Personal Data Vaults: A Locus of Control for Personal Data Streams

Min Mun, Shuai Hao*, Nilesh Mishra*, Katie Shilton,
Jeff Burke, Deborah Estrin, Mark Hansen, Ramesh Govindan*
Center for Embedded Networked Sensing
University of Caliornia, Los Angeles
{bobbymun,destrin}@cs.ucla.edu, kshilton@ucla.edu, jburke@remap.ucla.edu,
cocteau@stat.ucla.edu
Embedded Networks Laboratory*
University of Southern California
{shuaihao, nmishra,ramesh}@usc.edu

ABSTRACT

The increasing ubiquity of the mobile phone is creating many opportunities for personal context sensing, and will result in massive databases of individuals' sensitive information incorporating locations, movements, images, text annotations, and even health data. In existing system architectures, users upload their raw (unprocessed or filtered) data streams directly to content-service providers and have little control over their data once they "opt-in".

We present *Personal Data Vaults* (PDVs), a privacy architecture in which individuals retain ownership of their data. Data are routinely filtered before being shared with content-service providers, and users or data custodian services can participate in making controlled data-sharing decisions. Introducing a PDV gives users flexible and granular access control over data. To reduce the burden on users and improve usability, we explore three mechanisms for managing data policies: *Granular ACL*, *Trace-audit* and *Rule Recommender*. We have implemented a proof-of-concept PDV and evaluated it using real data traces collected from two personal participatory sensing applications.

Categories and Subject Descriptors

H.3.4 [**Systems and Software**]: current awareness systems (selective dissemination of information)

ACM CoNEXT 2010, November 30 – December 3 2010, Philadelphia, USA.

General Terms

Design, experimentation, management

Keywords

Personal participatory sensing, privacy policy

1. INTRODUCTION

Mobile smartphones have created new and disruptive information flows [28]. These phones have capabilities to record and transmit our location, take images or videos, track our physical activities to observe personal behaviors, share data with communities, or compute community statistics. These emerging *participatory sensing* [9] applications raise important and challenging questions whose answers can potentially affect network architecture [29]. In this paper, we focus on a sub-class of these applications in which individuals use mobile devices to collect personal data streams, which are used for observing and adjusting personal behavior, and as a tool for clinicians to judge the efficacy of their treatments. We call this class of applications *personal participatory sensing*, examples of which include PEIR [27], CenceMe [14], Micro-Blog [18], Common Sense [13] and MyExperience [16]. These applications are clearly privacy-sensitive: by itself, these data streams can reveal interesting user behaviors; in addition, when combined with external data sources or models and with the data streams of other users, these data streams can be used to infer highly private information about users, their preferences and proclivities.

In existing system architectures, users have little control over what information is being shared and must trust the service providers to protect their highly individualized and sensitive data. In addition, subtle changes in the granularity at which the data is shared could have significant privacy consequences. For in-

stance, sharing the zip code instead of the exact location may provide a layer of privacy, but even this may not be enough for some users if the places they visit each correspond to a distinct zip code.

The centerpiece of our approach to privacy for personal participatory sensing data is individually-controlled secure data repositories which we call *Personal Data Vaults (PDVs)*. The PDV decouples the capture and archival of personal data streams from the sharing of that information. Instead of individuals sharing their personal data streams directly with services, we propose the use of secure containers to which only the individual has complete access. The PDV facilitates the selective sharing of subsets of this information with various types of Access Control Lists (ACLs). Rather than relying on third-parties to correctly control the sharing, we argue that the owner of the data can actively participate in making controlled data sharing decisions.

Similar architectures [33], [11], [7], [31] have been introduced for location-based online social networks. We build on this earlier work to create a complementary platform for exploring the functional performance and legal implications of supporting privacy preserving personal participatory sensing applications using PDVs. Our system offers three significant privacy benefits over prior work. First, using *Granular ACLs*, our system provides users with fine-grained control on data. Users (or trusted guardians on behalf of users) can define not only who has access to what data but also the granularity of data for sharing, which gives users more flexibility in expressing sharing decisions. Second, *Trace-audit* logs and displays transactions and transformations of users' data and enables users to track who visibly viewed their data, how frequently they viewed the data, and how the data is used. It also gives users easy-to-read reports about continuous data sharing by visualizing the consequences of data sharing, using currently and previously shared information. Users can be alerted to update their sharing decisions when their sharing policies may have been judged by the system to be different from their intention; alternatively, users can periodically review these reports in a manner analogous to reviewing credit reports. Third, *Rule Recommender* pre-calculates constraint values for a set of pre-defined privacy policies, makes the computed constraints available for display, and facilitates the policy re-configuration.

Our architecture can also be used for grassroots participatory sensing applications, where groups of participants use privately-owned mobile devices to collectively measure aggregate phenomena of mutual interest, by adding popular privacy preserving techniques like data perturbation [17] and anonymity. However, in this paper, we limit ourselves to building a privacy architecture for personal participatory sensing, in which maintaining personal identities and trends is essential. We have implemented a proof-of-concept PDV and demonstrate how it helps users understand the implications of different privacy choices and informs decisions about data sharing. Our evaluation also uses real data traces collected from two personal participatory sensing applications. However, much work remains, including system and usability evaluation with a larger number and variety of users and use cases.

The rest of this paper is organized as follows. Section 2 introduces two personal participatory sensing applications. In Section 3, we describe the system design principles. In Section 4, we introduce the overall system architecture and detail the three mechanisms. In Section 5 and 6, we discuss implementation details and application experiences. Evaluation and related work are presented in Section 7 and 8. Finally, we conclude the paper and discuss directions for future research in Section 9.

2. APPLICATIONS

To guide the development of the architecture, we have used two personal participatory sensing applications in public health that are representative of other personal participatory sensing applications. These applications, which rely on mobile-phone based sensors for health monitoring, are based on a system called AndWellness [20] to improve personalized health care through mobile self-monitoring using Ecological Momentary Assessment (EMA) data. Although AndWellness can be configured to perform various kinds of self-monitoring, we use two example applications throughout this paper: *Ambulation* and *Waking-Survey*.

