support a scenario with context c, released information i, and user u, we define an access control rule (as done in [15]) of the form "if context is c then information i can be released to user u". Many languages have been designed to formally specify such rules [2]. We give examples of access rules for some of the above scenarios using plain text in Fig 1. The actual language is orthogonal to our discussion.

Ownership: If *user v carries an object o owned by another user $u at time T* then the information "*user v carries object o at time T*" can be released to *user $u*

Meeting: If *user v is at location L at time T during a scheduled meeting with another user $u* then the information "*user v is located at L at time T*" can be released to *user $u*.

Authority: If *user v is at location L at time T during night and $u is the head of security* then the information "*user v is located at location L at time T*" can be released to *user $u*

Friendship: If *user $u has friend permissions from user v for the time interval T* then the information "*user v is at location L at any time $t during the interval T*" can be released to *user $u* at time *$t*.

Fig. 1 Examples of access control rules for the RFID Ecosystem

The number of access rules for each user is large. To study the rules systematically and make them intuitive for users to understand, we divide the rules into categories based on their context. The basic intuition is that context can be deconstructed into answers to questions of the form 'What?', 'Why?', 'Where?', 'When?', and 'Who?' [14]. We define categories corresponding to each type of question. For example, the most prominent question for a time-based rule is 'When does it become applicable?'. Such rules are placed in the 'When' category. We give below a short description of each category.

Event-based (When). These are rules that become applicable in case of a special event. Scenarios 3, 4, and 8 fall into this category. In general, event-based rules allow users to access other users' locations in case of a predefined event. The released information may be in abstracted form such as counts, as in Scenario 8.

Role-based (Who). These are rules that become applicable due to the special role of the inquirer. Scenario 7 falls into this group. We have observed that in the RFID Ecosystem, there are few role-based rules. Most of them are defined for safety or physical security purposes. In other context-aware systems there may be need for more role-based rules. For example, an RFID system deployed in a business firm may have multiple roles like managers, directors, etc.

Location-based (Where). These are rules that become applicable due to the physical location of the user. Scenario 5 falls into this category. In general, we allow users to define a list of public locations and a list of friends. Whenever the user enters one of the public locations, her location becomes public to all of her friends.

Intention-based (Why). These are rules that become applicable due to real-time user input. All rules described above were passive. Once defined, they do not require user input for activation. In contrast, intention-based rules require real-time user input to get activated. Scenarios 6 and 9 fall into this category. Implementing such rules requires simple interfaces so that providing

the input does not become too burdensome for the user. We are considering ways to provide such interfaces.

Ownership-based (What). These are rules that become applicable due to ownership relation between a user and an object. Scenarios 1 and 2 fall into this category. In scenario 1, no information is released to the user, but instead an action is taken on his behalf. In scenarios 2, information (in abstracted form) is provided based on permission granted by the subject (Charlie).

4.2 Design Principles

To enable new types of applications in the RFID Ecosystem, new rules that fall into one or more of the above categories may need to be defined. The question that arises then is what are the privacy implications of adding new applications and defining new rules. We first address this question specifically for the ownership-based rules.

The main goal of the ownership-based rules is to enable people to find their objects. This is a very common application in RFID-based systems, so it is critical to support it well. Although seemingly simple, naive definitions of ownership rules can cause significant private information leakage.

As described earlier, for the ownership rule used in scenario 1, our definition reveals no information. If it did, the subject of the information (Alice) would not have any control in the release of her information. In contrast, information is released by the ownership rule used in scenario 2. The subject (Charlie) controls the release of information in this case.

In scenario 2 we reveal to Bob the information that *Charlie carries Bob's book* — i.e., Charlie is in the active process of transporting Bob's book between physical locations. Consider an alternative version that reveals the actual location of the book. This clearly leaks information, as revealing the book's location to Bob may cause complete disclosure of Charlie's location. Consider another version of ownership rule, one that reveals that *Charlie is with Bob's book*. This version can still reveal Charlie's location in the following scenario: Bob hides his book in the bathroom. If the system reveals to Bob that *Charlie is with Bob's book*, Bob can conclude that Charlie is in the bathroom. Thus the choice to release the information *Charlie carries Bob's book* is more privacy-preserving. This condition is true only if the book is not being carried by someone else and Charlie has changed his location in possession of the book. An RFID system can detect this by comparing the location of Charlie and Bob's book over a period of time.

