4-30-2018

Design and evaluation of a privacy architecture for crowdsensing applications

Alfredo J. Perez

Sherali Zeadally

Design and evaluation of a privacy architecture for crowdsensing applications

Alfredo J. Perez
Columbus State University 4225 University Ave. CCT 424 Columbus, GA, USA 31907
perez_alfredo@columbusstate.edu

Sherali Zeadally
University of Kentucky 315 Little Library Building Lexington, KY, USA 40506-0224
szeadally@uky.edu

ABSTRACT

By using consumer devices such as cellphones, wearables and Internet of Things devices owned by citizens, crowdsensing systems are providing solutions to the community in areas such as transportation, security, entertainment and the environment through the col- lection of various types of sensor data. Privacy is a major issue in these systems because the data collected can potentially reveal aspects considered private by the contributors of data. We propose the Privacy-Enabled ARchitecture (PEAR), a layered architecture aimed at protecting privacy in privacy-aware crowdsensing systems. We identify and describe the layers of the architecture. We propose and evaluate the design of MetroTrack, a crowdsensing system that is based on the proposed PEAR architecture.

Keywords
Privacy; Security; Crowdsensing; Community-based Sensing; Participatory Sensing; Opportunistic Sensing; Internet of Things; Wireless Sensor Networks.

1. INTRODUCTION

The ubiquity of Internet-connected consumer devices have the potential to improve issues affecting communities through the crowdsensing paradigm. In crowdsensing systems (also known as community-based sensing systems), people collect sensor data using their own consumer devices such as cellphones, wearables, and Internet of Things (IoT) devices (e.g., Internet-connected cars) with the goal of helping a community (e.g., a city or town) in areas such as environmental monitoring, transportation, entertainment, security, and healthcare.

Recent technological advances in the last decade in the miniaturization of sensors, computing power, and the mobile Internet, along with the ubiquity of these consumer devices have enabled the implementation and deployment crowdsensing systems [21][30]. The protection of privacy in crowdsensing systems plays an important role in their successful development and deployment because the use of consumer devices to collect sensor data presents significant privacy risks to all of the participants involved. Thus, the use of Internet-connected consumer devices in crowdsensing faces a tradeoff: on one hand we need to collect data as accurately as possible, but on the other hand the collection and sharing of crowdsensing data must preserve the privacy of users [1][31].

We review the privacy issues and solutions in crowdsensing systems, then we present the Privacy-Enabled Architecture (PEAR), a layered architecture aimed at protecting the privacy of users involved in crowdsensing systems. We describe the main components of the architecture, and we evaluate MetroTrack, an implementation of the proposed PEAR architecture.

2. CROWDSENSING SYSTEMS

Advances and integration of Micro Electro-Mechanical Systems (MEMS) with communication and computing devices in the 1990's paved the way for the development of Wireless Sensor Networks (WSNs). Even though early research in this area was performed in the 1970's through the Defense Advanced Research Project Agency's (DARPA) Distributed Sensor Networks program [30], the vision behind the development of WSNs during the 1990's was to collect data of interest in a particular geographical area while minimizing the cost to deploy and maintain the network, and maximizing the lifetime of the WSN and the coverage area the WSN monitors. Thus, the devices that were developed for WSNs were of small sizes (as small as 2 mm^2) while at the same time these devices could sense, transmit data, and harvest their energy needs from the environment. It was envisioned that WSNs could be active for years without human intervention and to be cheap enough to be deployed everywhere to enable the deployment of thousands of sensing de- vices in a single WSN [30].

As research progressed in the development of WSN systems, certain goals, in particular the ones that minimize the costs of deployment and maintenance were not achievable when deploying thou- sands of devices in a single WSN as initially thought. The consequence was that the actual deployments of static WSNs were done as small and focused WSNs deployments with only a few hundreds of devices [30]. In the early 21st century, cellular devices started to become ubiquitous and, in the past decade, these cellular devices became powerful enough to connect to the Internet. Eventually, these mobile devices were manufactured with integrated sensors such as accelerometers, gyroscopes, location sensors which could be potentially used to collect different types of data of interest.

Crowdsensing systems have been developed during last decade as an alternative to static wireless sensor networks in urban environments to address the collection of data in an inexpensive manner. These systems take advantage of the availability of the mobile Internet and the ubiquity of powerful, sensor-enabled consumer de- vices such as mobile phones, wearables and IoT devices [5][30]. Application areas where these systems are being deployed include environmental monitoring, transportation, entertainment, security, among others [17][20][21][22][27][28]. In this section we provide an overview of crowdsensing systems, the threats to privacy in crowdsensing systems, and a review of the solutions available to help mitigate these threats.

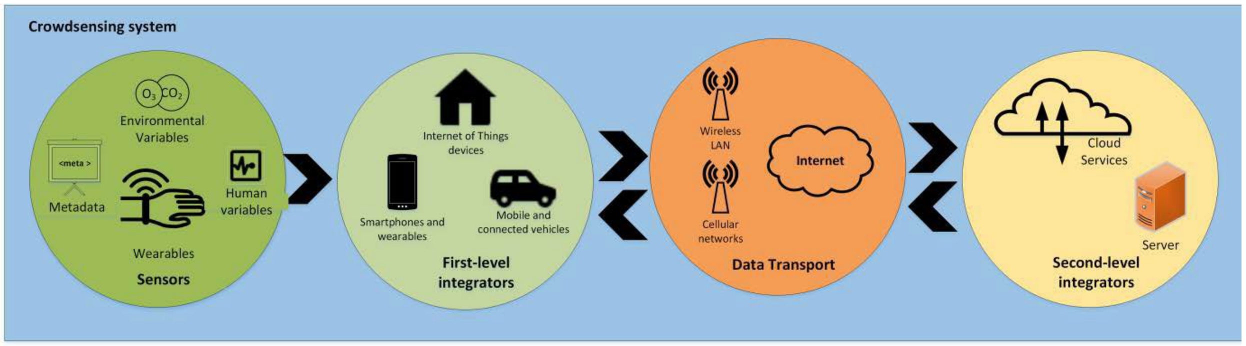


Figure 1. Hardware components of crowdsensing systems.

