MUPPET: Mobile Ubiquitous Privacy Protection for Electronic Transactions

Winnie Cheng[1], Jun Li, Keith Moore, Alan H. Karp
Digital Printing and Imaging Laboratory
HP Laboratories Palo Alto

HPL-2006-141(R.1)
March 22, 2007*

mobile technologies, privacy, security, access control, database, ubiquitous computing, information management, services infrastructure

Mobile companions such as smartphones and PDAs are very personal and carry a lot of sensitive data about their owners. With new services aimed at providing more targeted information retrieval through increased interactions with these devices, privacy concerns of individuals must be addressed. Existing solutions give users little control over release of this information. MUPPET is a privacy-aware information brokerage framework that incorporates a number of novel techniques to give users control over the release of their data. First, it introduces *Operation-focused Access Control*, a purpose-based access control model that supports flexible and fine-grain policies using typed operation labels. Second, our system allows *Reward-Driven Information Exchange*. It provides a protocol for explicit communication of justifications and rewards and tunable privacy policies based on ongoing evaluation of the information exchange. Third, MUPPET includes a Purpose Detection Engine with an intuitive user interface for purpose management and supports explicit as well as implicit purpose activations based on context or authorizations. To validate our design, the MUPPET prototype has been integrated with a coupon personalization application for two different service providers in an experimental retail kiosk setting.

* Internal Accession Date Only
[1] Massachusetts Institute of Technology, Cambridge, MA 02139

Approved for External Publication

MUPPET: Mobile Ubiquitous Privacy Protection for Electronic Transactions

Winnie Cheng
Massachusetts Institute of Technology
Cambridge, MA 02139
Email: wwcheng@mit.edu

Jun Li, Keith Moore and Alan H. Karp
Hewlett-Packard Laboratories
Palo Alto, CA 94304
Email: {jun.li, keith.moore, alan.karp}@hp.com

Abstract— **Mobile companions such as smartphones and PDAs are very personal and carry a lot of sensitive data about their owners. With new services aimed at providing more targeted information retrieval through increased interactions with these devices, privacy concerns of individuals must be addressed. Existing solutions give users little control over release of this information. MUPPET is a privacy-aware information brokerage framework that incorporates a number of novel techniques to give users control over the release of their data. First, it introduces *Operation-focused Access Control*, a purpose-based access control model that supports flexible and fine-grain policies using typed operation labels. Second, our system allows *Reward-Driven Information Exchange*. It provides a protocol for explicit communication of justifications and rewards and tunable privacy policies based on ongoing evaluation of the information exchange. Third, MUPPET includes a Purpose Detection Engine with an intuitive user interface for purpose management and supports explicit as well as implicit purpose activations based on context or authorizations. To validate our design, the MUPPET prototype has been integrated with a coupon personalization application for two different service providers in an experimental retail kiosk setting.**

I. INTRODUCTION

Mobile devices are becoming increasingly capable and connected. Mobile technologies have grown to deliver more computing power and features than desktop PCs of the last decade. Today, smartphones and PDAs can run complex operating systems and standalone databases. They often include a wide range of communication options from Bluetooth to GPS. With these advances, integrated services that were once impractical are turning into reality. Not only can users send e-mail and surf the web with these gadgets, they will soon be serving as a user's wallet and much more [2]. A study in 2005 [15] shows that users are getting more comfortable with these devices and there is a rising demand for value-added services.

Currently, little attention has been placed on the privacy ramifications of next generation mobile service infrastructures. A user's mobile device can potentially contain very sensitive information revealing the owner's identity, payment information and various usage histories. At the same time, service providers are leveraging the interconnectivity of information sources and advances in information retrieval to provide intelligent services that target the particular user. Automatic customization and personalization of content delivery bring relevant information to users when they want it and where they want it. As these mobile companions interact with various service providers in the environment, information may be given out unintentionally.

Existing privacy protection mechanisms (eg. W3C's P3P[11], EPAL[21]) often rely on pre-established trust relationships and policy specifications that define contracts on how data should be handled under different circumstances and the obligations of service providers to ensure proper handling of this data. Enforcement is left to the legal system. Legal consequences may deter some malicious uses of sensitive collected data. However, there are many grey areas in defining and justifying proper uses of collected data. Today, users have little control over the dissemination of their information.

Mobile environments pose additional challenges in protecting privacy. A mobile digital companion may interact in many *ad-hoc* networks with different service access points as the user roams around. This makes *a priori* trust relationships difficult to establish and manage. Imagine Alice the shopper who carries her smartphone around the shopping mall of the future. Her smartphone is her electronic wallet and contains her preferences and transaction history. As Alice walks through various stores in the shopping mall, advertisement displays and information kiosks communicate with her mobile device to find the most relevant products and offers for Alice. While the interaction between Alice's smartphone and these kiosks can enrich her shopping experience, these new mobile service infrastructures raise a number of concerns. The underlying communication should be transparent to a user like Alice. At the same time, Alice should have greater control over the information that is exchanged. We have developed MUPPET (Mobile Ubiquitous Privacy Protection for Electronic Transactions) to address these issues.

MUPPET is a privacy-aware information brokerage framework that incorporates a number of novel techniques to give users control over their data. In mobile settings, user intentions can be explicitly captured or implicitly inferred. Such intentions or purposes provide important cues to the types of information exchanges that should be allowed. Hence, MUPPET is designed based on the concept of purpose-based access control [4] but provides significant extensions in supporting greater flexibility and expressiveness. Moreover, with MUPPET, users become aware of the reason and reward involved in these information transactions and have full control

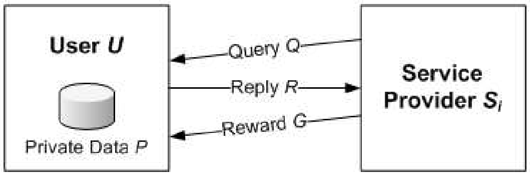


Fig. 1. System Model

over the purposes that can be activated.