Ambulation [30] is a tool for monitoring mobility patterns over time. The application collects user location, accelerometer and time data, and automatically detects the user's mobility modes (stationary, walking, or motorized) every 30 seconds. The phone uploads the collected mobility and location information to a service provider.

In **Waking-survey** [20], users are asked to respond to the following five questions each day when they wake up: What time did you go to bed last night? How long did it take to fall asleep last night? What time did you wake up this morning? How many hours of actual sleep did you get? How would you rate your sleep quality? The responses are automatically tagged with the time, date, and location data, and uploaded to the web server. In addition, users can opt in to collect continuous location and activity data.

For both applications, intuitive web-based visualization of the data is available to the user and his or her family, and friends or caregivers authorized by the user, allowing them to identify trends in the user's mobility or sleeping patterns, or to measure progress over time

in response to varying treatments.

3. SYSTEM DESIGN PRINCIPLES

An architecture for users to manage their own sharing decisions must satisfy several requirements. First, it should provide users with sufficient control over data sharing so that they do not reveal information more than necessary about themselves. Second, it should allow users to make and alter data sharing decisions over time, as the context of their privacy needs change. Third, it should help users interpret their data, so that they can make informed decisions about what risks or benefits revealing such data might entail. Finally, it should encourage individuals' investment in their data, giving them reasons to explore and review privacy decisions.

These considerations lead us to three design principles: *participant primacy, data legibility, and longitudinal engagement*. These principles have been derived from a legal framework for personal privacy [32].

- **Participant primacy** stipulates that users should retain control over their raw data and should be able to make decisions to share subsets of the data. Participant primacy does not preclude third parties from keeping copies of the data, if permitted by the owner. Ownership of the data need not imply physical ownership of the medium on which the data is stored; ownership can be assured by a third-party storage service.
- **Data legibility** dictates that the system must provide high-level tools and guidance on the implications of users' decisions about their data. Data legibility allows users to understand the complex risks of their participation and helps them make better decisions about data capture, sharing, and retention. Stronger interpretation of data usage can fortify participants as better data stewards.
- **Long-term engagement** specifies that the system should encourage the continued engagement of users. This allows users to check to see if their data are still visible and relevant, and make continuing, ongoing decisions about their sharing policies. These decisions can be triggered by new applications that generate new kinds of data, or by failures in occasional or periodic verification of agreements with third parties.

4. PERSONAL DATA VAULTS

The PDV is a personal data store that supports personal data ownership, selective sharing, and audit mechanisms to provide visibility into shared-data handling. Our system relies on two assumptions. The first is that each data owner has a logically distinct PDV, and the PDV is trusted. The second assumption is that when an owner shares data with another entity, there is an implicit or explicit legally-enforceable agreement about how the entity will use the received data.

An important challenge in PDV design, and one this paper addresses, is to devise mechanisms that embody the design principles discussed above. These design principles require that individual decision-making about controlled data sharing not become too complicated and time consuming since that would render the system unusable. To achieve the design principles, we place PDV between a user and content-service providers as shown in Figure 1. The PDV incorporates three mechanisms for managing data policies: Granular ACLs, Trace-audit and Rule Recommender.

Sharing data indirectly through PDV has many advantages over using centralized third-party data storage [11], [31]. The advantages include: 1) allowing users to own personal data, 2) allowing content-service providers to access data with the permission of the user, 3) reducing the possibility of data redundancy, and 4) reducing the chance of data loss issues and resource limitations by storing data off the mobile devices. Figure 1 shows a system use scenario: how data are filtered before being shared using Granular ACLs, and when and how users can re-configure the privacy policies using the Trace-audit and Rule Recommender. It also illustrates sample user interfaces for the Trace-audit and Rule Recommender.

In this section, we discuss in detail the three elements of the system: Granular ACL, Trace-audit, and Rule Recommender.

4.1 Granular Access Control Lists (ACLs)

Traditional ACLs specify which users or system processes are granted access to data as well as what operations are allowed on given data. ACLs for online social networks have focused on managing groups, who has access to what data, and paid less attention to controlling the level of data for sharing. In contrast, we try to emphasize the importance of controlling the granularity of data for sharing. Unlike online social networks, personal participatory sensing data are quite granular (e.g., thousands of GPS points or accelerometer readings) and range from location values to many application-specific values (e.g., sleep duration, wake-up time for Waking-survey), and the data are often processed through external and cross-user data sources, models, and algorithms to be used to infer complex phenomena about individuals. Subtle changes to the granularity of data for sharing could result in very different consequences to users due to different processes on the application side and users' implicit tolerance for privacy. Therefore, it is very important to give users full control over data, at which granularity data are shared as well as who access

