PRIVACY-AWARE COLLABORATION AMONG UNTRUSTED RESOURCE CONSTRAINED DEVICES

A Dissertation Presented

by

ANDRES DAVID MOLINA-MARKHAM

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

February 2013

Department of Computer Science

PRIVACY-AWARE COLLABORATION AMONG UNTRUSTED RESOURCE CONSTRAINED DEVICES

A Dissertation Presented

by

ANDRES DAVID MOLINA-MARKHAM

Approved as to style and content by:

Kevin Fu, Chair

Prashant Shenoy, Member

Gerome Miklau, Member

Wayne Burleson, Member

Lori Clarke, Department Chair
Department of Computer Science

To Elizabeth and my parents.

ACKNOWLEDGMENTS

I would like to thank in particular my advisor Kevin Fu and my thesis committee readers Prashant Shenoy, Gerome Miklau and Wayne Burleson for their service and feedback on this dissertation.

I also thank Kevin Fu for his guidance and support throughout my doctoral program. I thank Prashant Shenoy for the collaboration and multiple fruitful discussions that resulted in significant contribution to this dissertation. I also appreciate the opportunity of working with Wayne Burleson on interesting projects beyond the scope of this dissertation. Finally, I learned a great deal about privacy from Gerome Miklau in his seminar and multiple conversations we had during my program at UMass.

I would also like to express my gratitude to all the members of my research group, SPQR, for their feedback, collaboration, and productive discussions throughout the years, particularly Shane Clark, Ben Ransford, and Negin Salajegheh, who also co-authored papers with me. I would also like to thank David Irwin, Emmanuel Cecchet, and Aditya Mishra for their collaboration and assistance with the smart metering related work. Also, thanks to Deepak Ganesan for the discussions related to networking CRFIDs.

I am grateful for the guidance provided by George Danezis on privacy-preserving smart metering. Thanks to Ari Juels for the various discussions that motivated the work on CRFIDs. Also, I appreciate the opportunity to collaborate with researchers at the University of California, Berkeley, the Beth Israel Deaconess Medical Center, MIT and RSA Laboratories, particularly Dawn Song, Steve Hanna, and Daniel Kramer.

Finally I would like to express my appreciation toward the Department of Computer Science at the University of Massachusetts Amherst for providing me with a place to grow and develop as a researcher.

Portions of the dissertation appeared in the following publications:

"HICCUPS: Health Information Collaborative Collection Using Privacy and Security" by A. Molina, M. Salajegheh, and K. Fu. In Proceedings of the ACM Workshop on Security and Privacy in Medical and Home-Care Systems. November 2009.

"Private Memoirs of a Smart Meter" by A. Molina-Markham, P. Shenoy, K. Fu, E. Cecchet, and D. Irwin. In 2nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings. November 2010.

"Designing Privacy-preserving Smart Meters with Low-cost Microcontrollers" by A. Molina-Markham, G. Danezis, K. Fu, P. Shenoy, and D. Irwin. In Proceedings of the 16th International Conference on Financial Cryptography and Data Security. February 2012.

"BAT: Backscatter Anything-to-tag Communication" by A. Molina-Markham, S. S. Clark, B. Ransford and K. Fu. Chapter in Wirelessly Powered Sensor Networks and Computational RFID. Springer Signals and Communication. J. R. Smith (Ed.) December 2012. To appear.

The work in this dissertation was supported in part by NSF grants CNS-831244, CNS-0845874, CNS-0964641, CNS-0627529. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation. This work was also partially funded by a Sloan Research Fellowship and a Graduate Fellowship.

Family and Friends

I express my gratitude to all my friends and family, particularly to Elizabeth who has provided caring and support throughout the years. Gracias a mis padres

Guadalupe y Moisés por su apoyo incondicional. Gracias a mis hermanos Moisés y Eric por todos los buenos momentos que pasamos desde niños y por la ayuda que me han brindado durante mis estudios fuera de México. I also thank Anne and Bill for their help and for embracing me into the family.

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

PRIVACY-AWARE COLLABORATION AMONG UNTRUSTED RESOURCE CONSTRAINED DEVICES

FEBRUARY 2013

ANDRES DAVID MOLINA-MARKHAM

Ph.D., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Professor Kevin Fu

Individuals are increasingly encouraged to share private information with service providers. Privacy is relaxed to increase the utility of the data for the provider. This dissertation offers an alternative approach in which raw data stay with individuals and only coarse aggregates are sent to analysts. A challenge is the reliance on constrained devices for data collection. This dissertation demonstrates the practicality of this approach by designing and implementing privacy-aware systems that collect information using low-cost or ultra-low-power microcontrollers. Smart meters can generate certified readings suitable for use in a privacy-preserving system every 10 s using a Texas Instruments MSP430 microcontroller. CRFIDs—batteryless devices that operate on harvested energy from RF—can generate encrypted sub-aggregates in 17 s to contribute to a privacy-preserving aggregation system that does not rely on a trusted aggregator. A secure communication channel for CRFID tags via untrusted relays achieves a throughput of 18 Kbps.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

Page

Acknowledgments **v**

ABSTRACT **ix**

List of Tables **xv**

List of Figures **xvii**

1 Introduction **1**

1.1 Thesis Statement 2
1.2 Contributions 3
1.3 Privacy-Preserving Smart Metering 3
1.4 RFID-Scale Device Communication via Untrusted Relays 4
1.5 Privacy-Preserving Aggregation with RFID-Scale Devices 6
1.6 Organization 6

2 Background and Definitions **9**

2.1 Constrained Embedded Systems 9
2.2 Distributed Model for Data Collection and Analysis 10
2.3 Zero-Knowledge Proof Systems 12

2.3.1 Cryptographic Commitments 13
2.3.2 Camenisch-Lysyanskaya Signatures 14

2.4 Distributed Differential Privacy 15

2.4.1 Differential Privacy with a Trusted Aggregator 16
2.4.2 Differential Privacy with an Untrusted Aggregator 17

2.5 Secrecy and Integrity for Networking 19

2.5.1 Symmetric Key Primitives 19
2.5.2 Public Key Primitives 20
2.5.3 Identity-Based Encryption 20

3 Privacy Issues of Smart Metering 23

3.1 Smart Metering . . . 25
3.2 Dynamic Pricing and Optimizing Distribution . . . 26
3.3 Implications of Privacy Leakage through Smart Metering . . . 26
3.4 Methodology to Estimate Privacy Leakage . . . 27
3.5 Results . . . 30
3.6 Conclusion . . . 35

4 Privacy-Preserving Smart Metering with Low-cost Microcontrollers 37

4.1 A Zero-Knowledge Proof System for Billing . . . 39
4.2 Background on Microcontrollers . . . 41
4.3 Anatomy of a Smart Meter . . . 42
4.4 Meter Design Variables . . . 44
4.5 A Privacy-Preserving Smart Meter . . . 44
4.6 Experimental Evaluation . . . 47

4.6.1 Impact of Platform Selection . . . 47
4.6.2 Impact of Multitasking Approach . . . 47
4.6.3 Impact of ECC Utilization . . . 48
4.6.4 Impact of Signature Scheme Selection . . . 49

4.7 Feasibility and Costs in Real-World Deployments . . . 49

4.7.1 Cost Estimation Strategy . . . 51
4.7.2 Economic Feasibility . . . 51

4.8 Related Work . . . 53
4.9 Conclusion . . . 54

5 BAT: Backscatter Anything-to-Tag Communication 56

5.1 Anything-to-Tag Communication . . . 59
5.2 BAT Design Overview . . . 60
5.3 Applications . . . 64
5.4 BAT Evaluation . . . 70

5.4.1 Prototype Implementation . . . 70
5.4.2 BAT's Throughput . . . 70

5.5 Using Untrusted Relays . . . 73

5.5.1 Performance on Future CRFID Prototypes . . . 74
5.5.2 Shared-Key Generation . . . 75

5.6 Related Work . . . 77
5.7 Conclusion . . . 77

6 Privacy-Preserving Aggregation using Constrained Devices 80
6.1 Privacy-Preserving Aggregation with an Oblivious Aggregator 81
6.2 A Mechanism for Privacy-Preserving Aggregation 82
6.2.1 Simple Model Assumptions 82
6.2.2 Basic Construction 83
6.3 Implementation Using RFID-Scale Devices 85
6.3.1 Implementation Details 85
6.3.2 Measurements 86
6.3.3 Additional System Considerations 87
6.4 Fault Tolerance 89
6.4.1 Utility 89
6.5 Related Work 90
6.6 Conclusion 91