This paper is organized into six sections. First, it provides an overview of our system and the scope of the problem we are tackling. Section III describes the detailed design considerations and core system components in MUPPET. In Section IV, we elaborate on the MUPPET prototype and our deployment experiences in a retail kiosk environment. Finally, Section VI summarizes our system and proposes research directions for extending this work.

II. System Overview

MUPPET is intended to serve as a trusted privacy protection system running on the user's mobile device. Our system focuses on restricting the disclosure of sensitive information at the source. MUPPET assumes that all user data to be protected is stored locally on a device such as a smartphone or PDA. This device participates in the exchange of information involving user data with other service providers. MUPPET is a system that regulates and dynamically monitors the information exchanged during such electronic information transactions. It acts on behalf of the user in controlling access.

Our system model consists of a user U and a number of service providers S_i. The private data P is known only to user U initially. Each interaction with a particular provider is referred to as a *session*. During a session, the user may explicitly request information from the service provider. More importantly, the service provider may be able to bring relevant information to the user by actively profiling the user through a series of questions, which may compromise user privacy. Therefore, MUPPET takes on the following model where queries Q are always generated from the service provider and user sends replies R for these queries. Replies may include a subset of P. By answering these queries, the user expects some benefits or rewards to result from this communication. The problem MUPPET focuses on is the restriction of P while allowing rewards G to be earned.

Consider the scenario of Alice the user (U) going to a local electronics store to purchase a USB memory stick. She carries with her a PDA (acts-for U) that contains her grocery shopping list, purchasing history at various stores along with her medical allergy and personal identification information. Her shopping list and purchasing history may include items such as hygiene products (eg. wart remover) that she is sensitive about disclosing. We consider the following two threats. There is the intentional or inadvertent disclosure of information irrelevant to the purpose of the interaction, such as the electronics store disclosing Alice's need for wart remover. This threat is to the user. The other is that the user may attempt to answer queries incorrectly to gain rewards of higher value. Here the risk is to the service provider, but this risk is not substantial in our mobile service environment. These rewards typically involve personalized information retrieval, customization and service/product offerings that service providers are willing to present to users. Also, they are often of low monetary value.

In this scenario, Alice may be leery of an electronics store that questions her medical conditions. Information should only be released on a need-to-know basis. Furthermore, anonymity may be an important consideration. It has been shown that 87% of Americans could be identified by records listing solely their birth date, gender and ZIP code [1]. Even after a number of queries and interactive sessions, a service provider should not be able to obtain all three values. There are also situations where a user may be comfortable in disclosing a fuzzified version of the information. Alice may be sensitive about revealing her exact age but not her age group. In a retail setting, this information is very useful to the service provider in tailoring age-dependent products. With the right amount of information, the electronics store can help Alice locate products and present coupons on discounted items that she likes.

MUPPET operates on data in relational format and supports queries with basic SQL constructs. A user's private data is assumed to be stored in a trusted mobile device interacting with an untrusted service provider. This data is described in a number of database relations and the schema is assumed to be known to the service providers.

Based on our system model and the mobile environment, we arrive at the following system requirements.

- No Trusted Third Party
- Decentralized Solution
- Simple and Expressive Policies
- Dynamic Access Control

A user mobile device may interact with many service access points, many of which it may never have dealt with before. Assessing the integrity of these service providers is difficult and it is unreasonable to assume that a trusted third party can make access decisions on the behalf of all users or can accurately determine how much one can trust a service provider. Also, mobile devices need to continue to function amidst network disconnections. Hence, we require a decentralized solution with policy enforcement running locally on the mobile devices. In terms of the access control mechanism, policies should be simple to invoke and specify yet be expressive enough to support a variety of common access control models. Finally, as users come to know their environment and the service provider, they may want to activate/deactivate or edit policies. Also, the system should allow access to be optionally granted as a result of the reward involved.

III. Design

The approach our system takes is most closely related to work on purpose-based access control in database systems ([4], [9]). A *purpose* can be labeled with a relation, attribute, sets of attributes or tuples to define when the data object should be revealed. In these systems, a service provider that has been certified by a trusted third party to fulfill a particular purpose may retrieve data objects that have been labeled for that purpose. This method has the advantages that policy specification is very natural and access enforcement is lightweight. Hence, it is attractive for our mobile service environment. However, current systems provide a binary decision on the access of a data object. That is, either the access is allowed and data is revealed in its original form or access is denied. Often, a user may be willing to reveal some related information or fuzzified form of the data object. For example, as mentioned earlier, Alice the shopper may be willing to reveal her age group (that is, +/- 5 years of her age) but not her exact age. This information is still very valuable to a service provider that attempts to retrieve advertisements aimed at different groups. MUPPET introduces *Operation-focused Access Control* (OAC) to allow typed data objects that can be subject to different operations such as transformations and perturbations before disclosure. OAC also supports concurrent and generic composition of purposes not limited to hierarchical precedence. These features lead to a more flexible and fine-grain typed labeling scheme we term *operation label*.

In a mobile setting, information exchange is an interactive process where the user may change his intended purpose or policy definition. We make two important observations:

1) *Mismatch between user's perception and the actual service provider's need-to-know.*
   We noticed that there are situations in which a user may have prevented access to some data that may not initially be obvious as to why it is needed by a service provider. For example, an important profiling decision at a retail service provider is whether a user is shopping with a business account as this can affect the types and quantity of products desired. The corresponding query from the retailer is often expressed as the request for a business phone number and may be too intrusive.

2) *User may prefer to adjust the policies or allow access after knowing the reward involved.* A survey conducted in [8] on privacy concerns of loyalty card systems suggests that under some situations, a fraction of shoppers are willing to give up some privacy for the benefits they gain.