2.1 Hardware Architectures and Actors

The typical hardware components of crowdsensing systems are presented in figure 1, and they include [30]:

- *Sensors*: These components collect data (e.g., temperature, movement, noise, images) from physical actions or processes.
- *First-level integrators*: They perform initial data verification, aggregation and basic analysis (e.g., feature ex- traction) on the data collected by sensors.
- *Data transport*: In crowdsensing systems, data transport is provided by the Internet or any communication net- work that enables the end-to-end transfer.
- *Second-level integrators*: These components collect and store data sent from first-level integrators. They also pro- vide analytics services and feedback to first-level integra- tor devices and to external entities.
- These components are utilized to collect sensor data through *sensing tasks* which are software applications or scripts executed at first-level integrator devices. There are three major actors that participate in the data collection for crowdsensing systems:
- *Task organizers*: They develop the sensing tasks that will be deployed at first-level integrators.
- *Participants*: They represent the entities that own or are in custody of first-level integrator devices. They may be provided feedback based on the collected data.
- *External entities*: They represent third parties whom task organizers can share data (or information) generated from the sensor data.

The actors interact with the hardware in the crowdsensing system with the goal to collect sensor data and metadata through the data collection cycle shown in figure 2. This collection cycle is com- posed of the following processes:

- *Task distribution*: The goal of task distribution is to re- lease the sensing task to user participants. This is accomplished in two ways: participants either makes use of the sensing task from a server (second-level integrators), or the task is pushed to the users' devices (first-level integrators) from second-level integrators.
- *Data collection*: Once tasks are configured at the participants' devices, the tasks perform sensing and initial analysis that may include extracting features from sensor data, smoothing and filtering of outliers in the data, and data analytics that can be performed locally without the need of second-level integrators.
- *Data submission*: Tasks that execute at first-level integrators forward the collected data to second-level integrator devices. Depending on the system's goals, data submission can be performed continuously or based on events (identified in the data collection process), and data can be submitted in real time or later (e.g., at the end of the day).
- *Data analysis and sharing*: In this process, second-level integrator devices use the collected data from fist-level integrators to perform analytics services (e.g., data analysis and machine learning) and provide feedback to first- level integrator devices. The feedback may include the release of new sensing tasks to user participants, resulting in a new data collection cycle. In addition, data may be released to external parties outside the system through this process.

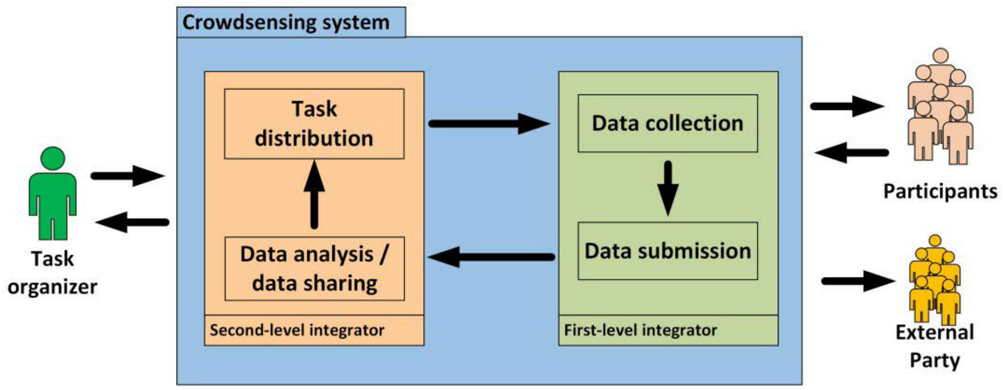


Figure 2. Processes for data collection in crowdsensing.

3. RELATED WORKS ON PRIVACY SO- LUTIONS FOR CROWDSENSING

As the devices that collect sensor data in crowdsensing systems are usually owned by citizens, the protection of their privacy becomes an important issue for the successful deployment of these systems. The data collected could be potentially linked with the identities of participants or the data could reveal aspects about individuals that are considered private [1][6][31][32][36]. Attacks to privacy in crowdsensing may be broadly classified as (1) *re-identification at- tacks*; (2) *contextual attacks.* We summarize the advantages and disadvantages of privacy protection mechanisms to mitigate re-identification and contextual attacks in table 1 and table 2 respectively.

3.1 Re-identification Attacks

Re-identification attacks are successful when a rogue entity discovers the identity of participants from the data (or metadata) collected (or submitted by the participants) in the system. These attacks may occur because the participants' identities are inferred from metadata such as network addresses/identifiers (e.g., IP addresses, MAC addresses and cookies) which are needed by network protocols to send and receive data, or they may be accomplished through the discovery of identities from any of the tasks performed by the crowdsensing system to collect data (figure 2). Privacy protection mechanisms against re-identification attacks which use network identifiers can be achieved by using double encryption via brokers [36][38], peer-to-peer (P2P) anonymization networks [2][6], and the utilization of disposable network identifiers such as pseudonyms [8][10][13][35].

In the case of re-identification attacks, because of the management of sensing tasks in the crowdsensing system, privacy solutions de- pend on the processes involved during data collection (as shown in figure 2). For example, for task distribution, the privacy solutions to avoid re-identification include the utilization of beacons that dis- tribute tasks through the broadcasting of signals from the beacons or access points (such as WiFi access points) to the first-level integrator devices [36], task downloading at crowded spaces (in the case of mobile, first-level integrator devices) [36], and the use of anonymization networks [2]. In the case of data collection and sub- mission processes, solutions to handle re-identification include the use of group-based signatures [15], data aggregation [34], and the use of representative samples from the data collected in a region [16][38].

3.2 Contextual Attacks

Given the data (or metadata) submitted to the system by participants, contextual attacks attempt to discover and associate aspects considered private by participants with their identity. Examples in this category include inferring contexts such as places, activities, behaviors, and/or health state based on the collected data. The goal of these solutions is to diminish or eliminate the risks of discovering and associating aspects or contexts considered private by participants from the data (or metadata) submitted them to the system. Table 2 presents the advantages and disadvantages of privacy protection mechanisms to mitigate re-identification attacks.