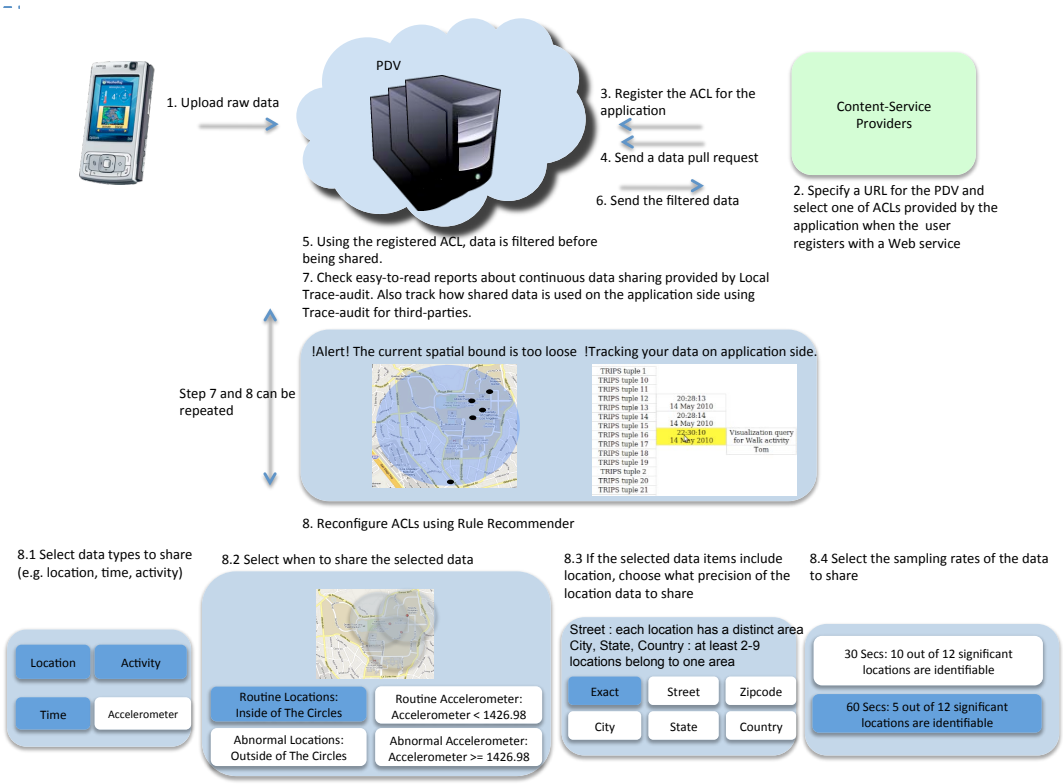

Figure 1: System usage scenario

Rule Implication	If the application named Ambulation queries, share the exact location when the user's in Westwood (within 1.5 km of the GPS coordinates of (34.06,-118.44)), otherwise, share location at a zip code level
ACL Representation	{"entity": {"type":"application","name":"ambulation"},"filters": [{"bound": [{"type":"location","format":"in-circle","center": {"latitude":34.06,"longitude":-118.44} ,"radius":1.5}],"precision": [{"type":"location","value":"exact"}] }, {"bound": [{"type": "location","format":"out-circle","center":{"latitude":34.06,"longitude":-118.44}, "radius":1.5}] , "precision": [{"type":"location","value":"zipcode"}] }] }

Table 1: An ACL example

Constraint	Type	Attributes
Bound	time	starttime, endtime
	location	format(in-circle,out-circle), center(GPS coordinates), radius(in km)
	number	lower, lowersymbol(=, <,<=), upper, uppersymbol(=,>,>=)
	text	attrname, text, symbol(=,!=)
Precision	time	value(private, second, minute, hour)
	location	value(private, exact, street, zipcode, state, country)
	number	value(private, average), timeframe(mintue, hour, day, week, month)
Frequency	time	unit(second, minute), value

Table 2: Types and attributes of the ACL constraints

to what data.

A Granular ACL contains an *entity* and a set of *filters*. The entity specifies the type and name of third parties accessing data from the PDV. The filter has a list of constraints which define data to be shared: Each constraint is described by type and its attributes (see Table 2 for more details). To be very specific, we use the JavaScript Object Notation (JSON) to format the ACLs (see Table 1)). When a third party, such as a content-service provider, requests data at the PDV, the PDV first determines the filters associated with the applicable entry to decide what granularity of data to send. Our prototype PDV currently supports the following three constraints (we intend to add more constraints for common use cases as they emerge).

Bounds. This constraint decides a subset of data to be shared with third parties by limiting data space. Users can limit a data space for sharing for a particular data collection by time interval(s), spatial location(s), numeric range(s) and text(s). Thus, for example, one can allow co-workers access to data annotated by "walking" and collected in the Westwood area between 9am and 5pm for Ambulation.

Precision. The constraint governs the precision of the time, location or numeric value reported to the user, and is motivated by a user study that suggests users would like control on the precision of information shared with others [23]. Thus, an owner may allow a friend to access timestamps that are accurate up to a minute and also allow access to precise locations, but only allow acquaintances access to timestamps accurate to an hour and location accurate to a street, a zipcode or a city. Other precision controls, such as averaging values over a fixed time window, are also possible. For instance, for Waking-survey, a user can share only the averaged sleep duration over the week with her physician. Or one can decide not to share data at all by setting it as private.

Frequency. Finally, the privacy risk decreases when the system operates with lower sampling frequency [22] and this constraint ensures that the data shared with the user does not exceed a specified temporal frequency. Many activity recognition or locational path reconstruction algorithms rely on the frequency of location information or other cues (for example, it may not be possible to accurately estimate highway driving trajectories with one location sample every 10 mins), and this constraint attempts to provide data owners with some control on the such use of the data.

4.2 Trace-audit

Our prototype PDV system permits two types of Trace-audit mechanisms: *local Trace-audit* to log operations performed inside PDV and *Trace-audit for third-party applications* to track actions occurring outside PDV (i.e. inside third-party applications that read data from the PDV).

As part of the management modules running inside PDV, the local Trace-audit logs read accesses to users' data residing in the data store. Each log entry is represented as the following tuple: `<timestamp, appId, opType, dataTable, dataField1, dataField2, ..., startRow, endRow>`, with the interpretation: `appId` application performed `opType` operation on data table `dataTable` at `timestamp` and the affected data fields were `dataField1, dataField2, ...` starting from row `startRow` to `endRow`. For instance, when Ambulation reads "latitude, longitude, timestamp, activity" data from table "Mobility" at 2010-06-24-11:22:33, the log entry will look like `<2010-06-24-11:22:33, Ambulation, READ, Mobility, latitude, longitude, timestamp, activity, 1, 2000>`. The generated logging information is then visualized and presented to PDV owners to help interpret what data have been shared with which applications, given the current set of sharing policies (Figure 2).