MUPPET is designed as a privacy advisor for the user and can accurately inform the user when a policy violation is detected. However, the final decision should rest with the user. Our system is built with such *reward-driven information exchange* in mind to help users make informed decision about their risks and provide the necessary infrastructure for promise assessments and auditing.

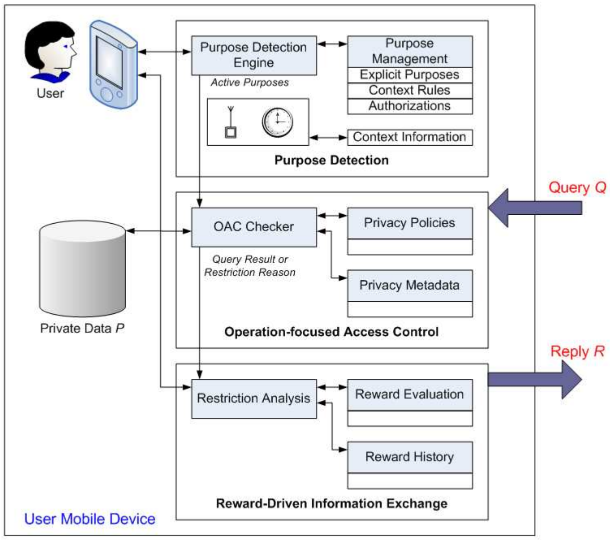


Fig. 2. MUPPET System Overview

Many privacy protection mechanisms assume the existence of a trusted third party that can certify the credentials of a service provider. These systems require a method to identify providers large and small. They also require connectivity to the trusted third party or reliance on public key infrastructure. Such third party certification is undesirable with the vast number of possible providers the user may interact with and the need to support disconnected operations. Our system leverages the rich human interaction and context information present in mobile environments. Users often have a good sense of the various purposes that they wish to achieve and hence the types of information exchanges that should be allowed. MUPPET supports both explicitly and implicitly activated purposes.

MUPPET is a mobile privacy protection system that incorporates these 3 concepts: *Operation-focused Access Control*, *Reward-Driven Information Exchange* and *Purpose Detection*. Figure 2 shows an overview of MUPPET. The purpose detection engine determines the active purposes that the user is currently engaged in. The OAC checker uses this information along with the defined policies to determine whether access should be granted and how data should be revealed. The reward-driven information exchange provides a protocol for service providers to communicate with the user the reasons for certain queries and to negotiate rewards.

A. *Operation-focused Access Control*

Operation-focused access control is a data-centric, purpose-oriented, access control mechanism. Privacy policies are specified by defining *operation label(s)* on data objects.

1) Policy Specification: An operation label can be assigned to a data object at various abstraction levels. It may be defined on a relation, an attribute, specific tuples or cells as shown

TABLE I
Descriptions of Sensitivity Types

Sensitivity Type	Description
UNMODIFIED	Value returned as-is.
CONJUNCTIVE	Value returned only if at most $(N-1)$ of N values of the conjunctive clause will be released.
DISJUNCTIVE	Value returned only if at most 1 of N values of the disjunctive clause will be released.
OVERLAP	Value returned only if in at most K overlapping queries recently.
BREADTH	Value returned only if at most B buckets of a data item will be released.
QUANTIZATION	Value returned after introduction of some random error within a specified range.
PERTURBATION	Value returned after value has been re-mapped.
AGGREGATE	Value is not returned but aggregate and other derived values may be returned.
EXISTENTIAL	Value is not returned but may be used in conditional expressions.
BLOCKING	No value is not returned.

below. An operation label is made up of a purpose and a sensitivity type. The sensitivity type specifies the details of the privacy concerns and how the data should be revealed. Multiple operation labels may be defined on a data object. For example, in a cell-level policy, the data objects are identified by the tuple matching condition and the attribute. Details on how to handle them are defined in the associated operation label.

operation_label := {purpose, sensitivity_type}
relation_level_policy := {relation_id, operation_label}
attribute_level_policy := {global_attribute_id, operation_label}
cell_level_policy := {condition, global_attribute_id, operation_label}

A sensitivity type can involve dynamic attributes, stateful query history and access patterns in specifying how access checks on the data object should be done and the data perturbations required prior to disclosure. A malicious service provider may attempt to gain access to information by asking a number of queries in slightly different forms. In many of these situations, access decisions must be dependent on historical interaction behavior. Furthermore, whether a user is sensitive about a particular data object may be tied to its actual value. For example, users may be willing to disclose body weight if they have succeeded in meeting a weightloss target. Some of the types currently used in MUPPET are shown in Table I.

The UNMODIFIED and BLOCKING types represent the two opposite ends of the information disclosure spectrum allowing the data to pass through in its original form or not at all. The DISJUNCTIVE and CONJUNCTIVE types express conditions in which the user may want to allow some items of a collection to be released but not the entire collection or more than a certain number of them. They are examples of M-of-N privacy concerns. For example, as mentioned earlier, the combination of birth date, gender and ZIP code can be used to identify a user with high probability. In this case, one may specify a CONJUNCTIVE policy on these attributes and once two of the attributes have been disclosed, the last one will not be revealed.

The OVERLAP and BREADTH types restrict access based on the fraction of data range that has been searched. In particular, the OVERLAP type prevents a service provider from zooming in on the exact value of a data object by limiting the number of overlapping results between queries. The BREADTH type is used to guard against queries that sample across a broad range of the result set.

Types may be defined to reduce the accuracy of the returned result. Some examples are QUANTIZATION, INPUT_PERTURBATION and OUTPUT_PERTURBATION. There are situations where queries are sent to determine the category of a user. These categories represent rough grouping of users based on certain attributes. For example, a service provider may wish to know the age group or income bracket of an individual. This information can be collected with high accuracy even with some errors introduced into the original data. The QUANTIZATION type addresses the privacy concerns of revealing sensitive data values that are involved in these queries yet allowing useful information to be collected by the provider. Input data is randomized by an error margin (+/- error e) before the result is returned. The probability p of getting a correct categorization with bucket size s is given by

For $e \leq s$:

$$p = \frac{2}{s}\left[\int_0^e \frac{e+x}{2e}\,dx + \int_e^{s/2} 1\,dx\right] = 1 - \frac{e}{2s}$$

When $e << s$, the probability is close to 1.