7 Privacy-Preserving Aggregation of Medical Telemetry 92
7.1 Case for a Distributed Model for Aggregating Medical Telemetry 93
7.2 HICCUPS 97
7.2.1 Background on Homomorphic Encryption 98
7.2.2 Access Model 99
7.2.3 Threat Model 100
7.2.4 Desired Properties 101
7.2.5 Design 101
7.3 Computing Aggregates through Counting Queries 105
7.4 Defining Evaluation Metrics 110
7.5 Related Work 113
7.6 Conclusion 117

8 Future Work 118
8.1 Privacy-Preserving Dynamic Queries for Smart Metering 119
8.2 Smart Phones and Other Personal Data Relays 120
8.3 Transportation Payments Using CRFIDs 121

9 Conclusions 124

BIBLIOGRAPHY 126

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

Table **Page**

3.1 Private questions and answers that fine-grained power consumption data reveals. 27

4.1 Running time of commitments and signatures across multiple platforms. The tasks are run exclusively and uninterrupted on each of the platforms. The signatures are performed on 16 bytes of data. DSA uses a 1,024-bit prime p, a 160-bit prime q, and SHA-256. The timing does not include the generation of randomness, which depends on the source. Prices are in USD (Sept., 2011). 48

4.2 Running time of commitments (single reading) and signatures (4 reading batches) on an MSP430F5438A at 25 MHz. These times are obtained when the algorithms are running exclusively and uninterrupted. Miracl is used for the elliptic curve versions (§4.5). The key sizes are in bits. 50

4.3 RAM utilization for the various algorithms we implement on an MSP430F5438A all using the Miracl library. The measurements do not include RAM utilization by an RTOS, a radio stack or I/O. 50

5.1 CRFIDs differ from supply-chain tags in that CRFIDs can manage their own memory and application logic. CRFIDs also differ from motes in that CRFIDs have shorter power cycles and depend on RFID readers for power and communication. 58

5.2 The cycle count for each security operation was measured using a hardware debugger and a UMass Moo. The reported time is calculated assuming a typical sending clock speed of 4 MHz. 73

5.3 Computation times for cryptographic operations on the following MCUs: (1) MSP430F5310 @ 8 MHz; (2) ARM Cortex-M0+ @ 30 MHz; (3) ARM Cortex-M0 @ 50 MHz. 75

5.4 Computation times for key agreements on the MSP430F2618 @ 4 MHz. All multi-precision arithmetic, EC arithmetic, and pairings are implemented by Miracl [32]. The Diffie-Hellman key exchange based on the ECDLP uses the NIST curve P-192. The Type-1 pairings use supersingular curves $E(\mathbb{F}_p)$ for a 512-bit prime and $E(\mathbb{F}_{2^{379}})$. 76

6.1 The cycle count for encrypting a noisy value ($c_i = g^{\hat{x}_i} \cdot \mathbb{H}(t)^{sk_i}$) measured using a hardware debugger and a UMass Moo. The underlying hash function was derived using SHA256. As a reference, the second column lists a block cipher with comparable strength based on the key size [112, 23]. The reported time is calculated assuming a clock speed of 4 MHz, typical when using harvested energy, and 16 MHz, the maximum when an additional source of power is available. The RAM requirements do not include the networking stack or any other additional application logic and are measured using IAR's IDE and compiler tools V.5.40.7. 86

7.1 Estimated overhead added by *HICCUPS* for performing a simple aggregation with 100 caregivers with 1000 records each. The table shows the overheads using four different primitives: securely equivalent RSA-1024 and ECC-160, as well as RSA-2048 and ECC-224. 113

LIST OF FIGURES

Figure **Page**

1.1 Smart meters are an example of constrained devices that gather potentially private data about individuals. They are implemented using low-cost microcontrollers that are significantly less capable than processors in personal computers. Bottom-right: Daily power trace of a home at a 30 second granularity. Top-right: Period corresponding to the same home from 8:34 am to 8:58 am at 1 second granularity. Automatically labeled power segments are represented by different colors. Appliance labels correspond to logged activities. Meter photographs [3]. 5

2.1 Distributed model for data collection and analysis. Individuals on the left produce data with the purpose of computing a combining function across individuals or across time. Each individual may not want to share his/her data with anyone in order to perform this calculation to preserve his/her privacy. A function may be performed on data from a single individual corresponding to a period of time or data corresponding to multiple individuals. 11

2.2 Techniques to provide privacy, implemented at different times in the lifecycle of the data, from the point the data is collected to the point some data is presented to an analyst 16

2.3 Slight modification to the general collaboration model in which a device or entity may be selected to aggregate without being trusted. 18

3.1 Smart meter installed in the state of Pennsylvania in 2012. 25

3.2 Architecture using a TED monitor as a smart meter. 28

3.3 Example day-long second-level power trace with labels from the day's activity log. 29

3.4 Identification of human presence with high probability for each day of the month. 29

3.5 Low power periods correspond to little human activity over a two-month trace for one home. 30

3.6 Power signatures for a dehumidifier and an air re-circulator. Note that the dehumidifier shuts off after it fills up. 32

3.7 Power segments from eating breakfast. The clustering algorithm automatically generates the color scheme. The labels are from the activity logs. 33

3.8 Example of the power segments from taking a shower, including labels from our activity logs. 33

3.9 An example of the same power segments from Figure 3.7, but at a 30 second logging granularity. 34

4.1 Architecture of the privacy-preserving smart metering system. A smart meter, in addition to its metrologic unit, has a microcontroller capable of encrypting and certifying its readings. The meter also has a wireless transceiver used to send encrypted readings to the consumer's device. The consumer uses the information from the meter for consumption planning, and in the computation of bills and corresponding proofs. 39

4.2 This graph provides a visual representation of performance improvements as seen across a few popular architectures. The trends in microprocessors targeting desktop computers and servers, as well as the performance improvements observed in ARM application microprocessors have followed exponential curves. However, the performance improvements observed in embedded ARM microprocessors and MSP430 microcontrollers have followed linear curves [10, 35, 80]. Note that a comparison based on microprocessors using millions of instructions per second does not capture all qualities of a microprocessor, but it helps to illustrate general trends. 43

4.3 Main components of a smart meter. On the left: A simple meter with a single microcontroller unit (MCU) that controls the metrologic unit, storage and communication interfaces. On the right: A smart meter that replaces the analog front end with an embedded signal processor (ESP) and has an additional application processor that controls communication, OS, power monitoring, and analytics. 43

4.4 Memory requirements on an MSP430F5438A. 52

4.5 Impact of ECC on computation using an MSP430F5438A. 53

5.1 BAT Overview: Relays collect packetized messages from tags and forward them through other relays, which deliver them. Packets are split into frames locally to maximize throughput. 59

5.2 Hardware used to implement BAT. The relay (foreground) is a USRP with RFX900 daughterboards driven by GNU Radio. The UMass Moo (background) is a CRFID tag derived from the DL WISP 4.1 [135]. 61

5.3 Gen 2 requires messages to achieve singulation before Read/Write commands are issued. In order to implement custom round-trip messages as in BAT ($R \rightarrow T$, $T \rightarrow R$), a Write command would have to be followed by a Read command. 62

5.4 The *framed* packet format accommodates up to $2^8 = 256$ bytes of data payload per packet by breaking packets into one or more frames of variable size. 63

5.5 BAT messages. A tag may request a relay to: *deliver a message*, *provide power*, or *check for messages addressed to the tag*. In some cases singulation may be necessary, but not always. 64

5.6 Bridge with a monitoring system in Flint, MI. Data is collected and transmitted to a remote computer once per hour. System and photo by Fondriest Environmental [60]. 66

5.7 Roadway monitoring via video cameras [141]. The use of CRFIDs could allow for the monitoring of larger areas with less maintenance and infrastructure than video monitoring. 69