The Trace-audit module for third party applications provides visibility on what happens with the user's data, obtained from their PDVs, inside an application. Like the local Trace-audit, an application implementing the Trace-audit module logs the data tuples accessed, the operation performed along with the entity on whose behalf the data were accessed. We envision that despite varied semantics associated with the operations performed on the data all applications can log against each data operation in the similar fashion to the local Trace-audit: `<timestamp, userId, opType, dataTable, tupleRange>`. Here `timestamp` is the timestamp when the query or operation is performed, `userId` is the user id on whose behalf the operation was performed, `opType` is the data query or operation, and the `tupleRange` is the tuple ranges accessed for the operation. For example, when an application reads raw GPS traces from id number 10500 to 11000 in GPS_RAW table at 2010-05-14 20:28:14, caculates speed values and updates data from id number 2300 to 2380 in TRIPS table at 2010-05-14 20:28:14. The logs look like the following: `<timestamp:2010-05-14 20:28:14, userId:System, opType:Data read for speed calculation, dataTable:GPS_RAW, tupleRange:[start:10500, end:11000]>, <timestamp:2010-05-14 20:28:14, userId:System, opType:Speed values added, dataTable:TRIPS, tupleRange:[start:2300, end:2380]>`. This granular data log is made available to the data owners via a standard API through which the data owner's PDV can pull the trace audit data from the applications for storage, visualization and analysis at the PDV. The logs are visualized as a timeline which provides all access times, the operations performed and the entity on whose behalf the operation was performed on a data tuple (see Figure 1). Our prototype PDV's trace audit

mechanism for third party applications currently provides data access and usage logs for traceability, but is limited to data shared by the PDV directly with third-party applications, and does not track data derived by the applications themselves. Extending this design to a richer information flow traceability, especially for derived data, is part of the future work.

4.3 Rule Recommender

The Rule Recommender provides a high-level interface for setting sharing policies. Specifically, it precomputes constraint values (ACLs) for a set of common high-level user intentions (e.g., share data when I'm at my usual/routine locations). These constraint values are derived from historical data and the Rule recommender makes the calculated values available to users (or trusted guardians on behalf of users) as shown in Figure 1, which facilitates the reconfiguration of ACLs. Thus, by using the Rule Recommender, users can more actively participate in controlled data sharing, and the on-going experience with the system will give users a better insight into the implication of privacy policies. Using the Rule Recommender alone is not enough, since the suggested policies may actually not match user intent over time; users will need to recompute the constraints either periodically, or in response to a Trace-audit analysis.

Many other privacy policies may emerge to describe users' privacy preferences. In this paper, we try to capture users' intentions behind privacy policies expressed in terms of the three constraints, bound, precision and frequency, from experience with many personal participatory sensing applications. We intend to include more privacy policies for common use cases as we incorporate more constraints. In this section, we explain how each of the three constraints is computed for pre-defined privacy policies.

4.3.1 Computing the Bound

Users may consider a number of factors to define the bound. For example, some users may wish not to reveal their location when they are at home or engaged in certain types of private or semiprivate activities such as going out on a date or visiting hospitals [12]. Others may wish to share data that are annotated with particular words (e.g., photos annotated with Halloween) [31]. This paper, however, focuses on cases in which users wish to share data of their regular/routine patterns or exceptional/abnormal behavior. A routine or anomalous behavior may be defined by any contextual information provided by users. We deal with location or numeric application-specific data like sleep quality for Waking-survey. The system performs different processes depending on the types of contextual information, location or application-specific information, that are used to define routines or anomaly. First, the following describes the steps required to calculate the bound value when location information defines a routine or anomaly:

- **Significant Location Identification:** The system finds places where users spent a continuous amount of time (15 minutes) within a certain distance (50 meters) and considers this a "stay" [25]. Stays within a distance (250 meters) are clustered into "significant locations" using density based clustering [34]. Time and distance thresholds were chosen based on varying the parameter combinations and analyzing which were most effective at presenting users' notions of significant locations.
- **Routine Place Selection:** Once everyday significant locations are identified, the system judges whether the identified location is routine or not. If the number of days of visiting the location per week is greater than or equal to two, the place is considered as a routine place. This threshold value was the most effective at presenting users' routine places in our small pilots and would likely be adapted in the future.
- **Spatial Bound Creation:** By default, spatial bounds are created using a 1km radius with each of routine places found as a center point. A user should be able to adjust the size of the spatial bound through a graphical user interface because different users want different sizes of spatial boundaries. The same user may also define different sizes of spatial boundaries for different applications (e.g., larger boundaries for Ambulation to track walking activities around routine places, smaller circles for Waking-survey to track sleep durations and wake up time at the places). The created spatial bound specifies inclusions of location areas for the rule, a user shares data when she is in routine locations, otherwise, it specifies exclusions of location areas.

Second, when numeric application-specific information determines a routine or anomaly, the system computes the bound value using Local Outlier Factor (LOF) [8]. LOF considers data points that have substantially lower density than their neighbors to be outliers. Using the local density approach, LOF is able to identify outliers in a data set that would not be outliers in another area of data set. This technique is effective to our system because using absolute values to define anomalies across different users and applications is dangerous. For instance, for the Waking-survey, a 60 minute difference from the average wake-up time can be normal for some users having dynamic schedules, while even a 15 minutes difference could be abnormal for others having

relatively fixed schedules. The created numeric bound specifies inclusions of numeric values for the rule, a user shares data when her exceptional behavior is found, otherwise, it specifies exclusions of numeric ranges.

4.3.2 Computing the Precision

Although our system supports different levels of data sharing for various data types like location, time and numeric values, in this paper we concentrate on helping users define the precision of location data to reveal. People may want to share location data at coarser levels like street, city, state and country so that their current or visited locations are not identifiable. However, users sharing decisions may not accurately represent their intentions, since they may not fully understand the privacy implications of their decision. The same decision could have different implications depending on users' mobility patterns. For instance, one could choose to share one's location at zip code level with friends, but there may be few places she goes to within a certain zip code area and consequently, her friends could guess her location. Our system investigates the implication of sharing location data at different aggregation levels by calculating how many significant locations belong to aggregated regions. To do this, the following steps are performed.