To thwart contextual attacks, privacy solutions can be divided into two groups: (1) solutions to manage contextual attacks at the data collection stage; (2) solutions to manage contextual attacks when the collected data is shared externally (to third parties). In the first group of solutions (privacy protection at data collection) solutions include the *bubble sensing* [24] approach which allows data collection in a particular context and the *virtual walls* approach which deny data collection at predefined contexts [7]. In the case of external data sharing, solutions to handle contextual attacks include anonymization in the release of microdata (e.g., k-anonymity [37], l-diversity [25], and t-closeness [23]), and methods to release aggregated data (e.g., statistical data) [9][12]. We summarize the main research contributions of this work as follows:

- We propose the design of a privacy architecture called Privacy Enabled Architecture (PEAR) which is a layered software architecture for crowdsensing systems wherein first-level integrators and second-level integrators are abstracted as data contributors. This allows second-level integrators to become data providers for other crowdsensing systems without defining new layers.
- We present the implementation of MetroTrack, a crowdsensing system based on the PEAR architecture that runs on the JavaEE frameworks and the Android platform.
- We evaluate the privacy mechanisms incorporated in the design of MetroTrack to protect participants’ privacy.

4. THE PRIVACY ENABLED ARCHITEC- TURE (PEAR)

We propose the PEAR architecture which consists of four abstraction layers namely *communication, anonymization, security and privacy, and processing*. First-level and second-level integrator de- vices are referred to as *integrators* in the PEAR architecture, as this abstraction allows second-level integrator devices to become data providers (i.e., first-level integrators) for other crowdsensing systems without having to define new layers to share data. This architecture also manages scalability by creating networks of second- level integrator devices. Figure 3 depicts an example of a PEAR-enabled system.

4.1 Communication Layer

This layer abstracts the communication protocols between integra- tor devices. Usually these protocols are implemented by the TCP/IP protocol stack suite. However, given the ubiquity of cellular net- works in crowdsensing systems, communication protocols for crowdsensing could be implemented over the Multimedia Messaging System (MMS) infrastructure, or Short Messaging System (SMS).

4.2 Anonymization Layer

This layer implements mechanisms that allow integrator devices to hide network location identifiers (e.g., IP addresses) from other integrators and external parties to avoid re-identification attacks. These anonymization mechanisms may be implemented through trusted third parties [38], or by using peer-to-peer anonymization networks (e.g., Tor). Systems may bypass anonymization depending on the goals of the system, or if participants give consent to include network identifiers as part of the data collected by the crowdsensing system.

4.3 Security and Privacy Layer

This layer implements mechanisms and protocols to encrypt data between integrator devices, and includes privacy-preserving mechanisms for integrator devices. Security mechanisms in this layer include symmetric and asymmetric cryptographic methods and protocols that guarantee end-to-end security between integrator devices. Privacy mechanisms in this layer implement algorithms/procedures to allow participants to handle their exposure to context privacy attacks (e.g., privacy rules, algorithms to handle location privacy) and mechanisms to handle privacy for second-level integrators when sharing bulk data release (microdata release) [9] and aggregated (summarized, statistical) data release with external parties.

Table 1. Privacy protection mechanisms for re-identification attacks.

Type of attack	Protection mechanism	Advantages	Disadvantages
Re-identification attacks from network identifiers	Double encryption via brokers [36][38]	Easy to implement Protects participants' IP network identifiers	Requires a trusted third party to forward the data before submission to cloud services May have significant power overhead due to encryption, an issue for battery-powered devices
	Peer-to-Peer (P2P) anonymization networks [2][6]	Does not require a trusted third-party Protects participants' IP network identifiers	May not be available to all crowdsensing devices due to OS or hardware constraints May have significant power overhead due to P2P network maintenance, which is an issue for battery-powered devices
	Disposable network identifiers [8][10][13][35]	Lightweight Mechanism can be implemented on any crowdsensing device Low-power consumption	May not protect participants from some type of network re-identification (e.g., IP addresses)
Re-identification attacks from task management	Broadcasting of signals from beacons to download sensing tasks [36]	Hides identity of participant by not having the participant's device to initiate network communication (device only receives data)	May require the modification of the infrastructure (e.g., WiFi access points) or special agreements to broadcast signals
	Task downloading at crowded spaces [36]	Assumes that there are enough people at a given place so that the identities of the participants are hidden in the crowd	May not be possible to use in rural or isolated places due to the difficulty in getting a crowd
	Use of Anonymization networks for task downloading [2]	Hides participant's identity through by forwarding the data to the anonymization system	May not be available to all crowdsensing devices due to OS or hardware constraints
	Group-based signatures for data submission [15]	May be easy to implement as it may be implemented as a shared credential	Privacy protection depends on the size of the group to guarantee low probability of re-identification
	Data aggregation [34]	Conceals identity by aggregating data (e.g., calculating averages) on participants' values before submission	Requires coordination between participants to calculate aggregates
	Use of representative samples of data collected in a region [16][38]	Conceals identity by choosing best/representative samples from a pool of participants	Require coordination between participant's devices to select the representative samples

Table 2. Privacy protection mechanisms for contextual attacks.

Type of attack	Protection mechanism	Advantages	Disadvantages
Contextual attacks during data collection	Method allows data collection at specific contexts [24]	Allows participant to specify rules to collect data, giving them more autonomy where to collect	Contexts may be hard to specify and may require also Artificial Intelligence (AI) to create usable rules
	Method denies data collection at specific contexts [7]	Allows participant to specify rules to collect data, giving them more autonomy where to collect	Contexts may be hard to specify and may require also AI to create usable rules Participants may inadvertently leak private data as a result of not specifying contexts correctly
Contextual attacks when data is shared externally	k-anonymity [37]	Works on microdata release Protects against identity disclosure Simple to implement and understand because any record in the microdata has at least k-1 records with similar attributes to hide the real record (s) to be protected	k-anonymity does not protect against attacks when background knowledge is used Does not protect privacy if all the values in the k-anonymity group share the same sensitive attribute
	l-diversity [25]	Works on microdata release Protects against sensitive attributes' disclosures Extends k-anonymity by imposing the condition that l well-represented values must be present in the group of data to be anonymized	May not protect the microdata release from attacks such as the *skewness attack* (the distribution of the sensitive values in the anonymization group is skewed from the overall distribution of the microdata) and *the similarity attack* in which all the values in the anonymization group for the records in the microdata have the same semantic (real life) meaning
	t-closeness [23]	Works on microdata release Protects against skewness and similarity attacks (improves over l-diversity) Assures that the distribution on the sensitive values in the anonymization group has a similar distribution when compared with the overall distribution of the sensitive values in the microdata	The value t may be hard to determine depending on data to be protected The t value may have different privacy implications depending on the type of data (e.g., a particular t value can be interpreted differently for numeric and/or categorical data)
	Differential privacy [9][12]	Works on statistical (summarized) data release	The calculation of the statistical value to achieve differential privacy depends on the type of data used to calculate the statistical value (e.g., it differs between categorical and numerical data)