5.8 The throughput of BAT depends on the size of frames. Shorter frames result in an increased number of round trips, overhead in frame responses, and inter-frame processing. Larger frames are more likely to result in a corrupted frame. The figure shows the average throughput accounting for retransmissions using different frame sizes. The maximum throughput observed in the current prototype is 18 Kbps, achieved when using a 112-byte frame. Larger frames significantly reduce throughput due to larger error rates. BAT uses Tari = 13 μs. Frame size does not significantly affect throughput when a Gen 2 M5e reader is used with two different Gen 2 high-capacity tags because of their low success rate—the number of attempts required for a write increases proportionally with the frame size. The M5e reader uses Tari = 12.5 μs. Tags are placed approximately an inch from the antenna. 71

6.1 Computing a sum via a distributed model to achieve differential privacy. 83

7.1 An alternative trust model would place researchers—including device manufacturers—in a position where they can submit privacy preserving queries. Only patients and their caregivers have full access to patients' telemetry. 94

7.2 Data flow that prevails currently. Patients upload their telemetry to the manufacturer's servers, from which it is made available to caregivers for analysis. 95

7.3 The current trust model requires that a manufacturer gather all patient data and deliver it to their corresponding caregivers. Patients do not currently have access to their data. 96

7.4 The figure illustrates the data flow that prevails currently. Patients upload their telemetry to the manufacturer's servers, from which it is made available to caregivers for analysis. 98

7.5 A researcher or manufacturer needs to compute an aggregate function from data across various caregivers. The query can be handled by an aggregator chosen among the caregivers that computes on encrypted data. 102

7.6 The aggregator is chosen at random to eliminate the probability of a compromised aggregator systematically leaking data. The rest of the caregivers compute sub-aggregates, which can be combined by the aggregator to produce a total aggregate for the manufacturers and researchers. 103

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 1

INTRODUCTION

This dissertation proposes a model for privacy-aware data collection systems that rely on embedded devices with low-cost or ultra-low-power microcontroller units. In these systems data remain with individuals who generate the data. Coarse grained aggregates are provided to analysts or service providers.

Common practice in data collection relies on sending raw data to a trusted aggregator, which analyzes and stores them. Individuals whose data are being collected have to trust this aggregator to protect their data from others and to respect their privacy by not analyzing the data beyond what is needed to provide a service. Another limitation of this model from the privacy standpoint is that once the information from an individual has been leaked, the individual no longer has control over its use.

The approach in this dissertation challenges current methods by removing the requirement of a trusted aggregator and by demonstrating feasibility on constrained devices.[1] It demonstrates that privacy-preserving techniques can be practically implemented using constrained devices such as those that depend on low-cost or ultra-low-power microcontrollers. Research over the last couple of years has converged in providing a more formal framework for quantifying privacy [49] and in developing

[1]For the purposes of this dissertation, a constrained device is an embedded system that relies on an ultra-low-power or low-cost microcontroller unit. In 2012 low-cost microcontrollers range in price from $0.25 to $9.00 USD Ultra-low-power microcontrollers consume under 1mW/MHz. For example, the Texas Instruments MSP430F2618 can consume as little as 803 μW/MHz; the ARM Cortex-M0+ as little as 11.2 μW/MHz; and the ARM Cortex-M0 as little as 16 μW/MHz. Microcontrollers in this class typically operate at a frequency of around 25 MHz and utilize 32-bit, 16-bit, or 8-bit architectures. MCUs can go as high as 100 MHz.

cryptographic techniques to support privacy-aware aggregation [140, 33, 37, 145]. The feasibility of the approach in this dissertation draws from these recent developments, providing a basis for closing the gap between theory and implementation.

The issue of privacy in data collection is increasingly critical with the spread of embedded systems that collect personal data. One example is the worldwide deployment of smart meters—76 million as of 2010 [116]. In the U.S. alone, over 36 million smart meters have been installed as of 2012, with 675,000 being installed monthly. It is estimated that by 2015 there will be 65 million [79] in the U.S. These smart meters rely on low-cost microcontrollers. Technological trends suggest that constrained devices will continue to be involved in data collection applications for a long time. One reason is that battery technologies and energy efficiency are not evolving at the same rate as the proliferation of data collection applications. Also, a new class of batteryless devices offers a promising solution for long-term and low-maintenance deployments [24]; these devices will continue to be subject to severe power constraints.

The successful implementation of the model presented here would allow individuals to gain control over their data. Systems for collection of private data could be implemented following a principle similar to the *least privilege principle* in security [134]: detailed data would not be shared, unless it were strictly necessary to receive a service. Future uses of previously collected data by utilities would be more difficult. For example, finding out that utility metering using measurements every second reveals sensitive information would not be an issue if data had remained with the consumer and a utility had only received the total dollar amount owed.

1.1 Thesis Statement

The work in this dissertation provides evidence to support the following thesis:

A model for performing distributed privacy-preserving computations without relying on trusted aggregators can be practically implemented on embedded systems that rely on low-cost or ultra-low-power microcontrollers.

A careful combination of cryptographic techniques and distributed system techniques may enable ubiquitous data collection of private information, such that individuals achieve adequate privacy guarantees and analysts obtain information with adequate utility. This dissertation challenges the idea that the best model for dealing with embedded constrained devices that collect or generate data is one in which devices pass data verbatim onto a more powerful system for aggregation and analysis. While this sink model may offer many benefits, such as increased storage and computational capabilities, it may not always be the most appropriate for implementing privacy-aware applications.

1.2 Contributions

The validity of the thesis is demonstrated by implementing each of the following systems:

1. Privacy-preserving smart metering with low cost microcontrollers
2. RFID-scale device communication via untrusted relays
3. Privacy-preserving aggregation with RFID-scale devices

This dissertation argues for the potential generalization of these systems to other applications and suggests research directions. It also identifies limitations of the approach, such as providing fault tolerance in aggregations and performing complex analyses with high utility.

1.3 Privacy-Preserving Smart Metering

A solution to the privacy issue of smart metering relies on the use of Zero-Knowledge Proof (ZKP) systems to compute billing and other aggregates from a power trace [108, 126]. However, current smart meters are implemented using low-cost microcontrollers such as the Texas Instruments MSP430, which has severe computational, storage, and memory limitations.

An approach for the implementation of these ZKP systems in constrained devices utilizes specific elliptic curve based primitives, which minimize the computational and memory footprints in a given microcontroller. The practicality of these systems is evaluated by determining the extent to which a ZKP system could be deployed on current smart meters and by measuring the performance that should be expected using newly developing microcontroller and RAM technologies.

A prototype meter equipped with a microcontroller like those in current smart meters is capable of producing certified readings for ZKP systems every 28 seconds. If a newer $3.30 USD MSP430 microcontroller is used, readings can be produced every 10 seconds.

1.4 RFID-Scale Device Communication via Untrusted Relays

Computational RFID tags have evolved from traditional supply-chain RFID tags, adding a general purpose microcontroller and sensors, which make them well-suited platforms for developing networks of batteryless nodes. However, the abstractions provided by current RFID communication are not well suited for CRFIDs. The current RFID standard of communication treats tags as external memory locations for *reading* or *writing* data. This severely limits the networking capabilities of CRFIDs.

Backscatter Anything-to-Tag (BAT) communication provides an alternative approach for networking this class of devices. In this networking stack, tags can send

Privacy Challenge

Fine-grained data leaks private information about individuals.