- **Significant Location Identification:** The same significant location identification method used for computing the bound is applied.
- **Location Tree Construction:** We adopt a tree structure to explore the implications of sharing location data at different aggregation levels. Figure 4(left) shows an example tree. The top level represents countries, followed by states, cities, zip codes, streets and places (significant locations). In this example, one visited Location 1 in '90006', in the city of 'Los Angeles', in the state of 'CA', in the United States.
- **Location Tree Traversal:** Next, the system walks through the tree and probes the lowest number of leaf nodes for each of aggregation levels. If a user wants at least x significant locations belonging to aggregated regions, aggregation levels having at least x leaf nodes or higher should be selected.

4.3.3 Computing the Frequency

There are different kinds of information such as activity, locational path, significant locations that rely on the frequency of location information. One type of common privacy attack is to infer the coordinates of significant places like home or work-place [26]. Hoh et al. [21] showed that reducing the GPS sampling frequency from one minute to four minutes reduced the home identification rate from 85% to 40%. Similarly, we focus on computing the frequency that could foil this privacy attack by adjusting sampling frequency. The following steps are performed:

- **Significant Location Identification:** The system identifies the ground truth significant locations from the raw GPS traces. We employ the same algorithm used for generating the bound. It reverse-geocodes the GPS coordinates of the locations to convert GPS coordinates to physical address using [1].
- **Significant Location Identification Rate Calculation:** Next, we measure the significant location identification rate (how many significant locations are correctly detected) by decreasing the sampling rate by 30 seconds. We measure the significant location identification rates by comparing the significant locations found from altered data and the ground-truth significant locations.

4.4 PDV Attacks

The PDV architecture mitigates privacy violations through a combination of legal, economic, and technical means. To understand this, we first define an attack to be successful if a user's data stream (or part thereof) is made available to an entity that is expressly denied, by the user's policies, access to the data.

This unauthorized access to data can happen in one of several ways: when an attacker eavesdrops on data either when it is being uploaded to the PDV, or accessed by a third party application from the PDV; when the PDV itself is compromised; or when a third-party application leaks the user's data to an unauthorized entity.

Eavesdropping can be generally guarded against by appropriately encrypting upload and access connections (e.g., by using HTTPS). The PDV can be protected using firewalls, access controls, and other technical security methods. Of course, the vulnerability of the PDV depends largely on the choice of technologies. We envision the emergence of a market of PDV operators who are incentivized by competition to provide legal assurances of data privacy, and who will then provide adequate levels of security to match those legal assurances. In a similar way, we expect that content service providers will also be incentivized to differentiate themselves from the competition by offering legal agreements on acceptable use, and by allowing access to audit logs for compliance monitoring.

5. IMPLEMENTATION

We have developed a prototype of the PDV architecture as described in this paper. There are many possible instantiations of a PDV: a personal server housed in a property owned by a user [31], a cloud service operated for profit (with the appropriate trust and owner-

ship guarantees) [11], a base virtual machine housed in a cluster in an infrastructure (also with the appropriate trust and ownership guarantees), and possibly others. For our initial implementation, we explore a complementary cloud-based option, using a bare virtual machine accessible only by the user. In an exploratory implementation of this concept, the PDV is implemented using the virtual hosting platform OpenVZ. Each PDV runs a Linux image and has Apache web server.

Data store. Each owner uploads data through secure means to his/her PDV. The data store is conceptually a collection of named databases, each possibly corresponding to an application or a data type. Simply using structured databases like SQL might work for a short term. However, in structured databases, as the needs of an application evolve, the schema and storage of the existing data must be updated. This often causes problems as new application needs arise, the system needs to make distributed upgrades for every PDV that requires a schema update. In addition, the applications may need different types of data at different times. For example, Waking-survey collects location data continuously and prompts user input data as needed. Thus, we use a schema-free database, CouchDB. Since no schema is enforced with CouchDB, each database can contain many types of data.

Identity. Every access to the PDV must contain an authentication token. This token identifies the entity accessing the PDV, which then determines what data can be shared. We adopt two different authentication mechanisms, X.509 [6] between the PDV and its owner, and OAuth [5] between the PDV and third parties accessing the PDV. The owner can access the PDV once she obtains her public key. Third parties can have access to data at the PDV only with the owner's permission.

Service interface. Both users and third-parties communicate with PDVs via HTTP and exchange data in JSON. The current PDV API allows users to securely add data to PDVs and authorized third parties to access to data filtered by Granular ACLs. The semantics of the API are described as follows:

- **Upload** allows the authenticated user to store her data at the PDV. Five arguments have to be specified: token, data type, data format version, phone version, and data.
- **Pull** allows the authorized third parties to read data from the PDV. Token, third party name, and start and end times of data to read have to be specified. The registered ACLs for the third party in JSON are applied to data before the disclosure of data.
- **Register** allows third parties to register ACLs, which are configured at the third party web servers, to the PDV. Token, third party name, third party webserver url, ACL, and communication type (pull/push) information must be specified.

Access control. is represented as ACLs in JSON. ACLs are evaluated in real time. Only location information at different aggregation levels is cached every five minutes.

Management. The current Rule Recommender and Trace-audit are written in Python and PHP respectively, and run offline and semi-automatically, but will be integrated into a running system in the future.

6. APPLICATION EXPERIENCE

In this section, we demonstrate how our system facilitates the fine-grained data sharing using the two policy management mechanisms and real data traces collected from Ambulation and Waking-survey. Because we are in the development cycle of Ambulation and Waking survey applications, they are not ready to take data at different granularities. We run data generated from the applications through the PDV and use our own tools for visualization. A small data set is used since our goal here is to demonstrate the functional benefits of the system, rather than to evaluate system performance at scale.

6.1 Using Ambulation through PDV

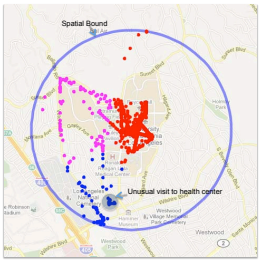

Figure 2: Local Trace-audit visualization showing the data that have been shared with the current sharing policy

We used a data set collected for a month from one of the authors. We ran this data set through PDV with the following usage scenario.