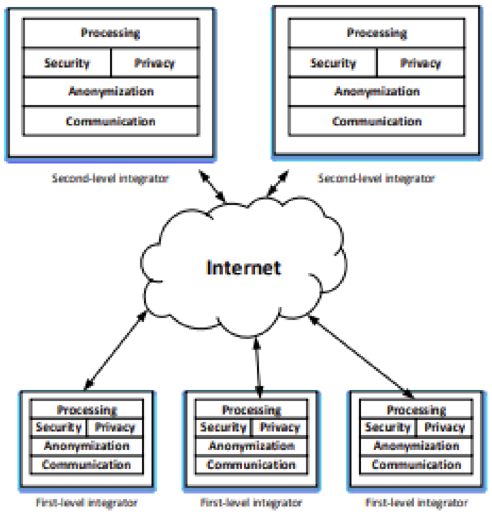


Figure 3. A PEAR-enabled system.

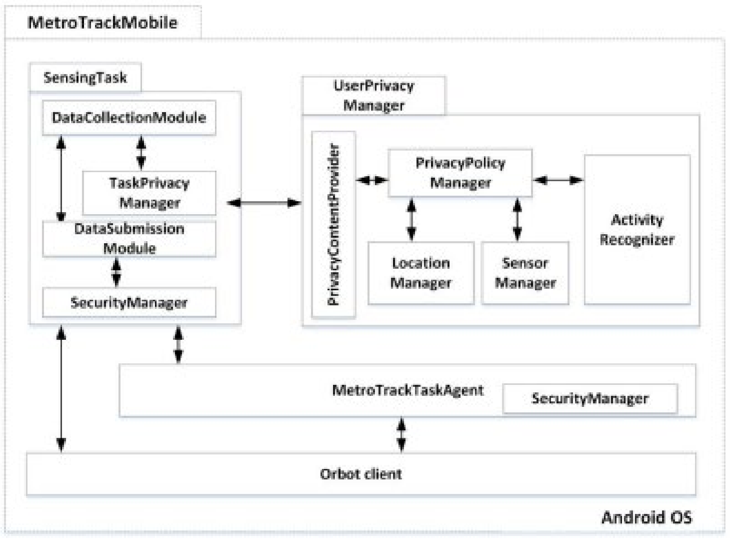


Figure 4. MetroTrack's mobile components.

4.4 Processing Layer

This layer includes mechanisms and protocols that collect and analyze sensor data. These mechanisms may be implemented at first- level integrator devices to perform initial data analysis (e.g., feature extraction, data smoothing) and at second-level integrators to ex- tract information (e.g., outlier detection, machine learning). The processing layer may include mechanisms to handle privacy implemented by task organizers for integrator devices (e.g. a task organizer specifies where sensor data should not be collected) and may also include mechanisms to provide incentives for participants. The processing layer mechanisms make use of software Application Programming Interfaces (e.g., an app development framework, a server-side framework) for in their implementation. The processing layer also implements mechanisms to perform efficient data collection (e.g., power optimization in case of first-level integrator devices).

5. MetroTrack: A PROTOTYPE SYSTEM USING PEAR

We describe the design of a prototype implementation of the PEAR architecture called MetroTrack, a system on which a city administration (task manager) can issue crowdsensing tasks to its citizens (participants) to collect data of interest. In this system, the citizens participate altruistically in the data collection, and tasks can be either participatory (e.g., uploading of photos/videos for security [3]) or opportunistic (e.g., tracking of road congestion, or road maintenance status [26]). *MetroTrack* consists of client (participant) ap- plications executing on Android-enabled devices, and server-side components that are deployed in the cloud using the Java Enterprise Edition (EE) framework.

5.1 MetroTrack's Mobile Components

Four applications make up the MetroTrack's mobile components for the Android OS (figure 4). Each of these components implements different layers of the proposed PEAR architectural model.

5.1.1 Orbot client

MetroTrack makes use of the Orbot [29] application which is the Tor's network proxy for Android. The Tor network [11] is an anonymization network that provides network anonymization to TCP flows. Tor works by having the client to create a path through Tor hosts from the client to a server. Messages along this path are encapsulated into layers of encryption (like an onion) at the client, and each host along the path removes an encryption layer (like peeling an onion), which allows the current host to know the next host to forward the message. Once the final layer is decrypted, the last host delivers the message to its original destination [11].

Android applications can use Orbot to access a server in the Internet through a local proxy in the device, or can incorporate Orbot as a component within an application. In our current design, Orbot is used as a proxy. Orbot is open source and can be downloaded from the Google Play market. Orbot serves as part of the anonymization and communication layers of the PEAR architecture for MetroTrack. A screenshot of the Orbot client is shown in figure 5.

5.1.2 MetroTrackTaskAgent

The MetroTrackTaskAgent is used by the MetroTrack system to deliver participants' information about new tasks issued in the system. The agent can also notify participants about updates on previously issued tasks. As shown in figure 4, MetroTrack TaskAgent uses Orbot to connect to the server components of MetroTrack.

The agent retrieves notifications about new tasks available to participants from the MetroTrack server components. Since Orbot does not provide end-to-end security, the task agent must secure the requests before using Orbot. Transport Layer Security

(TLS) security provides the security mechanism for the task agent. The MetroTask Agent is part of the processing layer of PEAR, with the security components of the task agent being part of the security and privacy layer of PEAR. Participants can download the task agent from the Google Play market.