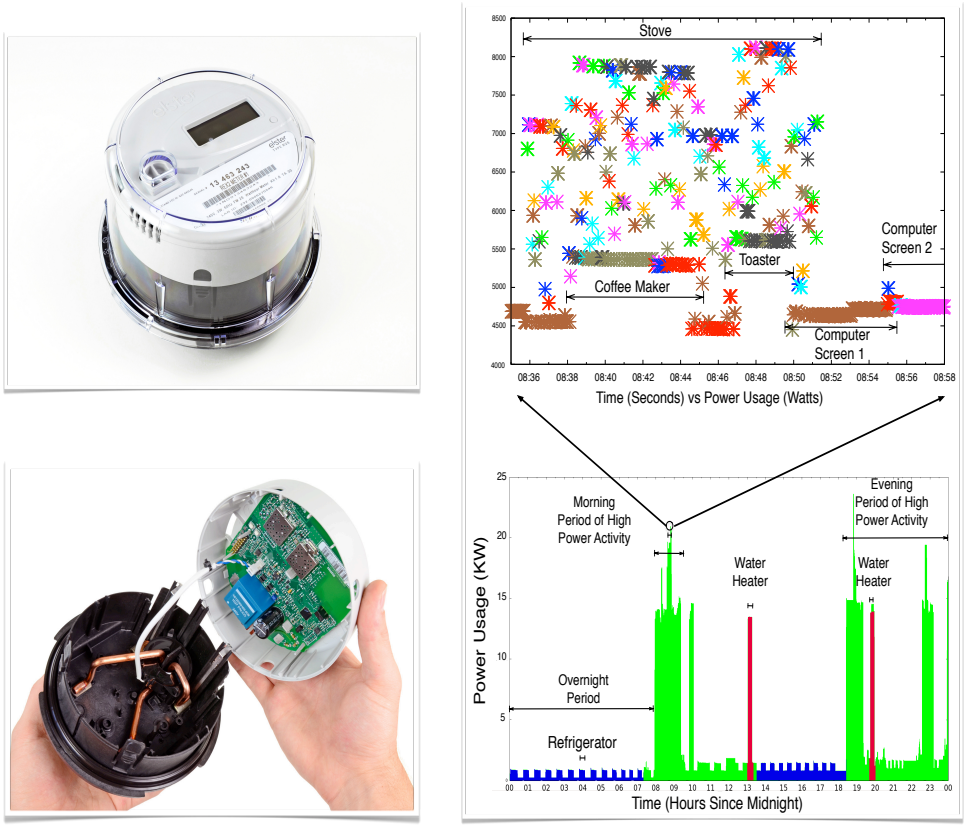

Figure 1.1. Smart meters are an example of constrained devices that gather potentially private data about individuals. They are implemented using low-cost microcontrollers that are significantly less capable than processors in personal computers. Bottom-right: Daily power trace of a home at a 30 second granularity. Top-right: Period corresponding to the same home from 8:34 am to 8:58 am at 1 second granularity. Automatically labeled power segments are represented by different colors. Appliance labels correspond to logged activities. Meter photographs [3].

and receive packets from tags or other computer systems via untrusted relays. These relays provide power and communication for these more capable tags. In order to provide secrecy and integrity to communication, BAT encrypts the payload and utilizes cryptographic message authentication codes (MACs).

The practicality of BAT is evaluated through measuring the throughput and cryptographic overhead of a prototype implementation using the UMass Moo [148], a current CRFID prototype, and software radio. The maximum throughput observed was 18 Kbps and data can be encrypted at a rate of 61 Kbps.

1.5 Privacy-Preserving Aggregation with RFID-Scale Devices

CRFIDs are an ideal platform for developing distributed applications to collect data such as the monitoring of infrastructures. In some cases it is important to allow an untrusted party to obtain an aggregate from data obtained from multiple devices such that individual entries are not revealed in the process. A solution to this problem relies on the distributed calculation of a perturbed answer to provide distributed differential privacy. However, CRFIDs are highly constrained devices that have variable power and limited RAM and computational capabilities.

Shi et al. [140] and Chan et al. [33] propose systems in which a group of individuals collectively generate a noisy aggregate to achieve privacy. The supporting cryptographic computation that each individual device needs to perform is implemented and measured on the UMass Moo. The feasibility of these approaches is evaluated based on the time it takes for a CRFID to contribute to the computation of these noisy aggregates. A Moo needs to make a 17 s computation to contribute to this aggregate. However, if fault tolerance is implemented, a Moo needs to perform a 2-minute calculation when 100 other devices are involved or a 4-minute calculation when 10,000 other devices are involved.

1.6 Organization

Chapter 2 provides general background and definitions regarding a system model for data collection using constrained devices. Chapters 3-6 develop the main contributions of this dissertation. Chapter 3 demonstrates the need for providing privacy in utility metering, and Chapter 4 evaluates the feasibility of implementing privacy-preserving meters on low-cost microcontrollers. Chapter 5 describes a networking stack for RFID-scale devices, and Chapter 6 discusses the practicality of implementing a privacy-preserving system for computing aggregates with RFID-scale devices. Chapter 7 explores the implications of implementing similar privacy-preserving systems for medical devices that collect medical telemetry with potentially sensitive information. Finally, Chapter 8 lists research problems for future work.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 2

BACKGROUND AND DEFINITIONS

This chapter provides background and definitions needed to understand the privacy-preserving model presented in this dissertation, as well as the techniques to provide privacy in this context. It begins with a description of constrained embedded systems and the data collection model. The privacy-preserving building blocks discussed are zero-knowledge proof systems, distributed differential privacy, and secrecy and integrity for networking.

2.1 Constrained Embedded Systems

Embedded systems are systems that are designed to perform only a handful of specific tasks. For example, an electric smart meter is an embedded system designed to measure the electric load on a home at fixed intervals of time, record these measurements, and display them or transmit them via a wireless channel. A smart meter is not designed to implement arbitrary functionality or functionality that changes often.

The techniques in this dissertation are relevant for embedded systems that are implemented using either low-cost microcontrollers or ultra-low-power microcontrollers. At the time of writing, low-cost microcontrollers are priced below $9.00 USD, and ultra-low-power microcontrollers consume under 1 mW/MHz. There are a wide range of microcontrollers that fit into this category with 8, 16 or 32-bit architectures and that run at a variety of clock speeds from 4-100 MHz. The Texas Instruments MSP430 (16-bit architecture, 4 MHz-25 MHz frequencies) [1], the ARM Cortex-M3 (32-bit

architecture, 20 MHz-100 MHz frequencies) [130], the ARM Cortex-M0 (32-bit architecture, 30 MHz-50 MHz frequencies) [2], and the ARM Cortex M0+ (32-bit architecture, 32 MHz-48 MHz frequencies) [7] are common platforms. Not all embedded systems fall into this category. Automotive, medical, or military applications may be implemented using high-performance microprocessors that cost up to $200 USD or include multicore processors [4].

2.2 Distributed Model for Data Collection and Analysis

This dissertation is concerned with the problem of computing a function on data that is generated by multiple individuals that do not necessarily trust each other with their data. Further, individuals may also not fully trust any third party to compute this function in a centralized fashion, as illustrated in Figure 2.1. This dissertation will primarily restrict its attention to devices that have computational, power or cost constraints. For simplicity, this work assumes that all the devices considered in a single application store their data with a compatible schema—for example, recording data about a time series.

The term *privacy-preserving* will be used in two different ways in this dissertation, depending on the application. The first will apply in the context of utility metering and the second in the context of aggregation across multiple individuals.

In utility metering, it is important that a function such as the calculation of a bill on an individual's usage reflect the exact monetary amount due, but hide the details of how that individual actually spent the billed resource. In other applications, a function is calculated using data from multiple individuals; such as calculating the number of medical devices that have experienced a malfunction in a given device population. In this case, a desired notion of privacy may be that the result of the calculation not reveal whether or not a particular device's input was included in the