Perturbation can be applied to preserved some statistical properties of data without disclosing actual data values. Generating pseudonyms can protect the anonymity of users, names and identifiers while allowing aggregate behavior to be analyzed. Information such as brand loyalty can be easily inferred without knowing the names of the brands involved. MUPPET provides two kinds of perturbations. INPUT_PERTURBATION hides the data values (eg. product name) before the query is applied whereas OUTPUT_PERTURBATION perturbs some of the result of the query that may be more suited for dynamic attributes (eg. concatenated attributes).

A user may never want to reveal some data values, but the privacy concern may not be as stingent as to define the data object as BLOCKING. It may be acceptable to release some derived results based on these values. The EXISTENTIAL type allows data value to be used only in the conditional evaluation portion of a query. The type can be configured to support some comparison operations and not others. The AGGREGATION type is based on the idea that an aggregate value such as SUM and COUNT may represent less sensitive

TABLE II
ENFORCEMENT AND PRECEDENCE RULES

Rule 1:
Access to a finer-grain data object is denied if access to ANY of the overlapping larger granularity data objects is denied.

Rule 2a:
A data object with default ALLOW_ALL_EXCEPT is denied access if there exists an active BLOCKING label on the object.

Rule 2b:
A data object with default DENY_ALL_EXCEPT is denied access if there exists no active label that is not of BLOCKING type on the object.

Rule 3:
At a particular granularity level, access is denied if ANY of the candidate labels prevents access.

Precedence Rule 1:
Among operation-labels on the same data object at the same granularity, the operation label with the highest priority purpose has precedence. This label is referred to as the candidate label.

Precedence Rule 2:
Among candidate-labels, the candidate label of the greatest strictness has precedence.

information than the individual values that it is aggregating over. The user can define a data object to be of this type and specify the aggregation and user-defined functions that are allowed.

MUPPET demonstrates the use of operation labels as typed data objects and the illustrated examples are only some of the types that may be defined.

2) Policy Enforcement: The OAC checker examines the specified policies and determines how the incoming query should be answered. The query may be denied completely, granted to execute directly on user data or be subject to transformations and perturbations. Before the OAC checker is invoked, the Purpose Detection engine identifies the set of currently active purposes. It also provides a priority value based on the user rankings of the precedence of these purposes. The active purposes determine which policies or operation labels should be considered.

Multiple or overlapping operation labels may influence the policy to be applied to a data object. These labels may be of different granularity and types. In this section, we show how they can be handled with a simple set of enforcement rules. Policies may conflict in two ways: within one granularity level or between different granularity levels.

Access checks are performed in order of the granularity of data items described in active labels. More specifically, the relation-level check happens first, followed by the attribute-level and tuple/cell-level checks. At each level, access is checked for the data objects that are involved in the incoming query. For access to be granted at a finer granularity, access must be granted for all overlapping larger granularity data items. If a relation is inaccessible, then none of its attributes may be revealed in any manner. This is stated as Rule 1 in Table II.

To support flexible policy authoring and eliminate ambiguity between operation labels, MUPPET requires all relations and attributes to have an operation label with the system default purpose *. These labels are used to assign either a ALLOW_ALL_EXCEPT or DENY_ALL_EXCEPT policy by using <*, UNMODIFIED> and <*, BLOCKING>, respectively. This enables easy exclusion and inclusion of access for specific purposes. For example, a user may want to allow his/her business phone number to be revealed when performing banking related operations. One can set the following policy on the business phone number attribute: <*, BLOCKING>, <Banking, UNMODIFIED>.

Whether a data object is accessible at a granularity level, in particular at the relation-level and attribute-level granularity, is dependent on its system default purpose operation label. For ALLOW_ALL_EXCEPT, there must be no active operation labels that completely denies its access (i.e. of type BLOCKING). On the other hand, for DENY_ALL_EXCEPT, there must be at least one active operation label that permits its access (i.e other than type BLOCKING). Rule 2a and 2b in Table II capture the privacy concerns of the system defaults. These defaults are configurable by the user or policy maker.

In either case, if access is not denied, there exists at least one active non-BLOCKING label and MUPPET selects the operation label with the highest priority purpose. This label, referred to as the candidate label, is then checked according to its configured type parameters to see whether access is allowed. CONJUNCTIVE labels, for example, will be checked to verify that not all of the sensitive data objects have been accessed in the recent past, which can be defined on a per-session basis or based on sliding queries window. At the end of the access check for the granularity level, all candidate labels must agree to allow access for the OAC checker to continue at a finer granularity level.

It is possible for different granularity levels to conclude with different candidate labels on a data object. In this case, we use *strictness* as a measure of the privacy sensitivity of a type. Each type belongs to a *Sensitivity Class* that has a *strictness* value associated with it. Table III shows how the types are grouped into these classes. The *strictness* value is assigned based on the relative information entropy characteristic of the class. For example, the EXISTENTIAL type never reveals the data directly and hence can be viewed as having a higher entropy (and greater strictness) than CONJUNCTIVE where some attributes may appear as-is during a session. Our decision is a conservative one using the candidate label that is most strict in such cases.

Notice that while some of these types theoretically exist at different granularity levels. Some may be more suitable for specific levels or data types. QUANTIZATION is most reasonably applied to numeric values and at attribute- or cell/tuple-level rather than on an entire relation. Section IV

TABLE III
SENSITIVITY CLASS AND STRICTNESS

Sensitivity Class	Strictness	Sensitivity Type(s)
`Public`	1	`UNMODIFIED`
`InterQuery`	2	`CONJUNCTIVE, DISJUNCTIVE, OVERLAP, BREADTH`
`Perturbation`	3	`QUANTIZATION, PERTURBATION`
`Reduction`	4	`AGGREGATE, EXISTENTIAL`
`Confidential`	5	`BLOCKING`

will discuss this in greater details.