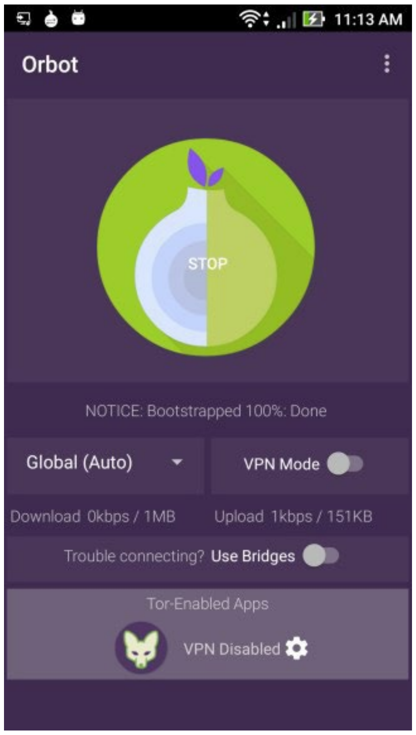
5.1.3 *UserPrivacyManager*

Through this component, participants can configure their own privacy settings for SensingTasks. These privacy settings are implemented as privacy rules based on sensor and date/time data, and they can be implemented as simple rules (e.g., “don’t provide data to this task if close to a particular location”), or more complex con- textual rules based on activity recognition (e.g., “don’t provide data if sleeping”). This module is composed of subcomponents such as PrivacyPolicyManager which implements the rules, the ActivityRecognizer module which recognizes activities based on sensor data, and the PrivacyContentProvider which provides sensor data to tasks based on the decisions of the PrivacyPolicyManager. The SensorManager and LocationManager components are part of the Android API and provide information to the PrivacyPolicyManager subcomponent.

The PrivacyContentProvider subcomponent provides raw data to a sensing task based on the decision of the privacy rules. In the cur- rent design, the UserPrivacyManager could be downloaded from Google Play. However, this module could become part of mobile operating systems as part of the privacy/security settings.

5.1.4 *SensingTask*

This component implements the collection of data for the task man- ager. Sensing tasks are downloaded from MetroTrack’s servers using Orbot. Each download has a unique identifier that is hardcoded when the SensingTask is compiled as an app ready to install. This design allows MetroTrack servers to authenticate the each of the task installations instead of authenticating participants. The rationale is that MetroTrack only needs to make sure that the data is coming from an authorized party, and this can be accomplished by hardcoding IDs into each download of the app.



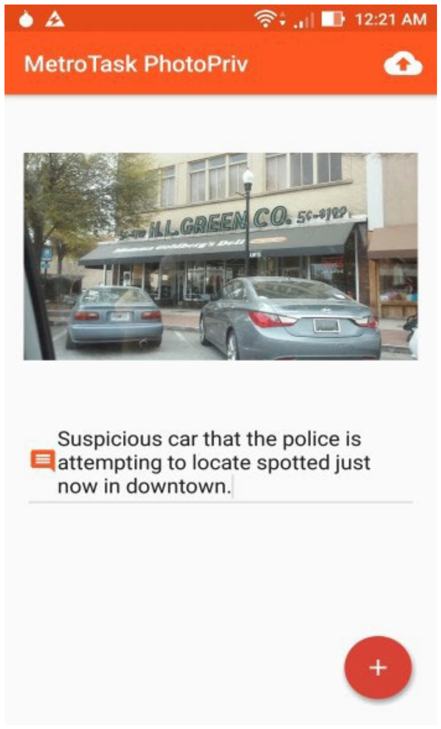
NOTICE: Bootstrapped 100%: Done

Global (Auto)

Download 0kbps / 1MB   Upload 1kbps / 151KB

Trouble connecting? Use Bridges

Figure 5. Orbot Android client.



Suspicious car that the police is attempting to locate spotted just now in downtown.

Figure 6. PhotoPriv: A MetroTrack task.

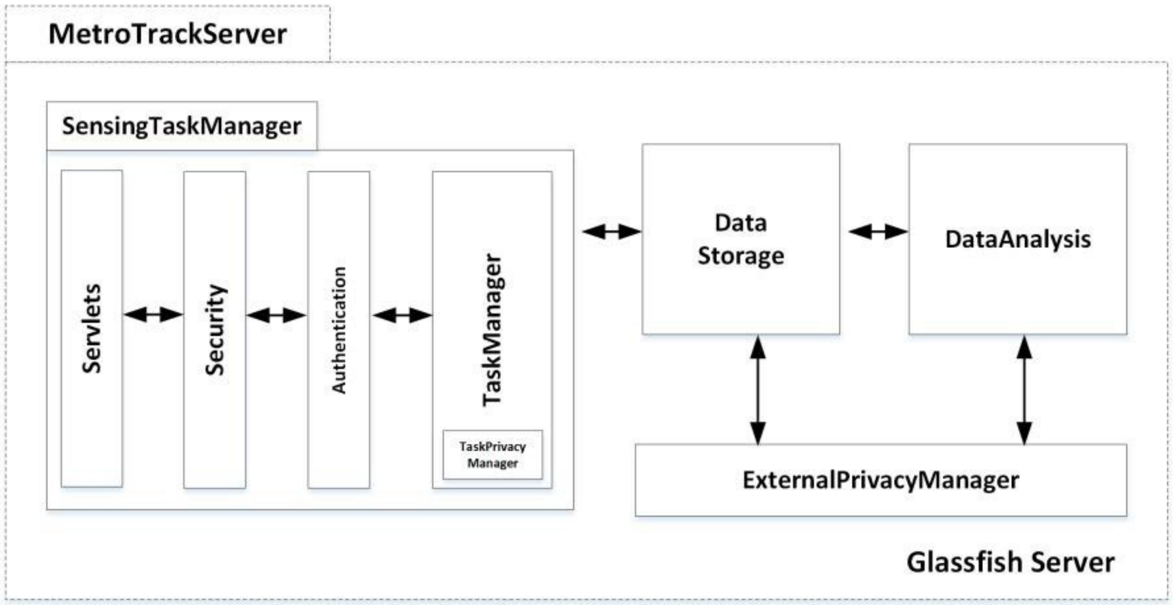


Figure 7. MetroTrack's server components.

Subcomponents of this module include: the DataCollectionModule which collects data from the UserPrivacyManager module and per- forms basic data analysis (e.g., feature extraction), the TaskPrivacyManager which implements privacy rules established by the task manager (and also includes mechanisms to show consents to participants), the DataSubmissionModule which prepares the data for submission, and the SecurityManager which manages authentication, session establishment, and end-to-end encryption with the server. The SecurityManager may use the MetroTrackTaskAgent to check if the current task is still valid. The SecurityManager also utilizes the Orbot component to submit data to the MetroTrackServers.