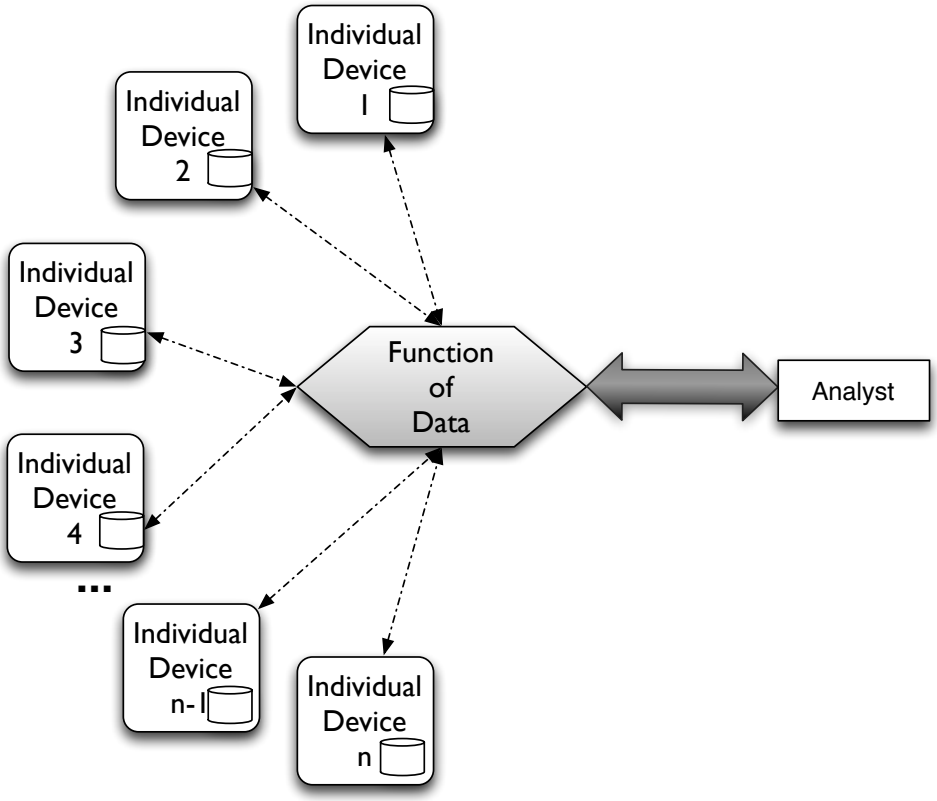

Figure 2.1. Distributed model for data collection and analysis. Individuals on the left produce data with the purpose of computing a combining function across individuals or across time. Each individual may not want to share his/her data with anyone in order to perform this calculation to preserve his/her privacy. A function may be performed on data from a single individual corresponding to a period of time or data corresponding to multiple individuals.

calculation. This concept of privacy will help in quantifying the extent to which an adversary may learn the actual input from any particular device.

The following sections discuss these two notions of privacy as well as some general techniques for achieving them. Specifically, Zero-Knowledge Proof systems (§2.3)— originally introduced by Goldwasser et al. [68]— provide a general cryptographic technique that would allow a party to disclose the output of a function to another untrusted party without disclosing additional information, such as the data inputs to the function. In the process, the party that receives the output of the function obtains proof with strong guarantees that the function was correctly computed. Differential privacy (§2.4), originally introduced by Dwork et al. [51, 43, 52] provides a clean framework for quantifying privacy. This dissertation is concerned with ways to achieve this notion of privacy in a distributed setting. Distributed techniques to achieve differential privacy were first introduced by Dwork et al. [50] and shortly thereafter developed by others, including Rastogi et al. [123], Shi et al. [140] and Chen et al. [37].

The broader term *privacy-aware* is used to describe system aspects that contribute to the goal of preventing information disclosure. For example, the networking stack BAT (Chapter 5), provides secrecy and integrity, and while these properties themselves do not *preserve privacy* per se, this secure channel plays an important role in implementing a privacy-preserving system.

2.3 Zero-Knowledge Proof Systems

Zero-Knowledge Proof (ZKP) systems [69] were originally developed as challenge-response protocols that allow a *prover* to demonstrate the knowledge of a *secret* to a *verifier*, without revealing any partial information that would help the *verifier* infer the secret, other than the fact that the *prover* knows the secret. These protocols relied on interactive verifications in which the *verifier* presents a series of challenges to the *prover* that can easily be responded to when the *prover* knows the *secret*,

but are extremely difficult to respond to reliably without knowledge of the *secret*; as the number of consecutive challenges increases, the probability of answering these challenges without knowing the secret decreases exponentially. Zero-Knowledge Proof Systems can be made non-interactive, for example by using the well known Fiat-Shamir heuristic [59].

Chapter 4 builds on ZKP systems to implement a privacy-preserving billing scheme with time-of-use based tariffs for smart electricity metering making use of commitments and Camenisch-Lysyanskaya signatures. In that setting, a customer fitted with a smart meter proves to a utility provider the amount to be paid for their electricity consumption within a specific time period, without revealing any details about their fine-grained consumption. The bill is calculated on the basis of detailed readings, every half hour or fifteen minutes, that are each billed according to the dynamic price of electricity at that time, or a pre-defined but time variable tariff scheme. These protocols are applicable when consumers do not trust the utility with their detailed electricity usage information, and the utility does not rely on consumers to honestly report their usage.

2.3.1 Cryptographic Commitments

Commitment schemes are cryptographic primitives that enable a party to create the digital equivalent of an envelope for a secret. Commitments support two important properties: *hiding* protects the secrecy of the committed message, and *binding* ensures it can only be opened to the committed message.

Pedersen commitments [120] are information-theoretically hiding and binding under the discrete logarithm assumption. They rely on a set of global parameters, namely a group G of prime order p with generators g and h. Under that scheme a commitment C to message $r \in \mathbb{Z}_p$ is computed as $C = g^r h^o$ where o is an *opening* nonce chosen uniformly at random in $\mathbb{Z}_p$. Opening a commitment C involves disclos-

ing the values r and o to a verifier. In addition to opening the commitment, efficient protocols exist for a prover to convince a verifier that they know the committed value without disclosing it.

Fujisaki-Okamoto commitments [61] are similar to Pedersen commitments, except that they make use of a group of composite, hidden order instead of a group of prime order. They allow the committed value to be any integer, including negative integers. Pedersen or Fujisaki-Okamoto commitments can be used depending on whether a device needs to encode negative values or not.

2.3.2 Camenisch-Lysyanskaya Signatures

Digital signatures allow a party to show the authenticity or integrity of a message or document. Different signature schemes may be used to achieve different security properties. A standard signature scheme, such as DSA, can be used to ensure the integrity of any further statement proved on the basis of previous measurements (e.g. meter readings). When a device is not trusted, a signature scheme such as Camenisch-Lysyanskaya (CL) signatures [28] can be used to sign messages or documents individually. For example in the case of time-of-use billing, a meter periodically commits to meter readings. Those commitments are signed and the customer can use the signature to prove functions of the bill to a verifier.

CL-signatures allow a requesting party to obtain a digital signature on a commitment from an authorized signer. In particular, Camenisch and Lysyanskaya [28] provide efficient protocols for computing a signature on a commitment message, as well as for constructing zero-knowledge proofs of knowledge of a signature on a committed or encrypted message. Note that there are two digital signature schemes attributed to Camenisch and Lysyanskaya; their earlier scheme [27] relies on the Strong RSA assumption, while the later scheme relies on a discrete-logarithm-based assumption

(the LRSW assumption) [99]. CL-signatures [28] can be implemented using elliptic curve groups, as long as there is an efficient bilinear map that is non-degenerate.

We describe the key generation function, the signing function and the signature verification function for CL-signatures using the notation in [129]:

1. $CLKeyGen(1^k)$. Given a security parameter k, and the number of block messages to sign n, the signer generates the first part of their public key: $(p, \mathbb{G}, \mathbb{H}, g, h, e)$, such that there is a mapping $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{H}$, which is bilinear, non-degenerate and efficient to compute. The signer then chooses the following parameters for their private key: $x, y, z_1, \ldots, z_n \in_R \mathbb{Z}_p$. Next, the signer uses these parameters to compute $X = g^x, Y = g^y$ and $Z_i = g^{z_i}$ for all $i \in [1, n]$. The public key is $pubkey = (p, \mathbb{G}, \mathbb{H}, g, h, e, X, Y, \{Z_i\}, \{W_i\})$, and the secret key is the public key concatenated with $(x, y, \{z_i\})$.

2. $CLSign((x, y, \{z_i\}), \{m_i\})$. To sign n blocks $\{m_i\}$, the signer first chooses $a \in_R \mathbb{G}$, and computes $b = a^y$. The signer then computes $A_i = a^{z_i}$ and $B_i = (A_i)^y$ for all $i \in [2, n]$. Finally, the signer computes $\sigma = a^{x+xym_1} \prod_{i=2}^{n} A_i^{xym_i}$. The signature is $sig = (a, \{A_i\}, b, \{B_i\}, \sigma)$.