B. Reward-driven Information Exchange

The OAC checker is conservative in enforcing policies. Users are likely to be, and are encouraged to be, more stringent on the policy specification initially when little is known about the service provider and its environment. When a query is denied, MUPPET provides an alternative protocol to allow the service provider to justify its reason for the data collection and to entice the user with rewards. MUPPET presents the user with the reason and reward to be supplied by the service provider along with an analysis of the privacy concerns on the requested query. The user can examine all the evidence to make an informed decision as to whether to grant temporary access.

MUPPET logs the decision and allows the user to evaluate the level of satisfaction from the usefulness of the information exchange and the promised reward. This evaluation history can guide the user's next decision or be used to allow certain less critical privacy violations to be allowed in the future. This siginificantly increases the usability of the system.

Furthermore, a user may not be aware that certain queries are important to the decision algorithms and data mining models of service providers. Reward-driven information exchange gives a second-chance for the service provider to explicitly communicate with the user and address any privacy violation misunderstanding.

With our system model, information exchange can be seen as a bartering system between what information a user is comfortable disclosing and the reward from the disclosure. In some cases, given an appealing reward, a user may be willing to tradeoff some privacy, and our system does not prevent the user from doing so. The responsibility of MUPPET is to provide accurate and detailed information to assist the user in making such decisions. Figure 3 shows the steps in our reward-driven protocol.

C. Purpose Detection

The Purpose Detection engine should determine the set of active purposes that match closely with the user's intent. A reliable mechanism to specify this intent is to ask the user to communicate with MUPPET explicitly. To this end, a graphical user interface allows the user to check the purposes that he/she wishes to enable. These purposes are grouped using *Purpose Containers* that are descriptive identifiers for collections of purposes to be enabled simultaneously. Purpose containers are arranged in a hierarchical fashion for ease of use as shown in Figure 4.

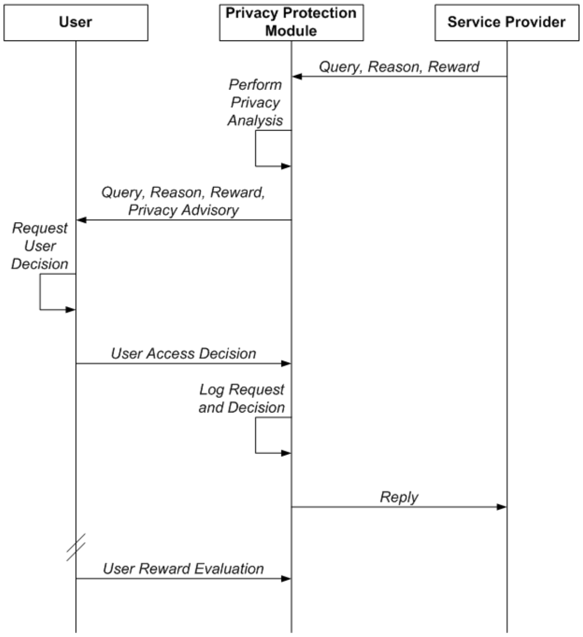
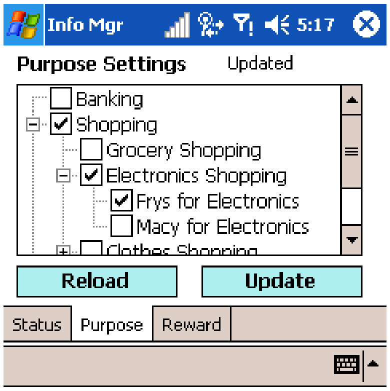


Fig. 3. Reward-Driven Protocol



Purpose Settings Updated

- □ Banking
- ☑ Shopping
  - □ Grocery Shopping
  - ☑ Electronics Shopping
    - ☑ Frys for Electronics
    - □ Macy for Electronics
  - □ Clothes Shopping

Fig. 4. Purpose Management User Interface

At the start of an information exchange session, the user is asked to confirm the current list. An explicit purpose is automatically activated if the user has enabled it. In addition, MUPPET provides two other implicit mechanisms for activating purposes. These implicit mechanisms do not reduce the user's control as they require the user to specify whether a particular implicit purpose is allowed to be activated by enabling it. Context information such as location and time can be used to activate recurring purposes such as weekly grocery shopping at the corner store.

A purpose can also be treated as a capability that the user grants a service provider. This capability, or authorization to activate a pre-determined set of purposes, is represented by an unguessable number. This unguessable number is chosen by the user and sent to the service provider during a prior interaction. Notice that authenticating the identity of the service provider is not always required. The user can use the physical environment (e.g. standing in an electronics store) to determine the expected behavior of the service provider. In subsequent interactions, this number provides all the needed authentication. The service provider can simply present this unguessable number to re-activate these purposes provided that they are enabled. The user can also revoke such a capability by simply ignoring the unguessable number or disabling the purpose. A given unguessable number may be tied to a specific service provider or it can be a shared value that is accepted from multiple service providers. The latter is useful for franchises where the user may share the same privacy concerns for a group of stores. This authorization-based concept [14] presents a number of attractive features for MUPPET. It allows arbitrary pre-defined policies to be associated with service providers. These may include policy templates designed by security experts. Authorizations are also easy to manage since the user can revoke them at anytime and policy updates can be reflected in new number-to-policy mapping maintained by the user device. They can also be transferred among service providers.