Figure 6 shows an example of a MetroTrack SensingTask called *PhotoPriv*. In this task, citizens are notified when people or objects of interest (e.g., stolen cars, hit and runs, amber alerts, thieves) are present in a location where a law enforcement agency may be attempting to locate by using crowdsensing. Contributors can use *PhotoPriv* to send anonymous geo-located photos that are uploaded with a message to law enforcement agencies. We used the NetCipher library in *PhotoPriv* to send data through the Orbot app to the server.

5.2 MetroTrack's Server Components

The MetroTrack's server consists of four major components, namely *SensingTaskManager, DataStorage, DataAnalysis,* and *Ex- ternalPrivacyManager*. Figure 6 shows the flow of data among these components. Our current design assumes that these components will execute in a Java EE application server (e.g., Glassfish server).

5.2.1 *SensingTaskManager*

This component handles the management of sensing tasks for MetroTrack's mobile components. The SensingTaskManager is used by the task organizer to announce new SensingTask apps to participant's devices. As mentioned in the previous section, each instance of a SensingTask downloaded by participants has its own identifier which allows it to be authenticated by MetroTrack servers. This is performed by having different compilations of the same Sensing- Task and offering these on demand. SensingTasks are meant to be lightweight, and a background process in the SensingTaskManager is constantly compiling and caching the SensingTasks. The

SensingTaskManager component also provides the mechanisms to handle security, authentication and sets up privacy rules for SensingTasks.DataStorage

This component abstracts the operations needed to store the data received by the SensingTaskManager into database systems. De- pending of the type of SensingTask, the data may be structured, unstructured, or a combination of both types of data. PostgreSQL, MySQL and other database management systems (or regular file systems) may be used to store data.

5.2.2 *DataAnalysis*

This component allows a task manager to perform inference, correlation, and data analysis based on the data received from Sensing- Tasks. This component can filter outliers, detect trends and pat- terns, and perform data analysis that could be only performed at the server. This module allows a task organizer to have a complete picture of the situation being studied. Task organizers may take measures such as preparing and releasing new tasks, or providing reports to third parties.

5.2.3 *ExternalPrivacyManager*

MetroTrack makes use of this component to handle privacy when data is shared with external systems or parties. The algorithms implemented in this module include mechanisms such as *k*-anonymity [37], l-diversity [25], t-closeness [23] to handle privacy for bulk data release (microdata release). For aggregated data, differential privacy mechanisms [12] may be used.

5.3 **Analysis of MetroTrack**

5.3.1 *Evaluation of privacy goals in MetroTrack*

MetroTrack system was designed to mitigate re-identification and contextual privacy attacks described in section 2.2. It achieves this by implementing privacy protection solutions through independent components at the mobile devices and the server which isolate the access to sensor data and provide the means for users to handle identifiable data. We summarize how the components in MetroTrack mitigate the various attacks in table 3.

5.3.2 *Tradeoff between privacy and estimation*

Allowing participants to establish their own privacy rules may induce noise in the estimation performed by second-level integrators from first-level integrators' sensor data. More research is needed to investigate the tradeoff between the participant's privacy rules versus the information loss in the system [38].

Table 3. MetroTrack's privacy mechanisms.

Data collection process	Privacy threat	MetroTrack privacy mechanisms
Task distribution	Re-identification attacks	Use of *Orbot client* to hide network identifiers when a *SensingTask* is downloaded Use of the *SecurityManager* in both the *MetroTrackTaskAgent* and in the *SensingTask* to encrypt data between mobile client and server
Data collection	Re-identification attacks	Use of the *TaskPrivacyManager* in the *SensingTask* module to implement data privacy algorithms (e.g., k-anonymity, l-diversity)
	Contextual attacks	The *SensingTask* cannot access sensor data directly, but through the *UserPrivacyManager* (principle of compartmentalization) Use of *UserPrivacyManager* to give options to users about which data to share with *SensingTask* *PrivacyPolicyManager* component share data with the *SensingTask* based on user's contextual rules Use of the *TaskPrivacyManager* in the *SensingTask* module to implement data privacy algorithms (e.g., k-anonymity, l-diversity)
Data submission	Re-identification attacks	Use of *Orbot client* to hide network identifiers when a *SensingTask* up- load data to the server *SecurityManager* in both the MetroTrackTaskAgent and in the Sensing-Task to encrypt data between mobile client and server
	Contextual attacks	Use of *TaskPrivacyManager* in the *SensingTask* module to implement data privacy algorithms (e.g., k-anonymity, l-diversity)
Data analysis and sharing	Re-identification attacks	Data are stored by using unique identifiers in each installation of a *SensingTask* to store and analyze data. No personal identifiable information is collected by the system from its users The use of Tor does not allow the system to

		obtain network metadata (e.g., IP addresses) about the users submitting data
	Contextual attacks	Use of the *ExternalPrivacyManager* in the *MetroTrack* server module to implement data privacy algorithms (e.g., k-anonymity, l-diversity, differential privacy)

5.3.3 *Tor as an anonymization network*

The utilization of Tor as the anonymization layer in MetroTrack does not allow to perform UDP transmission because Tor supports only TCP flows. UDP may be needed when a sensing task needs to deliver real-time sensor data to second-level integrators. As such, the evaluation of alternative solutions for providing anonymous network transfers from the point of view of privacy protection, quality of service and power consumption are needed.

5.3.4 *Layered architectural issues*

A layered architecture, as utilized in MetroTrack, may consume more power at first-level integrator devices than a cross-layer de- sign. Static wireless sensor network research has shown reduced power consumption of cross layer designs over layered designs. One possible solution to improve power consumption and at the same time enforce privacy is by using cloudlets [33] which are software modules that can be deployed in virtual machines in the cloud to offload processing from a mobile device.

5.3.5 *Choice of authentication mechanism*

The proposed authentication method requires multiple compilations for the same task because each of them will have its own hard- coded identification code. As such, the server may need additional storage as well as some type of background processing to keep enough compiled tasks available. In late 2016, it was found that the average Android app size is 15 MB [4]. Using 1TB SSD hard drive dedicated for this purpose could hold more than 6 million of these tasks.