Jenny uses Ambulation to track her activity profiles and share some location traces with her friends. But, she doesn't want them to know what she does during her spare times and decides to share exact location, activity, timestamp information only when she's in Westwood area where she goes almost everyday for school, which is known to most of her friends. Otherwise, she chooses to share location data at zipcode level. After one week, she receives an email reminder to check her Local Trace-audit summary and is redirected to the

PDV policy management site. The Local Trace-audit summary shows what data have been shared with her current sharing policy (See Figure 2). A circle represents the spatial bound and the points are her exact location traces shared. While reviewing her activity, the points marked with blue color remind her of an unusual visit to the health center that she didn't expect to share with her friends. She feels uncomfortable with her current sharing settings and decides to change the sharing policies with help from the Rule Recommender.

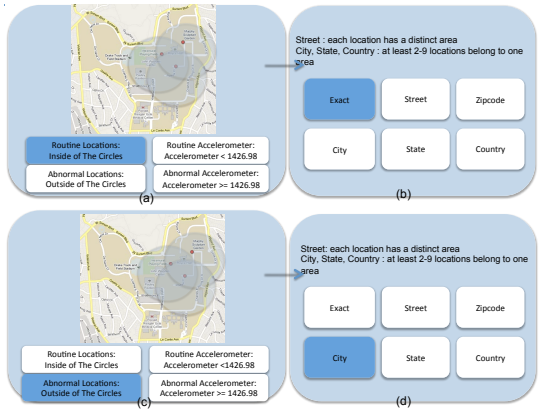

Figure 3: Sample Rule Recommender user interfaces

The Rule Recommender shows pre-computed constraint values for a set of pre-defined privacy policies as shown in Figure 3. She chooses "Routine Locations" and "Exact" options first to share exact location when she is at routine locations (Figure 3 (a) and (b)). Then, she picks "Abnormal Locations" and "City" options to share location at a city level when she is at abnormal locations (Figure 3 (c) and (d)). She does so because she finds that location data at a zip code level is not coarse enough to represent her intention that her visits to unusual significant locations should not be identifiable. Figure 4(left) shows the location tree constructed for her unusual significant locations.

From this tree, the system finds that when location data is aggregated to a city level or coarser, more than two of significant locations always belong to one area. Using the new privacy policies, she doesn't expose her visit to the sensitive location and makes sure that other significant locations are unidentifiable by sharing them at city level. At the same time, she is able to track her location traces in the area where she spends the majority of time everyday.

6.2 Using Waking-survey through PDV

We used a data set that is collected for fifteen days from one of the authors. We ran this data set through PDV with the following usage scenario.

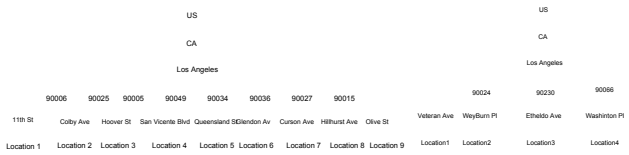

Figure 4: Location tree of significant locations to probe the implication of sharing location data at each of the aggregation levels for Ambulation(left) and Waking-survey(right)

Tom is interested in identifying the causes of bad sleep quality. He uses Waking-survey and shares the data with his physician. He first decides to share all information like bed-time, wake-up time, sleep duration, minutes to fall asleep and sleep quality. However, after checking the data shown on the application side, he feels uncomfortable with sharing his usual bedtime, wake-up time and sleep place information. In addition, too much information is shown and it is hard to interpret the data. He only checks cases when his sleep quality is abnormal. To find out a parsimonious way to share data with his physician, he uses the Rule Recommender system. From the choices provided by the system, he chooses "Abnormal sleep quality", "Abnormal sleep duration", and "Street" options, which implies that he is willing to share data only when abnormal sleep patterns in terms of sleep duration and quality are found (sleep duration <= 6 hrs or sleep duration >= 9 hrs or sleep quality = very good or bad) and location data has to be aggreated to a street level. As shown in Figure 4(right), each of the significant locations belongs to one distinct area when they are aggregated to street levels and it is coarse enough for the user to track where he sleeps without exposing the exact location information to the physician (The physician would not be able to identify exact locations as long as location data are aggregated because he is not familiar with the user's mobility patterns). By sharing exact wake-up time, bed time, minutes to fall asleep information and location at street level only when his sleep pattern is abnormal, less information was disclosed and the user was still able to observe what affects the bad sleep behaviors.

7. EVALUATION

A full evaluation of the PDV architecture will require a user study regarding the usability and utility of the approach to end users and service providers alike. Usability studies require a working system, and so this first work focuses on designing and creating such a prototype and on evaluating the performance of the PDV implementation, including lessons learned and possible enhancements. In our evaluations, we use two months of Ambulation data collected from a real user. The experiment is run with the PDV running on a Dual Core

AMD Opteron Processor 1210 with 2GB of memory. We instrumented the code for profiling the performance of various components. All experiments were repeated 10 times, and we report the average of these 10 measurements.

7.1 Performance of Data Storage

Our first experiment measures the scalability of the PDV implementation. Our initial target is to scale the PDV storage to tens or hundreds of thousands of data items. As described in Section 5, We choose to use CouchDB, instead of traditional SQL database for PDV data storage due to benefits of schema-free databases: flexibility and scalability. We examine if CouchDB can give the same performance of typical SQL. We assume that the application gets data from the PDV at least once per day to run its own processes and to provide users with feedback on time, and measure the time to retrieve a day's data from CouchDB as the size of data in DB increases from 0.06MB (1 day, 1303 entries) to 3.6MB (2 months, 68507 entries).

The data retrieval overhead (1.02 secs) was relatively small when the size of data in DB is small, 0.06MB. However, the overhead increases drastically as the user stores more data at the DB. It took 29.56 seconds and 52.89 seconds when the data size is 1.9MB and 3.6MB respectively. Views are the primary tool used for querying on CouchDB. The current implementation generates temporary views whenever it gets data pull requests from applications. But generating a view of a database with hundreds of thousands or millions of data is time and resource intensive. One solution is to generate permanent views for the data of a short period like a day as data arrives at the PDV, and access them when applications read data from the PDV. [4] shows that the second access to already created views is as fast as querying on indexed tables in MySQL. Our experiments show a consistent results with [4]; it took 0.003 second to access cached data of one day when the data size in DB is 3.6MB.