IV. Implementation and Evaluation

In this section, we discuss current status and implementation of MUPPET. To validate our system, we have integrated MUPPET with a Retail Kiosk environment under development at HP Labs and will elaborate on the experiences in specifying policies and the user interactions in two retail applications. Later in the section, we assess the expressiveness of our privacy framework by comparing MUPPET with other access control models. Finally, we show that our system is efficient for mobile settings with system performance evaluations.

A. MUPPET Prototype

We have implemented a version of MUPPET targeted for Pocket PCs. It consists of about 18000 lines of C# code using the Microsoft .NET Compact Framework. We have tested it on various versions of HP iPAQs including the HW6515A model running Microsoft Windows Mobile 2003. The prototype uses the embedded database SQL Server CE to store user data as well as the policies and metadata used in MUPPET. It has been demonstrated in a wireless network with multiple service providers. MUPPET communicates with service providers using the HP Labs C# ORB, a CORBA infrastructure for bi-directional object-based communication.

B. Application Deployment Experiences

As a proof-of-concept, we used MUPPET to protect a user's privacy in a retail store setting. Our system interacted with coupon offering applications running on kiosks. The retailer wishes to bring up discount coupons and product promotions that are of interest to the user. It does this by running a personalization engine that asks the user a number of questions to better understand the user's preferences and buying habits. Some of these questions may be deemed too sensitive for a user. MUPPET is used to express some of these privacy concerns and regulate the information exchange. Figure 5 shows the block-level interaction between MUPPET and the retailer.

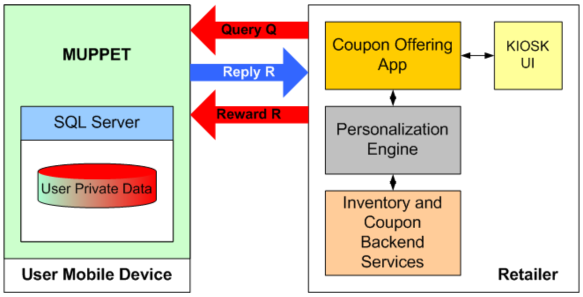


Fig. 5. Interaction between MUPPET and Retail Store Kiosk

In the following sub-sections, we discuss the deployment experiences of two different retail store examples: general grocery shopping and store-specific electronic shopping. We tried to consider privacy concerns from an end user's perspective and in most cases, we were able to translate it into policies enforced by MUPPET. There are a few cases that are not supported and often they are due to significant storage or performance cost inherent with the particular concern rather than insufficient expressiveness. For example, in the exceptional cases where a user is concerned about conjunctive release of various cell-level collections of data (Name of Products with Price $>$ \$10, Quantity of items in Category Clothing, Brands purchased in last 3 months) in a session, the query state must be kept or processed at each possible data collection. Our implementation currently prevents some types to be used at cell-level primarily for this performance reason.

In both of these examples, user identity is protected by restricting the combined disclosure of name, gender and ZIP code over a session. The age of user is revealed as a rough estimate. These policies are defined as follows using a purpose named *GeneralShopping* to encapsulate these concerns.

```
Attribute-Level Policy #1
attribute := PersonalInfo.Name (similarly
for Gender and ZIP)
label.purpose := GeneralShopping
label.type := CONJUNCTIVE("PersonalInfo.Name,
PersonalInfo.Gender, PersonalInfo.ZIP")
```

```
Attribute-Level Policy #2
attribute := PersonalInfo.Age
label.purpose := GeneralShopping
label.type := QUANTIZATION(5)
```

1) General Grocery Shopping: The first example mimics the consumer experience of walking into a neighbourhood grocery store to purchase various items. The user enters the store with a shopping list along with shopping history and personal information on the PDA.

For some users, portions of their shopping list act as a personal reminder not intended to be shared with the store. For example, a user may be embarassed about certain hygiene products that are on the shopping list. With MUPPET, a user can easily filter out these items. A user may also hide categories of information such as alcoholic beverage purchasing history. These two concerns are captured in the following cell-level policies. The product names of such items are protected.

```
Cell-Level Policy #1
tuple := (ShoppingList.ProductName !=
'Wart Remover')
attribute := ShoppingList.ProductName
label.purpose := GroceryShopping
label.type := BLOCKING[1]
```

```
Cell-Level Policy #2
tuple := (ShoppingHistory.Category !=
'Alcoholic Beverage')
attribute := ShoppingHistory.ProductName
label.purpose := GroceryShopping
label.type := BLOCKING[1]
```

2) Store-specific Electronics Shopping: MUPPET can also be used to protect store-specific data as shown in this example. The user's shopping history is a compilation of all purchases made at different stores. Historical purchasing patterns can be extracted from these data. However, in a competitive market, a user may not want to reveal shopping behavior and purchase prices from other similar stores as it can potentially affect an offer price in dynamic pricing models.

Consumer Fred has previously purchased from Alice's Electronics and Bob's Electronics. In addition to *GeneralShopping*, there are also extra policies that can be activated by the *ElectronicsShoppingAtAlice* and *ElectronicsShoppingAtBob* purposes, respectively.

Fred enters Bob's Electronics to check out the latest electronics gizmos. He approaches an advertisement display kiosk and the kiosk presents Fred's smartphone with a pre-negotiated unguessable number authorizing the activation of the *ElectronicsShoppingAtBob* purpose. (Alternatively, this can also be done using context-rule purpose activation.) Some of Fred's concerns in this scenario include limiting disclosure to relevant items and store-specific shopping history. For example, he may want a policy to reveal only electronic items on his shopping list. This can be expressed in the first policy below. As mentioned, Fred may only be willing to disclose shopping history at Bob's Electronics while shopping at this store. This is expressed in the second policy. Moreover, he does not want to reveal exact purchase prices and allows only total prices to be communicated. These are only a small subset for illustrative purposes.

```
Cell-Level Policy #1
tuple := (ShoppingList.Category =
'Electronics')
attribute := ShoppingList.ProductName
(similarly for other attributes)
label.purpose := ElectronicsShoppingAtBob
label.type := BLOCKING[1]
```

```
Cell-Level Policy #2
tuple := (ShoppingHistory.StoreID = Bob's
Electronics Store ID)
attribute := ShoppingHistory.ProductName
(similar for other attributes)
label.purpose := ElectronicsShoppingAtBob
label.type := BLOCKING[1]
```

```
Attribute-Level Policy #3
attribute := ShoppingHistory.Price
label.purpose := ElectronicsShoppingAtBob
label.type := AGGREGATE(SUM)
```