As this example illustrates, designing rules is difficult because there are often unintended consequences. To facilitate rule design, we propose to keep the following issues in mind.

Subject vs. Controller: Each rule is assigned a controller, the user who controls whether the rule is activated. As mentioned above, in the rule for Scenario 1 the subject is different from the controller. Such rules need to be designed carefully. We, in fact, allow no information to be revealed by this rule. For scenarios 2 to 6, and scenario 9 the rules that are defined, have the same subject and owner. These rules are relatively much safer to define. As long as the subject understands the rules, it is her responsibility whether she wants to release information through the rules. Similar to the ownership rule, scenarios 7 and 8 have subjects that are different from the controller. Scenario 8 tries to evade the privacy problem by just releasing the count, while scenario 7 involves a trusted controller with the required authority.

Scenario vs. Context: Another issue to consider is the ambiguity in the mapping between scenario and its context. We know that rules are defined for specific scenarios but are described using context. An attacker can create an alternative scenario that has the same context. This allows the attacker to obtain information about the alternative scenario even from rules that were not created for it. For example, the attack in which Bob hides books in the bathroom creates an alternative scenario which has the same context *Charlie is with Bob's book*. This requires us to define a more appropriate context *Charlie is carrying Bob's book* that makes it difficult for Bob to get Charlie's location information. In general, one should use appropriately defined contexts so as to make it burdensome for the attacker to replicate them using alternative scenarios.

4.3 Coarse Grained Control

We allow users to independently activate and deactivate each of the access rules they control. Although this provides fine grained control, it may become cumbersome for users to manage such rules. In addition, privacy disclosures may occur because of incomplete understanding and mismanagement of rules [15].

To address the above problem, we propose to provide users an alternate control mechanism that reduces the number of available options. The control mechanism is organized as a hierarchy of system-defined rules. Users just need to select the levels appropriate for them. At each level of the hierarchy, additional rules are activated. The PAC rules are activated at level 0. The semantics for the hierarchy follows a simple 'Tit for Tat' principle, which is similar to the principle of minimum asymmetry [10]. A user can choose to be at level i in which case the user's information can only be revealed using rules at or below level i, and in addition, the user will be able to access information using rules at or below level i. An extremely conservative user can choose level 0, thus ensuring high privacy. However, that user will only be able to access data using rules at level 0.

As an example, we give the following hierarchy. It is based on the classification given in Section 4.1. This makes it easier for the users to understand exactly which rules are activated at each level.

- **Level 0:** PAC rules + (role-based superuser rules)
- **Level 1:** Intention-based rules
- **Level 2:** Ownership-based rules
- **Level 3:** Event-based rules
- **Level 4:** Location-based rules

The hierarchy is provided for coarse grained control and is in addition to the fine grained control for each rule. Other hierarchies are also possible.

5 Implementing Policies with Authorization Views

In this section we describe some primitives useful for defining access control policies. We then discuss a database technique–authorization views–which is particularly well-suited for implementing these access control policies. For concreteness, we demonstrate the use of authorization views for the access control policy defined for the RFID Ecosystem.

5.1 Data model for the RFID Ecosystem

For the RFID Ecosystem, the collected data is stored in the form of tables, which we call base tables. In addition to the base tables, a set of materialized views are defined, which are called helper views. Materialized view is a database term that denotes a logical table populated using the result of a predefined query. Fig 2 gives the base tables and helper views used for the RFID Ecosystem.

Base tables
- ***LocatedAt(user** u**, location** l**, time** t**)***
- ***Owns(user** u**, object** o**)***: user u owns object o
- ***Head(user** u**)***: Indicates whether u is head of security

Helper Views
- ***Carries(user** u**, object** o**)***: u is carrying object o
- ***HasMeeting(user** u**, user** v**, time** t**)***: v has a scheduled meeting with u at time t
- ***FriendOf(user** u**, user** v**, time interval** T**)***: u has friend permissions from v for the time interval T

Fig. 2 List of base tables and some helper-views for the RFID Ecosystem

Some of the helper-views are populated through user input. For example, the $HasMeeting$ view defined in Fig 2 is populated using information from the user's schedule. Other helper-views, like the $Carries$ view, are populated by inferring high-level events. This is clearly a non-trivial process and may involve activity recognition. For inferring high-level events we use PEEX [12], a system being developed at the University of Washington that detects high-level events from RFID data.