7.2 Performance of Applying Granular ACL

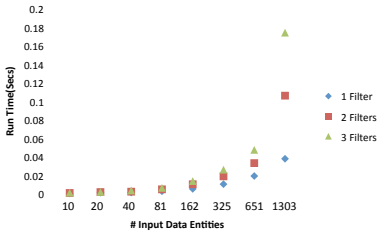

Figure 5: Run time to apply filters with varying numbers of input data entries

In this section, we measure the time to apply different Granular ACLs. The key parameters of our evaluation are the sizes of input data to be filtered and the number of filters used for selective data sharing. Our experiments are based on two assumptions. First, the application gets data from the PDV at least once per day to run its own processes and to provide users with feedback on time. In addition, location information at different aggregation levels like street or city are already computed and cached to support different precision of location data (all other processes required to apply ACLs are done in real time). Thus, we vary the number of input data entries from 10 to 1303 (one day's data), and apply 1, 2 and 3 filters to each of different sizes of data (a filter consists of each of the three constraints: bound, precision and frequency). As shown in Figure 5, the overhead of applying ACLs is negligible in practice: the maximum run time is 0.174 second when three filters (three combinations of bound, precision and frequency) are applied to one day's data. If data are collected every second, the number of data entries is 86,400 per day and it would roughly take 2.513 seconds and 11.53 seconds for applying one and three filters respectively (Ambulation data are collected every 30 seconds).

Additionally, we apply different types of filters to a same data set. One observation found is that the completion time depends partially on the number of entries satisfying the filter. For instance, applying a filter of one tag bound took 0.073 second when 1164 entries are returned while it took 0.036 second when 17 entries satisfy the given constraint.

7.3 Performance of Core Policy Management System Functions

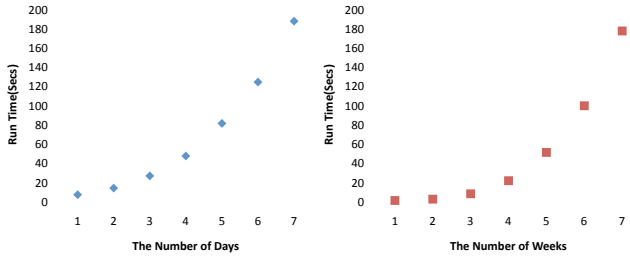

Figure 6: Run time to run significant location identification(left) and anomaly detection(right) algorithms with varying data sizes

In this section, we examine how the Rule Recommender processing overhead scales with the number of data points. We measure the performance of the core functions that are used to compute all three constraints and the most time-consuming, significant location identification and anomaly detection algorithm (LOF).

We quantify completion time to run the two algo-

rithms with varying the data size: one to seven days' data for significant location identification and one to seven weeks' data for LOF (we use the data collected from Waking-survey to run LOF that is collected only once per day and vary the number of weeks instead of days). As shown in Figure 6, the time to run both algorithms increases drastically as the duration for collecting data increases because both algorithms are based on k-nearest neighbor and the number of computations grows exponentially as more data points are added. The significant location identification algorithm must be run daily to find routine locations to compute the bound. Considering this, the overhead is acceptable; it took 6.19 seconds when one day's data were used. For LOF, the system should consider the trade-off between computation overhead and policy accuracy if we assume that the more historical data are used, the more accurate policy can be created.

8. RELATED WORK

Privacy regulation and protection are critical topics in the design of ubiquitous and pervasive systems [12],[15]. Diverse research has suggested methods to obscure, hide, or anonymize data to protect participants' privacy [17], [22], [23], [24], [10], [26]. Achieving anonymity relies on separating data from identifying features (such as obscuring home or workplaces in GPS traces) or recruiting data from large samples to obscure individual identities. Other popular privacy mechanisms include adding noise to data to make it less exact and therefore more difficult to extract identifying information. These kinds of popular privacy preserving techniques can be added to PDV to be used for grassroots participatory sensing applications [17], where groups of participants use privately-owned sensors to collectively measure aggregate phenomena of mutual interest. However, this paper focuses on building a privacy architecture for personal participatory sensing, in which maintaining personal identities and trends is essential and we don't deal with them.

Other recent work has proposed similar architectures to the PDV for online social network applications. They include techniques that: 1) implement access control lists at service providers based on social attestations (Lockr) [33]; 2) devise per-social-network overlays constructed out of personal storage entities (VIS) [11]; or 3) implement truly decentralized social network architectures (PrPl) [31]. Persona [7] uses encryption and out-the-band key exchange with other users to realize access control over data sharing. NOYB [19] obfuscates a user's sensitive data by using a secret function to permute the data of all online social network users. The PDV also has similarities to commercial developments for personal health records, such as Microsoft's HealthVault [3] and Google Health [2]. But, most of these techniques are built for online social networks and have focused on managing groups which govern who has access to what data. They have paid less attention to controlling the level of data for sharing, which is critical for personal participatory sensing in which subtle changes on the granularity of data for sharing could result in very different consequences to users due to different processes on the application side and users' implicit tolerance for privacy. Our work emphasizes the importance of controlling the granularity of data for sharing and introduces three mechanisms that are essential to support the fine-grained data sharing.

9. CONCLUSIONS

We present the PDV architecture, an individually-controlled secure data repository decoupling the capture and archiving of personal data streams from the sharing of that information. This paper describes the key motivation, requirements and designs of the PDV. To illustrate the feasibility of PDV, we developed a prototype of the architecture and demonstrated how our system facilitates the fine-grained data sharing using the two policy management mechanisms and real data traces collected from Ambulation and Waking-survey. We also evaluated the system resource requirements of the core PDV functions, storage and selective sharing, as well as policy management services that are essential to support PDV use. The most critical focus for future work is to evaluate and improve the usability of the system.

