Towards Trustworthy Participatory Sensing

Akshay Dua, Nirupama Bulusu, Wu-chang Feng *
Portland State University
akshay@cs.pdx.edu, nbulusu@cs.pdx.edu, wuchang@cs.pdx.edu

Wen Hu
CSIRO ICT Centre, Australia
wen.hu@csiro.au

Abstract

Grassroots Participatory Sensing empowers people to collect and share sensor data using mobile devices across many applications, spanning intelligent transportation, air quality monitoring and social networking. In this paper, we argue that the very openness of such a system makes it vulnerable to abuse by malicious users who may poison the information, collude to fabricate information, or launch Sybils to distort that information. We propose and implement a novel trusted platform module (TPM), or angel based system that addresses the problem of providing sensor data integrity. The key idea is to provide a trusted platform within each sensor device to attest the integrity of sensor readings. We argue that this localizes integrity checking to the device, rather than relying on corroboration, making the system not only simpler, but also resistant to collusion and data poisoning. A "burned-in" private key in the TPM prevents users from launching Sybils. We also make the case for content protection and access control mechanisms that enable users to publish sensor data streams to selected groups of people and address it using broadcast encryption techniques.

1 Introduction

"Participatory Sensing" is a revolutionary new paradigm that allows people to *voluntarily* sense their environment using readily available sensor devices such as smart phones, and share this information using existing cellular and Internet communication infrastructure. It has tremendous potential because it harnesses the power of ordinary citizens to collect sensor data for applications spanning environmental monitoring, intelligent transportation, and public health, that are often not cost-viable using dedicated sensing infrastructure.

*This research was supported by the National Science Foundation under award 0747442.

Participatory sensing differs from traditional sensor networks in that there is typically no single data *producer* (sensor data owner). As in any participatory system, such as Wikipedia, and online recommendation systems, participatory sensing is vulnerable to gaming. Moreover, data producers and *consumers* (sensor data users) are different autonomous entities. So, producers may want to restrict whom they share their data with.

The Intelligent Transportation Systems Laboratory (ITSL) at Portland State University, studies commute statistics collected from a small number of city-owned cars in Portland, Oregon. This data is analyzed to determine routes with shorter commute times and lower congestion. It is currently very sparse, particularly for suburban neighborhoods. Citizen contributed data (like Cartel [11]) can enable much denser instrumentation. But, concerned about data integrity, Rob Bertini, ITSL Director, poses this question: "If I do not want others to take the un-congested route, isn't it better for me to tamper my commute data?", pointing to the need to ensure the authenticity of contributed sensor measurements.

The DietSense project [16] at UCLA allows users to participate in public health surveys. Users can upload images of their diet for large-scale studies to assess dietary impact on health. The concern here is to enable users to share data only with the health care experts they trust.

These applications differ inherently in whether data is collected from mobile or static sensors, or from cellular or WiFi networks, or is tagged geographically (Cartel), or demographically (DietSense), or is shared with friends, experts, or neighborhoods. Across these applications, we believe that to ensure broad participation, alleviation of the following trust concerns is paramount.

- Content Integrity: How do you have confidence that the published sensor data is indeed what was sensed?
- Content Protection: How does one ensure that only

authorized entities can access the published data?

An ideal system providing content integrity and protection should be: *scalable and adaptive* to large numbers of participating users, *efficient* and perform at a high level while not under attack, and support different sensing modalities and applications. Content integrity and protection must be provided for continuous sensor data streams, as opposed to static data items such as files.

Although integrity and access control [13] are classical Internet security problems, existing solutions will not suffice. The aforementioned Web-based participatory systems have employed different methods for preventing, detecting and responding to content integrity violations, such as reputation rankings (vulnerable to collusion), providing users incentives not to cheat (highly application specific with no guaranteed integrity behavior), model checking (requires historical data which may not be available), averaging over very large data sets (could filter out most interesting rare, sparse data) and independent human comparisons for data tagging (may not be feasible for continuous data streams). Fundamentally, the Internet was not designed with accountability in mind [1]. A major challenge, is enabling broad user participation by making the system accountable for all data.