Assuming that it takes one minute to compile an Android app and that task organizers use only a single machine with only one core to compile for 24 hours, 1440 tasks could be compiled per day. Suppose that the task organizer uses a computer with 10 cores and enough RAM to compile tasks simultaneously, up to 10,000 tasks could be compiled per day. To deploy a sensing task that could be used by every resident in the New York metropolitan area (~20 mil- lion according to the 2015 US Census), a task organizer would need 100 machines working for 2 full days to generate enough sensing tasks for each habitant, which is feasible.

6. CONCLUSION

There has been a growing interest in the development of privacy- preserving architectures for crowdsensing systems in the last few years. To handle privacy issues when developing crowdsensing systems, this work has proposed the PEAR architecture. We described the components of the architecture and we presented a prototype system called MetroTrack. Finally, we evaluated MetroTrack and discussed future research issues that require further attention for the prototype system.

7. ACKNOWLEDGMENTS

We thank the anonymous reviewers for their comments which helped us to improve the content, organization, and presentation of this work. Alfredo J. Perez was supported by the US National Science Foundation and the US Department of Defense's ASSURE program under award 1560214. Sherali Zeadally's work was supported by a University Research Professorship Award from the University of Kentucky in 2016.

8. REFERENCES

1. Aikio, J., Pentikäinen, V., Häikiö, J., Häkkilä, J., Colley, A. 2016. On the Road to Digital Paradise.
2. Al-Muhtadi, J., Campbell, R., Kapadia, A., Mickunas, M.D., Yi, S. 2002. Routing through the mist: Privacy preserving communication in ubiquitous computing environments. In Proceedings. 22nd International Conference on Distributed Computing Systems. DOI=https://doi.org/10.1109/ICDCS.2002.1022244
3. Barbeau, S.J., Labrador, M.A., Winters, P.L., Perez, R. and Georggi, N.L. 2006. A general architecture in support of interactive, multimedia, location-based mobile applications. IEEE Communications Magazine, vol. 44 no. 11. DOI=http://dx.doi.org/10.1109/MCOM.2006.248179
4. Boshell, B. 2017 Average App File Size: Data for Android and iOS Mobile Apps. Available: https://sweetpric- ing.com/blog/2017/02/average-app-file-size/
5. Campbell, A.T., Eisenman, S.B., Lane, N.D., Miluzzo, E., Peterson, R.A. 2006. People-centric urban sensing. In Proceedings of the 2nd ACM annual international workshop on Wireless internet. DOI=https://doi.org/10.1145/1234161.1234179
6. Christin, D. 2016. Privacy in mobile participatory sensing: Current trends and future challenges. Journal of Systems and Software vol. 116, 57-68. DOI=https://doi.org/10.1016/j.jss.2015.03.067
7. Christin, D., López, P.S., Reinhardt, A., Hollick, M., Kauer, M. 2013. Share with strangers: Privacy bubbles as user-centered privacy control for mobile content sharing applications. Information Security Technical Report, vol. 17 no. 3, 105-116. DOI=https://doi.org/10.1016/j.istr.2012.10.004
8. Christin, D., Roßkopf, C., Hollick, M., Martucci, L.A. Kan- here, S.S. 2013. Incognisense: An anonymity-preserving reputation framework for participatory sensing applications. Pervasive and Mobile Computing, vol. 9 no. 3, 353-371. DOI=https://doi.org/10.1016/j.pmcj.2013.01.003
9. De Capitani Di Vimercati S., Foresti, S., Livraga, G. Samarati, P. 2012. Data privacy: Definitions and techniques. Inter- national Journal of Uncertainty, Fuzziness and Knowledge- Based Systems, vol. 20 no. 6, 793-817. DOI =http://dx.doi.org/10.1142/S0218488512400247
10. Deng, L., Cox, L.P. 2009. Livecompare: grocery bargain hunting through participatory sensing. In Proceedings of the 10th workshop on Mobile Computing Systems and Applications (HotMobile '09). DOI=http://dx.doi.org/10.1145/1514411.1514415
11. Dingledine, R., Mathewson, N. and Syverson, P. 2004. Tor: The second-

generation onion router. Naval Research Lab Washington DC.

12. Dwork, C. 2008. Differential privacy: A survey of results. In Proceedings of the 5th international conference on Theory and applications of models of computation (TAMC'08), 1-19.
13. Gisdakis, S., Giannetsos, T., Papadimitratos, P. 2014. Sppear: security & privacy-preserving architecture for participatory-sensing applications. In Proceedings of the 2014 ACM conference on Security and privacy in wireless & mobile networks, 39-50. DOI = https://doi.org/10.1145/2627393.2627402
14. Gruteser, M., Grunwald, D. 2005. Enhancing location privacy in wireless LAN through disposable interface identifiers: a quantitative analysis. Mobile Networks and Applications, vol. 10 no. 3, 315-325. DOI=https://doi.org/10.1145/941326.941334
15. Hoh, B., Gruteser, M., Herring, R., Ban, J., Work, D., Herrera, J.C., Bayen, A.M., Annavaram, M., Jacobson, Q. 2008. Virtual trip lines for distributed privacy-preserving traffic monitoring. In Proceedings of the 6th international conference on Mobile systems, Applications and Services, 15-28. DOI=https://doi.org/10.1145/1378600.1378604
16. Jaimes, L.G., Vergara-Laurens, I.J., Labrador, M.A. 2012. A location-based incentive mechanism for participatory sensing systems with budget constraints. In Proceedings 2012 IEEE International Conference on Pervasive Computing and Communications (PerCom), 103-108. https://doi.org/10.1109/PerCom.2012.6199855
17. Kanjo, E., 2010. Noisespy: A real-time mobile phone platform for urban noise monitoring and mapping. Mobile Net- works and Applications, vol. 15 No. 4, 562-574. DOI=http://dx.doi.org/10.1007/s11036-009-0217-y
18. Kapadia, A., Kotz, D., Triandopoulos, N. 2009. Opportunistic sensing: Security challenges for the new paradigm. In Proceedings of IEEE 1st International Conference on Communication Systems and Networks and Workshops (COMSNETS 2009). https://doi.org/10.1109/COMSNETS.2009.4808850
19. Kazemi, L., Shahabi, C. 2011. A privacy-aware framework for participatory sensing. ACM SIGKDD Explorations Newsletter, vol. 13 no. 1. 43-51. DOI=https://doi.org/10.1145/2031331.2031337
20. Khan, WZ, Aalsalem, M.Y., Arshad, Q. 2013. Mobile phone sensing systems: A survey. IEEE Communications Surveys Tutorials, vol. 15 no. 1, 402-427. DOI=https://doi.org/10.1109/SURV.2012.031412.00077
21. Lane, N.D., Eisenman, S.B., Musolesi, M., Miluzzo, E. and Campbell, A.T. 2008. Urban sensing systems: opportunistic or participatory?. In Proceedings of the 9th workshop on Mo- bile computing systems and applications, 11-16. DOI=http://dx.doi.org/10.1145/1411759.1411763
22. Lane, N.D., Miluzzo, E., Lu, H., Peebles, D., Choudhury, T. and Campbell, A.T. 2010. A survey of mobile phone sensing. IEEE Communications magazine, vol. 48 No. 9. DOI=https://doi.org/10.1109/MCOM.2010.5560598
23. Li, N., Li, T., Venkatasubramanian, S.,2007. t-closeness: Privacy beyond k-anonymity and l-diversity. In Data Engineering, 2007. In Proceedings IEEE 23rd