10. REFERENCES

[1] Freereversegeo. www. freereversegeo.com.
[2] Google health. https://www.google.com/health.
[3] Microsoft healthvault. http://www.healthvault.com.
[4] Mysql - couchdb performance comparison. http://metalelf0dev.blogspot.com/2008/09/mysql-couchdb-performance-comparison.html.
[5] oauth. http://oauth.net/.
[6] X.509. http://en.wikipedia.org/wiki/X.509.
[7] R. Baden, A. Bender, N. Spring, B. Bhattacharjee, and D. Starin. Persona: An online social network with user-defined privacy. In *SIGCOMM*, 2009.
[8] M. M. Breunig, H. P. Kriegel, R. T. Ng, and J. Sander. Lof: Identifying density-based local outliers. In *ACM SIGMOD*, 2000.
[9] J. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy, and M. Srivastava. Participatory sensing. In *ACM Sensys WSW Workshop*, 2006.
[10] C. Cornelius, A. Kapadia, D. Kotz, D. Peebles, M. Shin, and N. Triandopoulos. Anonysense: privacy-aware people-centric sensing. In *MobiSys*

'08: Proceeding of the 6th international conference on Mobile systems, applications, and services, pages 211–224, New York, NY, USA, 2008. ACM.

[11] R. Cceres, L. Cox, H. Lim, A. Shakimov, and A. Varshavsky. Virtual individual servers as privacy-preserving proxies for mobile devices. In *Proc. of 1st ACM SIGCOMM Workshop on Networking, Systems, and Applications on Mobile Handhelds (MobiHeld)*, 2009.

[12] D. K. D. Anthony and T. Henderson. Privacy in locationaware computing environments. In *Pervasive Computing*, 2007.

[13] P. Dutta, P. Aoki, N. Kumar, A. Mainwaring, C. Myers, W. Willett, and A. Woodruff. Common Sense: Participatory Urban Sensing Using a Network of Handheld Air Quality Monitors (demonstration). In *Proc. SenSys*, 2009.

[14] S. E.Miluzzo, N.D. Lane and A. Campbell. Cenceme injecting sensing presence into social networking applications. In *Proc. of EuroSSC*, 2007.

[15] G. H. et al. Physical, social and experiential knowledge in pervasive computing environments. In *Pervasive Computing*, 2007.

[16] J. F. Froehlich, M. Y. Chen, and et al. Myexperience: a system for in situ tracing and capturing of user feedback on mobile phones. In *ACM MobiSys*, 2007.

[17] R. K. Ganti, N. Pham, Y. Tsai, and T. F. Abdelzaher. Poolview: Stream privacy for grassroots participatory sensing. In *Sensys*, 2008.

[18] S. Gaonkar, J. Li, R. R. Choudhury, L. Cox, and A. Schmidt. Micro-blog: sharing and querying content through mobile phones and social participation. In *Proceedings of MobiSys*, 2005.

[19] S. Guha, K. Tang, and P. Francis. Noyb: Privacy in online social networks. In *WOSP 08: Proceedings of the First Workshop on Online Social Networks*, 2009.

[20] J. Hicks, N. Ramanathan, D. Kim, M. Monibi, J. Selsky, M. Hansen, and D. Estrin. Andwellness: An open mobile system for activity and experience sampling. In *Proc. of Wireless Health*, 2010.

[21] B. Hoh and et al. Enhancing security and privacy in traffic-monitoring systems. In *IEEE Pervasive Computing*, 2006.

[22] B. Hoh, M. Gruteser, R. Herring, J. Ban, D. Work, J. Herrera, and et al. Virtual trip lines for distributed privacy-preserving traffic monitoring. In *Proceeding of the 6th international conference on Mobile systems, applications, and services*, 2008.

[23] J. I. Hong and J. A. Landay. An architecture for privacy-sensitive ubiquitous computing. In *Proceeding of the 6th international conference on Mobile systems, applications, and services*, 2004.

[24] J. Horey, M. M. Groat, S. Forrest, and F. Esponda. Anonymous data collection in sensor networks. In *Proceedings of the 4th Annual International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, 2007.

[25] J. Kang, W. Welbourne, B. Stewart, and G. Borriello. Extracting places from traces of locations. In *Mobile Computing and Communications Review*, 2005.

[26] J. Krumm. Inference Attacks on Location Tracks. *Lecture Notes in Computer Science*, 4480:127, 2007.

[27] M. Y. Mun, S. Reddy, K. Shilton, N. Yau, and et al. Peir, the personal environmental impact report, as a platform for participatory sensing systems research. In *Mobisys*, 2009.

[28] H. Nissenbaum. Privacy as contextual integrity. In *Washington Law Review*, 2004.

[29] A. Parker, S. Reddy, and et al. Network System Challenges in Selective Sharing and Verification for Personal, Social, and Urban-ScaleSensingApplications. In *HotNets*, 2006.

[30] J. Ryder, B. Longstaff, S. Reddy, and D. Estrin. Ambulation: A tool for monitoring mobility patterns over time using mobile phones. In *Social Computing with Mobile Phones Workshop at IEEE SocialCom*, 2009.

[31] S. Seong, J. Seo, M. Nasielski, D. Sengupta, S. Hangal, and et al. PrPl:A Decentralized Social Networking Infrastructure . In *ACM Workshop on Mobile Cloud Computing and Services: Social Networks and Beyond*, 2010.

[32] K. Shilton, J. Burke, D. Estrin, M. Hansen, R. Govindan, and J. Kang. Designing the personal data stream: Enabling participatory privacy in mobile personal sensing. In *The 37th Research Conference on Communication, Information and Internet Policy (TPRC)*, 2009.

[33] A. Tootoonchian, S. Saroiu, Y. Ganjali, and A. Wolman. Lockr: Better privacy for social networks. In *CoNEXT*, 2009.

[34] C. Zhou, D. Frankowski, P. Ludford, S. Shekhar, and L. Terveen. Discovering personal gazetteers: an interactive clustering approach. In *ACM GIS*, 2004.