This paper proposes a novel approach to addressing the above problems with a trusted hardware platform, which besides the main processor, consists of a trusted platform module(such as the TPM [4]), or an *angel*. The key idea is that the angel is present in each user device and is used to (i) check the integrity of contributed sensor measurements, and (ii) implement state-of-the-art cryptographic algorithms for content protection. With an angel, sensor data integrity verification is local to the data producer, and hence, this model is not only scalable, it is also inherently resilient to collusion among producers. The contributions of the paper are:

- We make the case for trustworthy participatory sensing, and motivate the problems of content integrity and protection. Current research has focused on user privacy and anonymity [12, 9, 8, 18], with little work on content integrity and protection.
- We propose, implement and evaluate a proof-of-concept *trusted hardware platform* based system, on Nokia N800 devices with SecFlecks [10], that is resilient to software compromise, efficient and scalable. To the best of our knowledge, this is the first work to propose a TPM based solution to sensor data integrity.
- We propose, implement and evaluate a proof-of-concept content protection system for participatory sensing on Nokia N800 devices. Producers forward sensed data to a trusted portal which uses broadcast encryption to efficiently encrypt data for a given set of consumers.

2 Threat Model

Our system consists of mobile user devices and a data publishing portal. Our threat model considers software-based attacks on the user device. The data publishing portal, on the other hand, is trusted for now and we defer addressing threats to its software for future work. Since our goal is enabling confidence in shared data and data sharing, we focus on risks to data integrity and protection, described in Table 1.

We do not consider widely-addressed threats that impact general Internet services, or wireless sensor networks. These include denial-of-service attacks, selective forwarding attacks, and sinkhole and wormhole attacks. We do not address unauthorized sensing of people, which requires legal policy and enforcement.

3 System Model and Approach

We assume that multiple data producers and consumers interact and exchange data through web portals. Each producer or consumer device has a tamper-proof trusted hardware element (angel). The angel only executes code signed by a trusted party, and is used to implement sensory content integrity and protection (see Figure 1).

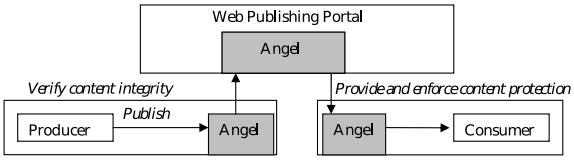

Figure 1: System Model

Why use an angel-based approach? A majority of the Internet attacks occur due to software compromises [2] and the angel removes this threat. It assures the integrity of the software running on the platform. Thus, sensors function as expected, implying that they capture data from actual events. Now, events could also be faked, sensors may get damaged, or are purposefully turned off. Currently, our system does not address these issues.

Recent advances show that trusted hardware platforms will become commonly available [7]. By providing a trusted third party within the sensor device, we can recast the problems of sensory content integrity and protection in a new way, allowing for simpler, more powerful solutions, and system accountability. For example, the problem of verifying sensor data integrity at a remote observer through indirect observation requiring complex statistical analysis and correlation to other data sources is

Risk Type	Threat	Risk Level	Risk Response	Counter Measure
Integrity	*Spoofing, Poisoning, Collusion, Sybil*: An adversary (or malware) may contribute bogus sensory data individually, in groups, or by launching Sybils.	High	Prevent poisoned data commits	Data Validation
Confidentiality	*Snooping*: An adversary (or group) may gain sensitive information by compromising data consumers. It can dynamically select the set of users to attack.	High	Prevent data decoding by unauthorized consumers	Encryption

Table 1: Threat Model Considered For Participatory Sensing

transformed to the simpler problem of verifying it at the sensor source through direct measurement. The newer integrity checking problem formulation is scalable, efficient, and can be applied to check various sensor modalities, location, time, and other attributes. Because all integrity checking is local, the system becomes resilient to collusion, and integrity violations are detected as they happen. The angel has a "burned-in" RSA private key that strongly bonds the identity of a user to his device, making it difficult to launch a Sybil.

4 Proof-of-Concept

This section describes how our current implementation uses the angel to provide *content integrity* and *protection*.

4.1 Angel

We use the Trusted Platform Module (TPM) as our angel. The TPM is a micro controller that resides on a platform (e.g. PC, laptop, mobile device) and provides it with hardware-based cryptography as well as secure storage for sensitive credentials. These credentials can then be used at a later time to authenticate the platform, or attest to its integrity.

As stated in our system model, (Section 3) producers and consumers carry trusted hardware platforms, envisioned to be the existing compact mobile devices that people carry (e.g. smart phones). Unfortunately, there are no commercially available compact mobile devices with built-in TPMs. To provide existing devices with trusted capabilities, we have developed a *trusted sensing peripheral* (Figure 2) that interfaces with them via a Bluetooth connection. In our implementation, the mobile device used is a Nokia N800 Internet Tablet with 128 MB of memory running Linux kernel version 2.6.21 on an ARMv6 processor.