PACView($u,v,L,T):
$LocatedAt(\$u, L, T)$,$LocatedAt(v, L, T)$

PACView2($u,o,L,T):
$LocatedAt(\$u, L, T)$,$LocatedAt(o, L, T)$,$Owns(\$u, o)$

OwnershipView($u,v,o,T):
$Owns(\$u, o)$,$Carries(v, o, T)$

MeetingView($u,v,L,T):
$HasMeeting(\$u, v, T)$,$LocatedAt(v, L, T)$

AuthorityView($u,v,L,T):
$Head(\$u)$,$Night(T)$,$LocatedAt(v, L, T)$

FriendshipView($u,v,L,$t):
$FriendOf(\$u, v, T)$,$LocatedAt(v, L, \$t)$,$inInterval(\$t, T)$

Fig. 3 Some authorization views for the RFID Ecosystem

5.2 Authorization views

Each user can only access a part of the RFID data. The accessible data is specified by the access control rules. We need a method that takes a set of access control rules, like the ones described in Fig 1, and computes exactly the data that is accessible by the user. For this purpose, we use authorization views, a common database access control technique [17]. Authorization views are logical tables that specify exactly the accessible data.

For each access rule, an authorization view is defined that stores the accessible data corresponding to the rule. For example, the PAC rule 1 has the authorization view PACView($\$u, v, l, t$). The view is defined in datalog notation as

$$\text{PACView}(\$u, v, l, t) \text{:- LocatedAt}(\$u, l, t), \text{LocatedAt}(v, l, t)$$

Here the two references to the LocatedAt table are separated by ',' to denote conjunction. Datalog [6] is a formal language for expressing view definitions. The above definition states that if $(\$u, l, t)$ and (v, l, t) are tuples in the LocatedAt table then the tuple $(\$u, v, l, t)$ is present in PACView. In other words, PACView($\$u, v, l, t$) stores the location information of all users v that were colocated with $u. Fig 3 gives the authorization views corresponding to the PAC rules as well as for the rules of Fig 1.

5.3 Answering queries using Authorization views

As explained above, authorization views specify the accessible data for each user. However, we are interested in answering queries issued by a user. These queries are written in terms of base tables and helper views and not in terms of authorization views. We do not expect a user to know all the authorization views defined for him. Moreover, we do not want the user to have to change her query whenever a new access rule (i.e a new authorization view) is defined.

Answering queries using views is a well studied problem in databases [7]. We illustrate the method using an example. Suppose that user $\$u$ queries the location of user v at time t. Let us denote this query as Q. Thus, Q asks the location of v from the LocatedAt table. To answer Q, it is first rewritten in terms of the authorization views, i.e views defined in Fig 3. Multiple rewritings of Q are possible. We choose the maximal rewriting, one that contains more data in its answer than any other possible rewriting. Suppose that the maximal rewriting uses PACView($\$u, v, L, t$). Thus, the rewritten query checks the location of v in PACView($\$u, v, L, t$). Due to the way PACView has been populated, the tuple $(\$u, v, L, t)$ will appear in it only if v was colocated with $\$u$ at time t. If the tuple $(\$u, v, L, t)$ appears, the answer to the query is L and it is given to the user. If not, the query does not return any answer, thus hiding the actual location of v.

Some access rules require the system to perform an action on behalf of the user. Such rules can not be completely implemented using authorization views. For example, in scenario 1, the system sends an e-mail on Bob's behalf to Alice. For implementing such rules we need an application defined on top of the authorization views that performs these actions. For example, there may be an application that sends e-mails to recipients based on entries of a materialized view.