3. $CLVerifySign(pubkey, \{m_i\}, sig)$. The verifier performs the following computations and outputs *accept* if the following equalities hold: $e(a, Y) = e(g, b)$; $e(a, Z_i) = e(g, A_i), \forall i \in [1, n]$; $e(A_i, Y) = e(g, B_i), \forall i \in [1, n]$; and $e(g, \sigma) = e(X, a) \cdot e(X, b)^{m_1} \cdot \prod_{i=2}^{n} e(X, B_i)m_i$.

2.4 Distributed Differential Privacy

Rastogi et al. [124] classify the mechanisms to provide privacy to databases depending on where they are implemented in the lifecycle of the data, from the point the data is collected to the point some data is presented to an analyst. This classification is illustrated in Figure 2.2. The data can be subject to *local perturbation* where

individuals trust no one but themselves; alternatively in *data publishing*, data can be aggregated by a trusted entity which then transforms the dataset into a different dataset in a way that it preserves some characteristics of the original dataset while providing privacy to individuals. Finally, data may be aggregated by a trusted entity and analysts are only allowed to perform queries. A query processor will compute the correct answer to the query and will add noise to it according to the *sensitivity* of the query.

This last mechanism, known as *output perturbation*, allows for the implementation of querying systems that provide *differential privacy*. This notion of privacy, originally introduced by Dwork et al. [51, 43, 52] provides a clean definition that allows for the quantification of loss of privacy in statistical databases in a way that is independent of the additional information that an adversary may possess.

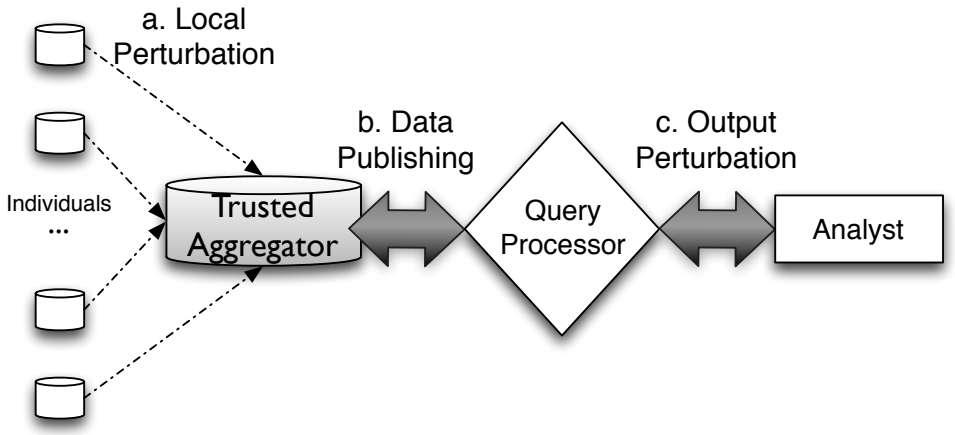

Figure 2.2. Techniques to provide privacy, implemented at different times in the lifecycle of the data, from the point the data is collected to the point some data is presented to an analyst

2.4.1 Differential Privacy with a Trusted Aggregator

Differential privacy is a concept originally developed in the context of statistical databases, which can be thought of as a way to prevent a query—from an analyst that does not have direct access to the database— from revealing whether or not

a particular record was used in the computation of the answer to that query. One way of providing answers to queries to satisfy this notion of privacy is via an output perturbation mechanism as illustrated in Figure 2.2.

A formal definition of this notion is below, as well as a description of a general technique for achieving it. Related work provides ways to eliminate the requirement of a trusted aggregator (§2.4.2).

Formally, a computation $\mathcal{F}$ on a set of datasets $\mathcal{D}$ provides (ε, δ)-differential privacy if for each pair of datasets $D_1, D_2 \in \mathcal{D}$ that differ by at most one record and for all the outputs in $S \subset \text{Range}(\mathcal{F})$, the following inequality is satisfied:

$$Pr[\mathcal{F}(D_1) \in S] \leq \exp(\varepsilon) \cdot Pr[\mathcal{F}(D_2) \in S] + \delta$$

Intuitively, this means that the output of the computation is independent of the inclusion of a particular record, for most records. The parameter ε provides a way to trade privacy with accuracy, and the parameter δ provides a way to relax the condition for which achieving $(\varepsilon, 0)$ is difficult.

A general mechanism for providing differential privacy to certain kinds of queries on a dataset is to allow a trusted aggregator to first compute an exact answer to a query and then add noise drawn from a Laplace or Geometric distribution according to the sensitivity of the query. As some research has pointed out, it is possible to combine this approach with a cryptographic solution in order to eliminate the need for a trusted aggregator. Logically, this is illustrated by Figure 2.3.

2.4.2 Differential Privacy with an Untrusted Aggregator

Several authors have provided cryptographic solutions to eliminate the need for a trusted aggregator in the process of adding noise after an exact answer to a query has been calculated [50, 123, 140, 37]. The general idea behind these approaches is to allow participants to collectively generate the noise to be added to the answer

while providing partial sub aggregates for that query that when combined output the answer to the query.

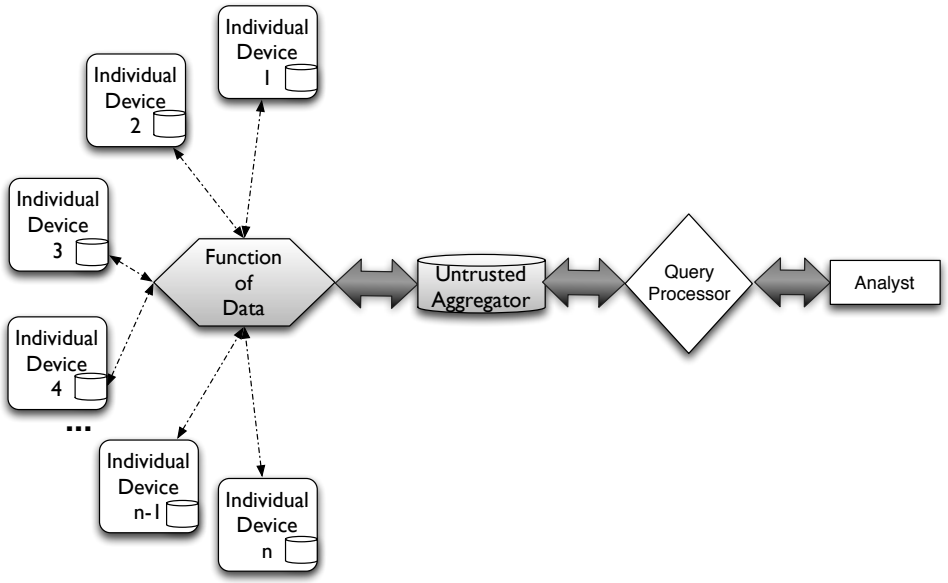

Figure 2.3. Slight modification to the general collaboration model in which a device or entity may be selected to aggregate without being trusted.

For example, it is known that with the use of homomorphic encryption, a set of parties can compute an exact counting query in such a way that each participant provides an encrypted subtotal to the counting query; adding the encrypted subtotals will give the ciphertext corresponding to the total counting query. Thus, decrypting this value would allow for the computation of the total without revealing the individual contributions. Of course, this does not provide differential privacy. However, if instead of asking that each individual return an exact subtotal, each returned a subtotal plus a small amount of noise, such that the total noise added is drawn from a Geometric distribution, then the final answer will be the actual answer to the query plus some noise, just as when a trusted aggregator is used. In this case however, the individual parts are not revealed until all the contributions are combined.

Actual solutions proposed in the literature are more complex in order to provide fault tolerance [33] or improve scalability and practicality [37]. Another system that

uses a distributed approach of a sort is GUPT [104], however, the main purpose of this system is to maximize the utility of a computation.

2.5 Secrecy and Integrity for Networking

A communication channel that provides secrecy and integrity is an important component for creating a distributed system for privacy-aware collaboration. Throughout this dissertation, it will be required that data from each individual device travels encrypted from one place to another to ensure that eavesdroppers are not able to infer the values that each reports during the computation of an aggregate. Also, some applications may require an additional mechanism to ensure that data is not tampered with during travel and that messages come from a verifiable location. This last feature does not necessarily compromise privacy because one may be able to verify that a given message came from a particular individual, without seeing what it contains, e.g. because the payload of the message is encrypted.