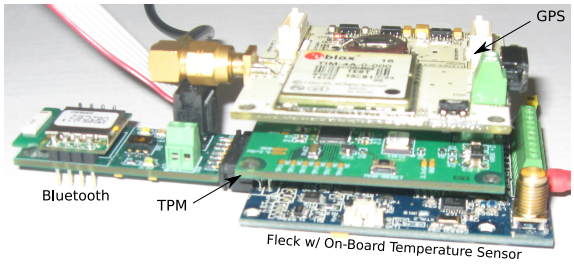

Figure 2: Trusted sensing peripheral with GPS sensor

4.2 Trusted Sensing Peripheral

We use the secFleck [10] as our trusted sensing peripheral. The secFleck is a small device consisting of an Atmel TPM chip mounted on a Fleck sensor board. The TPM chip is based on version 1.2 of the Trusted Computing Group (TCG) specification [4] and uses a 2048 bit key for RSA operations. The RSA private key is "burned in" to the TPM chip and never exposed, thus signatures created by the TPM are irrefutable.

The secFleck has 8 KB of memory and an 8 MHz Atmega micro controller. It has an on-board temperature sensor and can extend sensing functionality by stacking multiple sensors, such as GPS, camera, and microphone. Also attached to the secFleck is a Parani-ESD hardware module that adds wireless serial communication capabilities using Bluetooth technology, to enable communications between the secFleck and the associated Nokia Internet Tablet. Figure 2 shows the secFleck with an attached Bluetooth hardware module and a GPS sensor.

4.3 Content Integrity

Any solution to content integrity must ensure that the data published, is indeed the data sensed by the mobile producer device, even when the producer is malicious. The TPM provides platform attestation capabilities that the trusted sensing peripheral uses to assure the publishing portal (see Figure 1) about the integrity of its plat-

form. In this process, called *platform attestation*, the TPM vouches that the software running on the peripheral has not been modified in an unintended manner. This software includes the Fleck OS and the device drivers associated with the attached sensors and Bluetooth adapter.

The producer's mobile device, is responsible for tasking the trusted sensing peripheral and being a communication proxy between the portal and the peripheral. The TPM in the sensing peripheral also attests the integrity of the data collected by the attached sensors. This process, called *data attestation* begins with the TPM signing a SHA-1 digest of the collected data. The data is then sent to the portal along with the TPM's signature and the associated digest. The portal can then verify the source and integrity of all the data it receives from the peripheral.

Our current implementation includes the tasking, collection and attestation of data but not the platform attestation process. We discuss briefly how platform attestation will be implemented. At start-up, the TPM in the trusted sensing peripheral will perform an integrity measurement, in the form of a SHA-1 digest, of all the software components. The measurement will then be stored securely within one of TPM's Platform Configuration Registers (PCRs). When the portal sends a request for attestation, the TPM will sign the digest contained in its PCR and send the signature along with the digest back to the portal. The portal will then verify the signature to ensure that the received digest matches the one corresponding to the respective trusted sensing peripheral configuration. The portal must know a priori the digest of a particular peripheral configuration.

Once *tasked*, the trusted peripheral sends a *task response* after each *sensing interval*. The response message contains a sequence number to prevent replay attacks, collected data, a 20 byte SHA-1 digest and a 256 byte RSA signature.

4.4 Content Protection

A producer often needs to share data with multiple consumers. A naive and wasteful option is to encrypt the same data separately for each consumer. A more efficient way is to use broadcast encryption, allowing a producer to simultaneously encrypt data for *any* set of consumers, while also providing cryptographically enforced access control.

To that end, we have implemented the *Augmented Broadcast Encryption (ABE)* scheme [6] using the Pairing Based Crypto (PBC) library developed at Stanford University by Benn Lynn [14]. ABE is a public-key broadcast, trace and revoke system that is fully collusion resistant, secure against adaptive adversaries and publicly traceable. It requires short cipher-texts and private keys of $O(\sqrt{N})$, where N is the number of users in the system.

According to [6], if public traceability is not required, and the tracing assumptions of [5] are used, the system can still be proven secure and private keys optimized to $O(1)$. This can easily be implemented with the trusted sensing peripheral, which can hold the secret tracing key in the TPM's sealed storage. Additionally, the ABE private keys belonging to the user could also be held by the TPM. The strong binding of keys to the TPM ensures that only the associated mobile device can access the keys and decrypt the data received from the portal.