6 Challenges: Probabilistic data and events

RFID readers produce streams of low-level observations of the form: "Tag 344 was seen at antenna 13 at 3:20pm". However, users are more interested in higher-level events such as "Alice entered the conference room at 3:20pm", or "Charlie carries Bob's book". In fact, as we have seen, success of some access control rules depends crucially on detecting such higher-level events. There are several challenges raised by extracting abstract events.

First, extracting abstract events is difficult due to the low reliability (e.g. missed readings) and the inherent ambiguity in the data (e.g. a sequence of tag reads can correspond to more than one abstract event). This is addressed by PEEX [12], a system for cleaning and extracting higher level events from RFID data. PEEX handles the issues by using a probabilistic model, and thus extracting and managing probabilistic events.

The second challenge is in implementing access control policy over probabilistic events. The implementation using authorization views described in Section 5 assumes deterministic RFID data. In presence of uncertain data, we propose to exploit techniques used in probabilistic databases [16] to implement authorization views. This is a crucial challenge that we illustrate using the PAC rule.

Suppose a user A asks: "Is user B currently at location L?". If A is at L, then PAC allows the correct answer. If A is not at L, then PAC refuses to reveal B's location. However, for probabilistic data, probabilities p_A and p_B would be assigned to the chance that A is at L and B is at L respectively. In the probabilistic context, the correct answer is no longer yes or no but in fact p_B. Yet the system cannot return p_B in the case that A is not at L. One approach is to return $p_A \cdot p_B$, the probability that both A and B are at L. This reveals too much, however – even if p_A is small (A is not likely to be at L), A can still compute p_B. More generally, the requirement is that if A is likely to be at L (p_A is large) then the system should reveal p_B. Otherwise, the system should hide this information. One ad-hoc strategy is to return $min(p_A, p_B)$. We plan to explore more principled approaches that fulfill this requirement.

7 Conclusion

We define an access control policy for the RFID Ecosystem. We start from PAC, an intuitive, spatially-based default access control policy that users can reason about. To support more powerful applications, we define additional rules based on simple and intuitive design criteria. We show how these rules might be composed for coarse grained access control. In addition, we specify primitives for defining the rules, and demonstrate the use of authorization views as a natural technique for implementing them.

References

1. Agrawal, R. et. al. Hippocratic databases. In *VLDB*, 2002.
2. Barth, A. et. al. Enterprise privacy promises and enforcement. In *WITS*, 2005.
3. Bellotti, V. et. al. Design for privacy in ubiquitous computing environments. In *ECSCW*, page 75, 1993.
4. Bettini, C. et. al. Protecting privacy against location-based personal identification. In *SDM*, 2005.
5. Borders, K. et. al. CPOL: High-performance policy evaluation. In *SIGSAC*, 2005.
6. Ceri, S. et. al. Datalog: A self-contained tutorial. Technical report, Mailand, 1989.
7. Duschka, O. et. al. Answering recursive queries using views. In *SIGMOD*, pages 109–116. ACM, 1997.
8. Dwork, C. et. al. Differential privacy. In *ICALP*, 2006.
9. Hull, R. et. al. Enabling context-aware and privacy-conscious user data sharing. In *MDM*, 2004.
10. Jiang, X. et. al. Approximate information flows: Socially-based modeling of privacy in ubiquitous computing. In *Ubicomp*, volume 2498, pages 176–193. Springer, 2002.
11. Jonker, W. et. al., editor. *Secure Data Management, Second VLDB Workshop, SDM 2005*. Springer.
12. Khoussainova, N. et. al. Probabilistic rfid data management. Technical report, UW-CSE-07-03-01, Mar. 2007.
13. Kriplean, T. et. al. Physical access control for captured rfid data. *IEEE Pervasive Computing*, to appear.
14. Lederer, S. et. al. Towards a deconstruction of the privacy space. In *Proc. of Workshop on Privacy in Ubicomp 2003*, Oct. 2003.
15. Lederer, S. et. al. Personal privacy through understanding and action: five pitfalls for designers. *PUC*, 2004.
16. Re, C. et. al. Query evaluation on probabilistic databases. *IEEE Data Eng. Bull*, 29(1):25–31, 2006.
17. Rizvi, S. et. al. Extending query rewriting techniques for fine-grained access control. In *SIGMOD*, 2004.