We should note that security analyses typically concern confidentiality, integrity, and availability [20]. However, this dissertation is not primarily focused on ensuring availability because the devices in question—particularly the RFID-scale devices—are such that it is relatively simple to jam a channel so that a device can no longer communicate. Thus, that problem is beyond the scope of this dissertation.

2.5.1 Symmetric Key Primitives

When a pair of individuals share a symmetric key, they can typically communicate more efficiently because the functions to encrypt and decrypt are faster than their public key counterparts. In fact, the implementations described in Chapters 4 and 5 both use the AES symmetric block cipher to encrypt and decrypt data because software implementations are fast, and some low-cost and ultra-low-power microcontrollers implement this cipher in hardware. However, as will be more explicit

in each of the applications described in this dissertation, the successful utilization of this block cipher requires that two parties share a key. In that case, keys can be generated by utilizing key agreements such as the Diffie-Hellman key exchange or an identity-based encryption key agreement scheme, for example. The latter offers some attractive benefits, including the possibility of generating shared keys in a non-interactive fashion (§2.5.3).

2.5.2 Public Key Primitives

Identity-Based Encryption provides a useful technique that simplifies the task of computing pairwise keys to allow devices to establish encrypted pairwise communication links with other devices. Devices may compute these pairwise keys on demand without needing to know potential recipients beforehand. In principle, a device may not know how many other devices are or will be in a network—or even which devices it will need to communicate with in the future.

An approach with pre-shared encryption keys has several limitations, such as key revocation, key expiration and an inability to specify recipients. Therefore a public key encryption scheme is highly desirable in this setting. Identity-based key exchange offers several advantages over more traditional key exchange methods, like the Diffie-Hellman key exchange. For instance, the Diffie-Hellman key exchange is particularly vulnerable to man-in-the-middle attacks, unless a third party authenticates protocol participants. This limitation is usually addressed by the addition of a Certificate Authority (CA), but key-management is difficult in CA-based systems [58].

2.5.3 Identity-Based Encryption

Identity-based key agreement schemes allow for the creation of private key/public key pairs, such that the public key is any string and the corresponding private key can only be granted by a trusted entity—a private-key generator (PKG). Thus, for example, it would be easy for a device to encrypt a message so that only another

device with serial number x could obtain the corresponding decrypting private key from the PKG. Additionally, the sender device may be able to append an expiration to the public key.

Sakai, Ohgishi and Kasahara [132] proposed a non-interactive identity-based key exchange protocol similar in spirit to the Diffie-Hellman key exchange protocol. However, Sakai, Ohgishi and Kasahara use a pairing on an elliptic curve. A symmetric pairing is a non-degenerate bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ that takes two elements from a cyclic elliptic curve group $\mathbb{G}$ and returns an element in another group $\mathbb{G}_T$. The bilinear property implies that

$$e(P_1 + P_2, Q) = e(P_1, Q) \cdot e(P_2, Q)$$

and

$$e(P, Q_1 + Q_2) = e(P, Q_1) \cdot e(P, Q_2),$$

for all elements of $\mathbb{G}$. Also, $e(aP, Q) = e(P, aQ) = e(P, Q)^a$.

Thus, given the symmetric pairing e as above, where $\mathbb{G} = \langle P \rangle$, and two hash functions $H_1 : \{0,1\}^* \to \mathbb{G}$; $H_2 : \mathbb{G}_T \to \{0,1\}^n$, the *setup*, *key extraction* and *key agreement* functions for the Sakai, Ohgishi and Kasahara key exchange can be described as follows:

Setup: A private-key generator (PKG) chooses a secret master key s uniformly at random from $\mathbb{Z}_p$; and sets $R = sP$ as the public key together with the public parameters, including H_1 and H_2.

Key extraction: The PKG will compute the private key of a device A as $d_A = sQ_A$, where Q_A is a hash of A's id $Q_A = H_1(id_A)$.

Key agreement: Two devices A and B would create a shared key if they know each other's identity strings id_A and id_B. That is, A would compute $K_A =$

$H_2(e(d_A, Q_B))$ and B would compute $K_B = H_2(e(d_B, Q_A))$. Note that $K_A = K_B$ because

$$H_2(e(d_A, Q_B)) = H_2(e(sQ_A, Q_B))$$

and

$$H_2(e(sQ_A, Q_B)) = H_2(e(Q_A, Q_B)^s);$$

similarly,

$$H_2(e(d_B, Q_A)) = H_2(e(Q_B, Q_A)^s).$$

Therefore the equality $K_A = K_B$ follows directly from the symmetry of e.

This identity-based key exchange offers a simple way of generating shared keys in a non-interactive manner without the need of a certification authority. There are more efficient pairing based key agreements—some of them trading efficiency at the cost of a small interaction, such as the Smart-Chen-Kudla key agreement [36]. Also, it is important to note that the SOK key exchange uses a so-called Type-1 pairing [62]. In general however, it is possible to implement pairing-based protocols more efficiently using a more general definition of a pairing. That is, given a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, where $\mathbb{G}_1$ and $\mathbb{G}_2$ are cyclic elliptic curve groups and $\mathbb{G}_T$ is a cyclic subgroup of the multiplicative group of a finite field of the same order as $\mathbb{G}_1$ and $\mathbb{G}_2$, a Type-1 pairing is a pairing where $\mathbb{G}_1 = \mathbb{G}_2$. While Type-1 pairings offer the simplest way to describe a pairing-based cryptosystem, they are often not the most efficient to implement. A Type-3 pairing is a pairing where there are no efficiently computable homomorphisms between $\mathbb{G}_1$ and $\mathbb{G}_2$. The most efficient known pairings to compute are Type-3. While Type-1 pairings are simpler to design cryptosystems, Chaterjee and Menezes [34] provide a natural transformation of a Type-1 protocol to a Type-3 protocol. Chapter 5 discusses further details and performance implications of using different protocols and primitives in the context of RFID-scale networking.

CHAPTER 3

PRIVACY ISSUES OF SMART METERING

This chapter shows that even without detailed knowledge of appliance signatures *a priori* or prior training, it is possible to extract complex usage patterns from smart meter data using off-the-shelf statistical methods. The methods outlined in this chapter are able to label specific types of activity in the home over time based on a number of characteristics, including the level of power consumption, its intermittency, and its duration [108].[1]

Issues of privacy involving smart meter data are becoming increasingly important due to the widespread deployment of smart meters, which collect and send data to a centralized location. This model has serious privacy implications since the aggregator inadvertently gains detailed information about household activities. The current practice for achieving privacy is simply trusting this aggregator—usually the utility provider—to protect information from others and respect the privacy of individuals. In a Facebook-world where users willingly share invasive details of their private lives with friends and strangers, the ability to extract this information may not appear to be an egregious violation of privacy. However, with a relatively small amount of data, it is possible to infer detailed information about household activity—questions such as how many people are in a home at a given time and whether a resident went out for dinner on a particular evening, for example.

---

[1]This section draws from previously published work: "Private Memoirs of a Smart Meter" by A. Molina-Markham, P. Shenoy, K. Fu, E. Cecchet, and D. Irwin. In 2nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings. November 2010.

Research on nonintrusive load monitoring (NILM) has shown that it is possible to disambiguate individual appliance usage from an aggregate smart meter power trace by using prior knowledge of an appliance's power signature [97]. Such techniques reduce or eliminate the need for outlet- or appliance-level meters, since they are able to extract detailed usage information for individual appliances from an aggregate household power trace. The approach in this chapter shows that prior knowledge is not required in order to correlate power traces with user activity. Entities that gather large amounts of data would potentially be able to predict even more detailed facts, such as residents' genders and ages. Such information is a foundation for building powerful analytic tools for predicting behavior that could potentially be misused by companies or even criminals.

The ability to correlate power segments to human activity increases with the granularity of the measurements. Therefore this work provides a basis to support the idea that the data that is reported directly to analysts should have a coarse granularity. Even if dynamic pricing requires high granularity measurements for calculating a bill, only information at coarse granularities should be shared with utilities.