International Conference on Data Engineering (ICDE 2007), 106-115. DOI=https://doi.org/10.1145/1217299.1217302

24. Lu, H., Lane, N., Eisenman, S. and Campbell, A. 2008. Bubble-sensing: A new paradigm for binding a sensing task to the physical world using mobile phones. In Proceedings of International Workshop on Mobile Devices and Urban Sensing.
25. Machanavajjhala, A., Kifer, D., Gehrke, J. and Venkitasubra- maniam, M. 2007. l-diversity: Privacy beyond k-anonymity. ACM Transactions on Knowledge Discovery from Data (TKDD), vol. 1 no1. DOI=http://dx.doi.org/10.1145/1217299.1217302
26. Mednis, A., Strazdins, G., Zviedris, R., Kanonirs, G. and Selavo, L. 2011. Real time pothole detection using android smartphones with accelerometers. In Proceedings of 2011 International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS). DOI=https://doi.org/10.1109/DCOSS.2011.5982206
27. Mendez, D., Perez, A. J., Labrador, M. A., Marron, J. J. 2011. P-Sense: A participatory sensing system for air pollution monitoring and control. In Proceedings of the 2011 IEEE International Conference on Pervasive Computing and Communications (PERCOM), 344-347. DOI=http://dx.doi.org/10.1109/PERCOMW.2011.5766902
28. Mun, M., Reddy, S., Shilton, K., Yau, N., Burke, J., Estrin, D., Hansen, M., Howard, E., West, R., Boda, P. 2009. PEIR, the personal environmental impact report, as a platform for participatory sensing systems research. In Proceedings of the 7th international conference on Mobile systems, applications, and services (MobiSys '09), 55-68.DOI=10.1145/1555816.1555823
29. Orbot: Tor for Android, Available: https://guardianpro- ject.info/apps/orbot/
30. Perez, A. J., Labrador, M. A., Barbeau, S. J. .2010. G-sense: a scalable architecture for global sensing and monitoring. IEEE Network, vol. 24, No. 4, 57-64. DOI=http://dx.doi.org/10.1109/MNET.2010.5510920
31. Perez, A.J., Zeadally, S., Jabeur, N. 2017. Investigating Security for Ubiquitous Sensor Networks, in Proceedings of the 8th International Conference on Ambient Systems, Networks and Technologies (ANT-2017). DOI= https://doi.org/10.1016/j.procs.2017.05.432
32. Perez, A.J, Zeadally, S. 2017. PEAR: A Privacy-Enabled Architecture for Crowdsensing. In Proceedings of the International Conference on Research in Adaptive and Convergent Systems (RACS '17), 166-171. DOI= https://doi.org/10.1145/3129676.3129685
33. Satyanarayanan, M., Lewis, G., Morris, E., Simanta, S., Boleng, J., Ha, K. 2013. The role of cloudlets in hostile environments. IEEE Pervasive Computing, vol. 12 no. 4, 40-49. DOI=https://doi.org/10.1109/MPRV.2013.77
34. Shi, J., Zhang, R., Liu, Y., Zhang, Y. 2010. Prisense: privacy-preserving data aggregation in people-centric urban sensing systems. In Proceedings of IEEE INFOCOM, 1-9. DOI = https://doi.org/10.1109/INFCOM.2010.5462147
35. Shilton, K., Burke, J.A., Estrin, D., Hansen, M., Srivastava, M. 2008. Participatory privacy in urban sensing. Center for Embedded Network Sensing.

36. Shin, M., Cornelius, C., Peebles, D., Kapadia, A., Kotz, D., Triandopoulos, N. 2011. AnonySense: A system for anonymous opportunistic sensing. Pervasive and Mobile Compu- ting, vol. 7. no. 1, 16-30. DOI=https://doi.org/10.1016/j.pmcj.2010.04.001
37. Sweeney, L. 2002. k-anonymity: A model for protecting privacy. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, vol. 10 no. 05, 557-570.DOI=http://dx.doi.org/10.1142/S0218488502001648
38. Vergara-Laurens, I.J., Mendez, D., Jaimes, L.G. and Labrador, M.A. 2016. A-PIE: An algorithm for preserving privacy, quality of information, and energy consumption in Participatory Sensing Systems. Pervasive and Mobile Computing vol. 32, 93-112. DOI=http://doi.org/10.1016/j.pmcj.2016.06.020

ABOUT THE AUTHORS:

Alfredo J. Perez is an Assistant Professor in the TSYS School of Computer Science at Columbus State University (Columbus, GA, USA). He received his doctoral degree in Computer Science and Engineering from the University of South Florida (Tampa, FL) and his bachelor's degree in Systems Engineering from Universidad del Norte (Barranquilla, Colombia). His research interests include privacy, Internet of Things, mobile/ubiquitous computing and sensing, and computer networks. He is a member of the U.S. National Academy of Inventors.

Sherali Zeadally is an Associate Professor in the College of Communication and Information at the University of Kentucky (Lexington, KY, USA). He received his doctoral degree in Computer Science from the University of Buckingham, England and his bachelor's degree in Computer Science from the University of Cambridge, England. His research interests include cybersecurity, privacy, Internet of Things, and computer networks. He is a Fellow of the British Computer Society and a Fellow of the Institution of Engineering Technology, England.