5 Evaluation

We evaluated our content integrity and protection prototypes to explore their feasibility.

Experiment	Conclusion
Attestation Code size	13 Kbytes
Data Attestation time	1.92 ($\pm$ 0.01) secs for 100 bytes
Data Verification time	0.78 ($\pm$ 0.001) secs for 100 bytes
Energy cost of computing an RSA signature	11.06 mJ
Bluetooth Communication overhead	0.79 ($\pm$ 0.001) secs for 100 bytes

Table 2: Summary of results

5.1 Content Integrity

We extend the trusted sensing peripheral with an onboard temperature sensor and GPS. Our goal is to determine the energy and latency costs, for attestation (includes RSA signature computation) and verification of sensor data by the peripheral.

We consider three task schedules: 1, 10, and 100 sensor readings taken at one second intervals. Each reading is 10 bytes, consisting of two bytes of temperature data and 8 bytes of GPS data. Every schedule is repeated 40 times, giving us a large enough sample to calculate a 95% confidence interval without making assumptions about the data distribution. A summary of our results can be seen in Table 2. The energy required to compute an RSA signature is constant as the signature is always computed over a fixed-size digest of the readings. We plan to explore other modes of communication with the trusted sensing peripheral to reduce the Bluetooth communication overhead.

5.2 Content Protection

Although our ABE implementation has not been integrated with the trusted sensing peripheral yet, we have

measured its performance on the Nokia N800 mobile platform. We implemented ABE as a Key Encapsulation Mechanism (KEM) [5] in which the publishing portal (broadcaster) periodically encrypts and broadcasts a symmetric group session key (DES, 256 bits) to all the users of the system — one of which is the Nokia N800. Since this is an evaluation from the client perspective, we focus only on decryption performance.

Figure 3 plots the private key size for different maximum user counts the ABE scheme is setup with. The user count is an initialization parameter, not necessarily the number of active users. While large, the private key storage required is reasonable for a Nokia N800 class device.

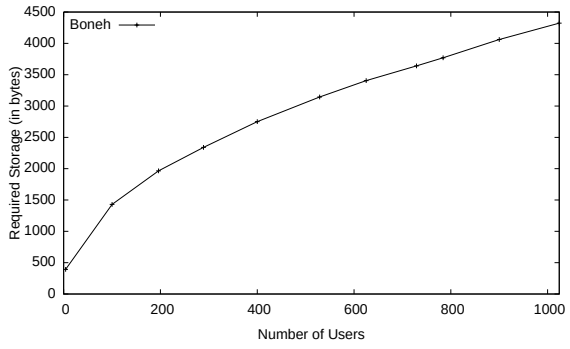

Figure 3: Storage required for ABE private key

Table 3 shows the code size, mean time and energy required for decryption. The energy and time required for ABE decryption are prohibitive, but because it is used as a KEM, decryption will be performed rarely. Decryption time is constant, and independent of the number of users.

Metric	Value
Code Size (ABE library)	13 Kbytes
Decryption Time	8.8 secs
Decryption Energy	6.63 J

Table 3: ABE Client-side requirements

6 Discussion

Our approach is in contrast to how the TPM is generally used: as part of the device itself, attesting the load-time integrity of the mobile device platform and also providing secure storage. But participatory applications collecting data continuously are also vulnerable to run-time compromises. With proper hardware support (e.g AMD SVM [3]) and a system like Flicker [15], run-time protection is possible, but most mobile platforms do not provide this support. Pioneer [17] does not require special hardware for platform attestation, but is sensitive to, and requires prior knowledge of the platform's computational capability.

Our system mitigates run-time compromise by limiting access to the trusted sensing peripheral. Besides Bluetooth access for a small command set (to interface with mobile devices), the peripheral provides no external interface. Applications cannot be installed or removed at run-time. The only way to change its functionality is to update its firmware using a direct physical connection.

7 Summary and Future Research

This paper made the case for developing trustworthy participatory sensing applications using a trusted sensing peripheral to provide sensory content integrity and protection. The trusted sensing peripheral attests the sensor data integrity at the source. Local integrity checking is inherently resilient to collusion, making it difficult for an adversary to fabricate data, or launch a Sybil.

While evaluations demonstrate the practical potential of our system, we plan to explore several key research challenges in future work. For content protection, we hope to replace the trusted data dissemination portal with an untrusted network between mobile devices, requiring them to perform the broadcast encryption step themselves. We hope to optimize broadcast encryption to work on mobile devices so that producers can directly encrypt data for a given set of consumers. This will result in a secure, efficient, peer-to-peer data dissemination.