Our system does not permanently exclude the exchange of these restricted data items. For example, it is possible for the kiosk to find out Fred's shopping history at Alice's Electronics. However, this must be done via explicit consent provided by the user through the mobile device. The kiosk can send a reason and a reward along with the query to entice Fred. The reason provides justifications for data-use and the reward may be an explanation of what Fred can gain by disclosing his shopping history at Alice's Electronics (eg. guaranteed-lowest-price campaigns) or a redeemable discount coupon on a popular sale item. MUPPET provides a venue for this explicit communication and advises the user of the privacy risks. The final decision resides with the user.

C. Expressiveness

The Operation-focused Access Control in MUPPET assumes arbitrary composition of operation labels on a data object. In this way, it can support not only general purpose-driven access but also a variety of other access control models.

1) Compartmentalization: Many access control policies relate to separating data into compartments and granting requestors access rights to different compartments. This is evident in the shopping list examples shown earlier where only parts of the shopping list are accessible depending on the purpose. The ability to compartmentalize is important in building systems that can support the need-to-know principle. With operation labels, compartments can be identified by labels with

[1] For BLOCKING type at cell-level, the *tuple* field specifies the tuples that can access the attribute.

different purposes, where rights to access a compartment is equivalent to activating the associated purpose.

2) Role-based Access Control and Identity-based Access Control: Role-based Access Control (RBAC) and Identity-based Access Control (IBAC) are commonly used to enforce privilege separation. Depending on the group association, role or identity of the requestor, access is restricted accordingly. While these schemes typically require an orthogonal identity or trust management component, the basic policy specification can be supported using operation labels. These labels can be used to differentiate the lowest granularity of access, for example, by assigning a different label to each identity. They can also be combined to represent logical groupings of permissions.

3) Capability and Authorization-based Access Control: With fine-grain labels to data objects, the ability to activate these labels can be seen as a capability as demonstrated in our authorization-based purpose activation support. These labels can potentially support more complex capabilities and authorizations as new sensitivity types are added.

D. Performance

Performance is not a crucial factor for the MUPPET environment as most of the latencies in query processing can be hidden by other user interactions on the kiosks and service access points. Nonetheless, for completeness, we note that without optimization the average response time was 2 seconds using about 10 MB of memory. These measurements were taken on the HP iPAQ HW6515A with 312 Mhz Intel XScale Processor and 64 MB meomry. The memory usage shows the run-time allocation needed and the average response time is taken over different types of queries measuring the query processing time as well as the messaging overhead and the local area wireless network round-trip.

The storage and time efficiency of MUPPET is adversely impacted by the absence of various important features in Microsoft SQL CE Server. For example, the lack of logical database views implies that temporary tables must be created in many cases before applying data transformations. Even with these workarounds and non-optimized code, the average response time is below the 10 second tolerance [18] needed to keep a user focused on a task.

V. Related Work

For mobile environments, a number of research projects ([7], [17], [19], [6]) have utilized context information as guiding parameters in making access decisions. The Cerberus system described in [6] addresses security issues in smart spaces, a pervasive computing environment of interacting sensors and devices. Their security policies involve context information such as location and time specified as boolean expressions. Neumann and Strembeck [19] demonstrated that context constraints can be implemented as additional dynamic RBAC (role-based access control) attributes. In [7], the authors introduced GEO-RBAC, an extension to traditional role-based access control to incorporate spatial and more detailed location-based information. In our design, we acknowlege the importance and usefulness of context information in access control and MUPPET supports context rules in determining the active purpose. However, our system deviates from the role-based access control model and hence, is not hindered by the many drawbacks of RBAC such as the need for identity management and its limited expressiveness.

There are also systems that have focused on other aspects of privacy in ubiquitous environments. Langheinrich[16] devised a privacy-awareness system for ubiquitous computing environments. His system provided a secure messaging infrastructure for exchanging and negotiating privacy policies. Hong [12] performed extensive studies in his thesis to understand users' privacy concerns and designed a system to capture these concerns in XML format. His work is more closely related with policy authorization languages focusing primarily on the representation problem.

There have been a number of emerging standards on policy authorization languages. The World Wide Web Consortium is advocating P3P (Platform for Privacy Preferences) [11] as the standard format for expressing data-collection and data-use practices on the web. IBM's EPAL [21] is targeted for enterprise business-to-business applications and allows the specification of privacy policies in XML so that information can be protected and used in accordance with the responsible organization's privacy policies. MUPPET and P3P both provide data-centric policy specification. However, P3P policies are not machine-enforceable and are subject to different interpretations by different interacting user agents. On the other hand, EPAL is designed to be enforceable but in reality, obligations such as limited disclosure and retention are impossible to enforce without trusted third parties and trust management. User mobile devices can interact with a large number of service providers as the user roams around. Hence, it may not be possible to assume that trust can be easily established or managed.