For data integrity, our approach attests the integrity of raw sensor data. When it is desirable for applications to process raw data at the mobile device to extract high-level features, we recommend and plan to implement attesting both the application and mobile platform. Secondly, not every user may have a trusted sensing peripheral. In this case, we want to explore whether the trusted data could be used to clean or validate the untrusted data, and if so what fraction of trusted contributors is required. Finally, an angel cannot protect against sensor measurements being corrupted by physical sensor damage, or bio-fouling. We hypothesize that statistical data cleaning methods might be more effective in conjunction with our collusion-resistant data integrity framework. If true, it will simplify detection of such errors, and is a major goal for our long-term research.

References

[1] Assurable Global Networking Workshop. `http://csc-ballston.dmeid.org/darpa/registration/agn_proceedings.htm`.

[2] CERT Advisories. http://www.cert.org/advisories/.

[3] SVM: AMD's Virtualization Technology. www.xen.org/files/xs0106_amd_virtualization.pdf.

[4] Trusted computing group. https://www.trustedcomputinggroup.org/home.

[5] D. Boneh, A. Sahai, and B. Waters. Fully collusion resistant traitor tracing with short ciphertexts and private keys. In *Proceedings of Eurocrypt*, volume 4004, pages 573–592, Saint Petersburg, Russia, 2006. Springer.

[6] D. Boneh and B. Waters. A fully collusion resistant broadcast, trace, and revoke system. In *Proceedings of ACM CCS*, pages 211–220, Alexandria, Virginia, 2006. ACM.

[7] B. Colwell. Keynote talk: Computing Architecture Futures 2007. In *FCRC*, San Diego, California, 2007.

[8] R.K. Ganti, N. Pham, Y.E. Tsai, and T.F. Abdelzaher. PoolView: stream privacy for grassroots participatory sensing. In *Proceedings of ACM SenSys*, pages 281–294, Raleigh, North Carolina, 2008. ACM.

[9] B. Hoh, M. Gruteser, R. Herring, J. Ban, D. Work, J. Herrera, A. M. Bayen, M. Annavaram, and Quinn Jacobson. Virtual trip lines for distributed privacy-preserving traffic monitoring. In *Proceedings of ACM MobiSys*, pages 15–28, Breckenridge, Colorado, 2008. ACM.

[10] W. Hu, P. Corke, W. C. Shih, and L. Overs. secfleck: A public key technology platform for wireless sensor networks. In *Proceedings of EWSN*, pages 296–311, Cork, Ireland, 2009.

[11] B. Hull, V. Bychkovsky, Y. Zhang, K. Chen, M. Goraczko, A. Miu, E. Shih, H. Balakrishnan, and S. Madden. Cartel: a distributed mobile sensor computing system. In *Proceedings of ACM SenSys*, pages 125–138, Boulder, Colorado, 2006. ACM.

[12] A. Kapadia, N. Triandopoulos, C. Cornelius, D. Peebles, and D. Kotz. AnonySense: Opportunistic and Privacy-Preserving Context Collection. *LNCS*, 5013:280, 2008.

[13] P. A. Karger. Non-discretionary access control for decentralized computing systems. Technical Report TR-179, MIT, Cambridge, Massachusetts, 1977.

[14] B. Lynn. PBC library. *Online: http://crypto.stanford.edu/pbc/*.

[15] J.M. McCune, B.J. Parno, A. Perrig, M.K. Reiter, and H. Isozaki. Flicker: An execution infrastructure for TCB minimization. In *Proceedings of ACM SIGOPS/EuroSys*, pages 315–328, Glasgow, Scotland, 2008. ACM.

[16] S. Reddy, A. Parker, J. Hyman, J. Burke, D. Estrin, and M. Hansen. Image browsing, processing, and clustering for participatory sensing: lessons from a DietSense prototype. In *Proceedings of ACM SenSys*, pages 13–17, Cork, Ireland, 2007. ACM.

[17] A. Seshadri, M. Luk, E. Shi, A. Perrig, L. van Doorn, and P. Khosla. Pioneer: verifying code integrity and enforcing untampered code execution on legacy systems. *Proceedings of ACM SIGOPS Operating Systems Review*, 39(5):1–16, 2005.

[18] K. Shilton, J.A. Burke, D. Estrin, M. Hansen, and M. Srivastava. Participatory Privacy in Urban Sensing. In *Proceedings of the MODUS Workshop*, St. Louis, Missouri, 2008.