Access control has been an important part of database systems. Earlier systems ([10], [20], [13], [22]) are primarily role-based and view-based in which depending on the user's security level, a subset of the database becomes accessible. More recently, statistical databases ([5], [3]) are a class of database systems aimed at privacy-preserving data-mining. The goal is to develop accurate models from collected data in public databases without access to precise information in individual data records. Results from this area have shown that this is possible even when fields are masked out or perturbed. MUPPET leverages this finding. Sensitivity types can be used as a generic mechanism in specifying data transformations and perturbations for private databases. Statistical databases focus mostly on hiding the anonymity of many users whereas MUPPET is more interested in protecting various kinds of sensitive data for a particular individual. Current private databases have limited privacy protection. In [4], Agrawal urged future database systems to be more privacy-aware and proposed the use of purpose as the central concept for designing these systems. He gave a strawman system where each attribute of a database table has additional information associated with it describing the purpose, list of external recipients and the retention period of the data to be enforced by the internal database engine. Later in [9], Byun et al extended an existing role-based access control mechanism with this purpose concept. However, purpose behaves very much like roles, and their system has the same disadvantages as other RBAC systems. Our system draws on the seminal work in purpose-based access control ([4], [9]) but is a significant step forward and addresses many of the complexities in designing a flexible working system for mobile applications.

VI. Conclusion and Future Work

MUPPET is an information brokerage framework that gives users control over the release of their data. It introduces *Operation-focused Access Control* which uses typed operation labels as an expressive mechanism to annotate privacy concerns. In this paper, we have presented the central concepts of this purpose-based protection and have shown that it can also be used to specify other common access control models. MUPPET also includes three ways to activate purposes. These include both explicit and implicit mechanisms to benefit from the rich user interaction and contextual information in mobile environments. Authorization-based purpose activation adds another dimension in supporting pre-negotiated access rights by considering the ability to activate certain purposes as a capability granted to these service providers. Finally, MUPPET exposes a reward-driven protocol where service providers can communicate reward information to the user and users can make informed access decisions.

We have shown that MUPPET is useful and practical through the integration of our prototype with retail kiosk applications. Extensions to our work include developing tools that make it easier for users and adminstrators to specify these policies and to understand the result of multiple overlapping operation labels. We would also like to automatically infer a user's privacy concerns and dynamically propose policies that may be suitable. Moreover, typed operation labels offer a fine-grain and descriptive way to tag data objects. We are also looking at techniques to extend this in describing information flow where sensitive data may pass through multiple parties.

Acknowledgment

The authors would like to thank members of the RAPID project within the Digital Printing and Imaging Lab at Hewlett-Packard Laboratories for their suggestions and help in the integration with the retail kiosk environment. Also, Mark S. Miller has provided insightful feedback on the work. Alex Zhang and Alex Koslov provided expert advices on the design of the personalization engine. Marco Cassassa Mont at HP Labs Bristol pointed us to various related work and made suggestions for future extensions.

References

[1] (2006, June) Mine data not details. Wired Magazine.

[2] L. Aaronson. (2006, July) Your cell phone is so money. Popular Science.

[3] D. Agrawal and C. Aggarwal, "On the design and quantification of privacy preserving data-mining algorithms," in *Symposium on Principles of Database Systems*, 2001.

[4] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, "Hippocratic databases," in *Proc. 28th International Conference on Very Large Databases*, Hong Kong, China, August 2002.

[5] R. Agrawal and R. Srikant, "Privacy-preserving data mining," in *Proceedings of the ACM SIGMOD Conference on Management of Data*, 2000, pp. 439–450.

[6] J. Al-Muhtadi, A. Ranganathan, R. H. Campbell, and M. D. Mickunas, "Cerberus: A context-aware security scheme for smart spaces." in *PerCom*, 2003.

[7] E. Bertino, B. Catania, M. L. Damiani, and P. Perlasca, "Geo-rbac: a spatially aware rbac." in *SACMAT*, 2005, pp. 29–37.

[8] M. Bosworth. (2005, July) Loyalty cards: Reward or threat. ConsumerAffairs.com.

[9] J.-W. Byun, E. Bertino, and N. Li, "Purpose based access control of complex data for privacy protection." in *SACMAT*, 2005, pp. 102–110.

[10] J. C. D. Ferraiolo and R. Kuhn, "Role-based access control (rbac): Features and motivations," in *Proceedings of the 11th Annual Computer Security Application Conference*, New Orleans, LA, USA, 1995, pp. 241–248.

[11] L. C. et al. (2005) The platform for privacy preferences 1.1 (p3p 1.1) specification. http://www. w3.org/TR/2005/WD-P3P11-20050104/Overview.html.

[12] J. I.-A. Hong. (2005) An architecture for privacy-sensitive ubiquitous computing. PhD Thesis, University of California at Berkeley, Computer Science Division.

[13] Informix. (1997) Informix online dynamics server 7.2 - adminstrators's guide.

[14] A. H. Karp, "Authorization based access control for the services oriented architecture," in *Proceedings of the 4th Conference on Creating, Connecting and Collaborating through Computing*, Berkeley, CA, 2006.

[15] A. T. Kearney. (2005) Study finds mobile phone user embracing mobile data services. http://www.atkearney.com/main.taf?p=1,5,1,167.

[16] M. Langheinrich, "A privacy awareness system for ubiquitous computing environments," in *UbiComp '02: Proceedings of the 4th international conference on Ubiquitous Computing*. London, UK: Springer-Verlag, 2002, pp. 237–245.

[17] W. S. M. Fahrmair and B. Spanfelner, "Security and privacy rights management for mobile and ubiquitous computing," in *Workshop on UbiComp Privacy*, Tokyo, Japan, September 2005.

[18] R. B. Miller, "Response time in man-computer conversational transactions," in *Proceedings AFIPS Fall Joint Computer Conference*, 1968, pp. 267–277.

[19] G. Neumann and M. Strembeck, "An approach to engineer and enforce context constraints in an rbac environment," in *SACMAT '03: Proceedings of the eighth ACM symposium on Access control models and technologies*. New York, NY, USA: ACM Press, 2003, pp. 65–79.

[20] Oracle. (1997) Oracle 8 enterprise edition - server administration's guide.

[21] G. K. P. Ashley, S. Hada and M. Schunter. (2003) Enterprise privacy authorization language (epal) 1.1 specification.

[22] Sybase. (1997) Sybase adaptive server enterprise security adminstration.