Contribution

The work in this chapter is the first to demonstrate that it is possible to identify *power segments* without prior knowledge using off-the-shelf statistical techniques. An analysis of power traces at fine granularity demonstrates potential privacy leakage by smart meters. Sending power traces with resolution of a few seconds to utility companies would enable them to answer specific questions about individuals' activities. While other research on nonintrusive load monitoring has depended on prior knowledge of a business, residence, or set of appliances, this chapter shows that as the granularity of measurements of a smart meter increases, the need for prior knowledge to infer information decreases.

3.1 Smart Metering

Recently, there has been an increasing focus on "greening the home" using a combination of fine-grained power consumption monitoring, smart appliances, and renewable energy sources, e.g., rooftop solar panels. The trends have led to the design of smart electric grids that provide support for various technologies, including net metering, demand response, distributed generation, and microgrids [94]. An important component of a future smart grid is the installation of smart (or net) meters in homes that support both dynamic pricing and a two-way flow of electricity between homes (or microgrids) and the larger grid. As these meters become more sophisticated, they are able to measure household power consumption at ever finer time-scales. Initial deployments of the Advanced Metering Infrastructure (AMI) in Ontario, Canada support meter readings at 5 to 60 minute intervals [31]. The next generation of smart meters will reduce these time intervals to one minute or less. For instance, in July 2010, PECO, one of the largest providers of electricity and gas in the U.S., selected *Sensus* to provide an AMI with meters that support one minute intervals [139] such as the one illustrated in Figure 3.1.

Figure 3.1. Smart meter installed in the state of Pennsylvania in 2012.

3.2 Dynamic Pricing and Optimizing Distribution

One of the motivations for smart meters is the possibility of implementing dynamic pricing schemes that provide incentives for consumers to use electricity at times when the demand is lower. This would result in a significant reduction in infrastructure over-provisioning, which has been the main approach for dealing with peak demands. Additionally, utility companies envision that this consumer feedback would allow for better handling of temporary surpluses and enable collaboration with users to stop or minimize consumption when the grid is overloaded.

Another goal of implementing smart meters is the ability to collect information that would allow utilities to perform mid- and long-term planning. For example, utility companies would benefit from knowing the consumption trends of populations, such as whether more electric cars are being charged or neighborhoods are switching to more energy-efficient appliances.

3.3 Implications of Privacy Leakage through Smart Metering

Recent work by Quinn [122] provides an overview of the privacy implications of fine-grained power consumption monitoring. While Quinn does not present specific techniques or conduct a detailed data analysis, he posits that those with access to smart meter data will be able to infer answers to many questions about a household's personal, and potentially private, activity. While the answers to some of these questions may seem innocuous, e.g., when do people watch TV, others are quite disturbing, e.g., is there a newborn in the house. Table 3.1 highlights a few of these private questions, along with the power consumption pattern that may reveal their answer. The table also lists the monitoring granularity we believe a smart meter requires to accurately identify the necessary pattern. For instance, a relatively low level of power consumption and variation may indicate that no one is home, while power activity every few hours throughout every night may indicate regular nighttime

Question	Pattern	Granularity
Were you home during your sick leave?	Yes: Power activities during the day No: Low power usage during the day	Hour/Minute
Did you get a good night's sleep?	Yes: No power events overnight for at least 6 hours No: Random power events overnight	Hour/Minute
Did you watch the game last night?	Yes: Appliance activity matching TV program No: No power event in accordance with game showtime	Minute/Second
Did you leave late for work?	Yes: Last power event time later than Google maps estimated travel time No: Last power event time leaves enough time for commute	Minute
Did you leave your child home alone?	Yes: Single person activity pattern No: Simultaneous power events in distinct areas of the house	Minute/Second
Do you eat hot or cold breakfast?	Hot: Burst of power events in the morning (microwave/coffee machine/toaster) Cold: No power event matching hot breakfast appliances	Second

Table 3.1. Private questions and answers that fine-grained power consumption data reveals.

feedings for a newborn. Even answers to seemingly innocuous questions may prove valuable to third-parties, e.g. for adjusting insurance rates, targeting advertising campaigns, resolving legal disputes, or conducting criminal investigations.

3.4 Methodology to Estimate Privacy Leakage

Revealing complex usage patterns *is not difficult* with an approach that *opaquely labels different types of household activity.* This approach leverages simple off-the-shelf clustering and pattern recognition techniques on 2 months of power consumption data from 3 homes. To gather the data, each home's main circuit breaker is instrumented

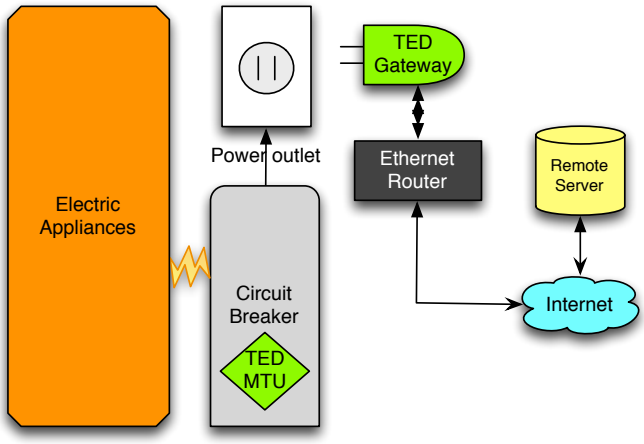

Figure 3.2. Architecture using a TED monitor as a smart meter.

with a TED energy monitor [8] that logs household power consumption every second. Figure 3.2 graphically depicts the architecture. The TED monitor uses the home's power circuits to transmit power readings to a TED gateway that makes them available via a built-in web browser. An embedded SheevaPlug computer in each home downloads second-level data each hour from the TED gateway and transmits it to a central repository for analysis. Each entry in the TED data log consists of a power tuple (t, p) that includes a timestamp t and the average power consumption p in kilowatts over the previous second. The one-second logging granularity is smaller than that of existing smart meters [9], which allows for the identification of many patterns that are not possible with current meters.

This analysis consists of four steps: 1) pre-process power traces using an off-the-shelf clustering algorithm to identify and label similar types of power events, 2) tag each power event with one or more defining characteristics, 3) filter out automated appliances by observing their signatures during periods of low power activity, and 4) map opaque labels to real-life events using a small amount of externally gathered knowledge.

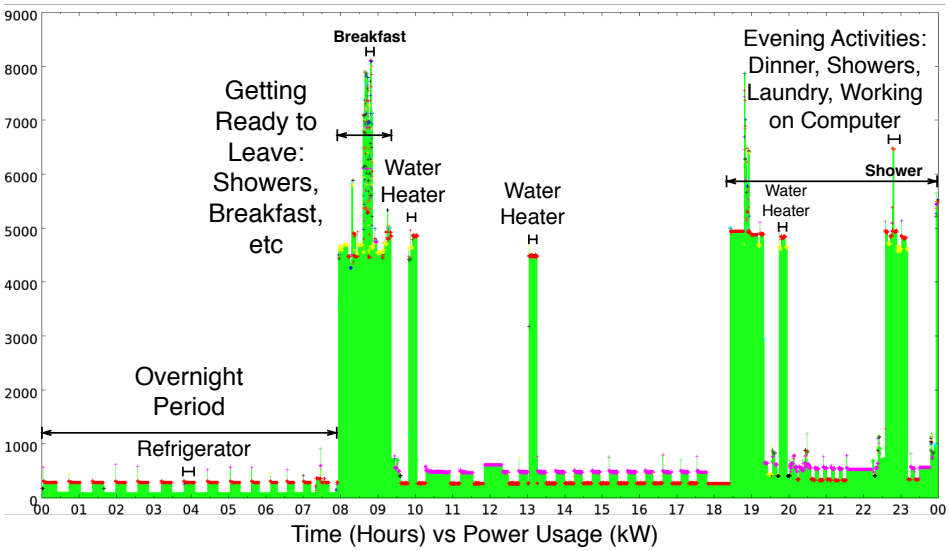

Figure 3.3. Example day-long second-level power trace with labels from the day's activity log.

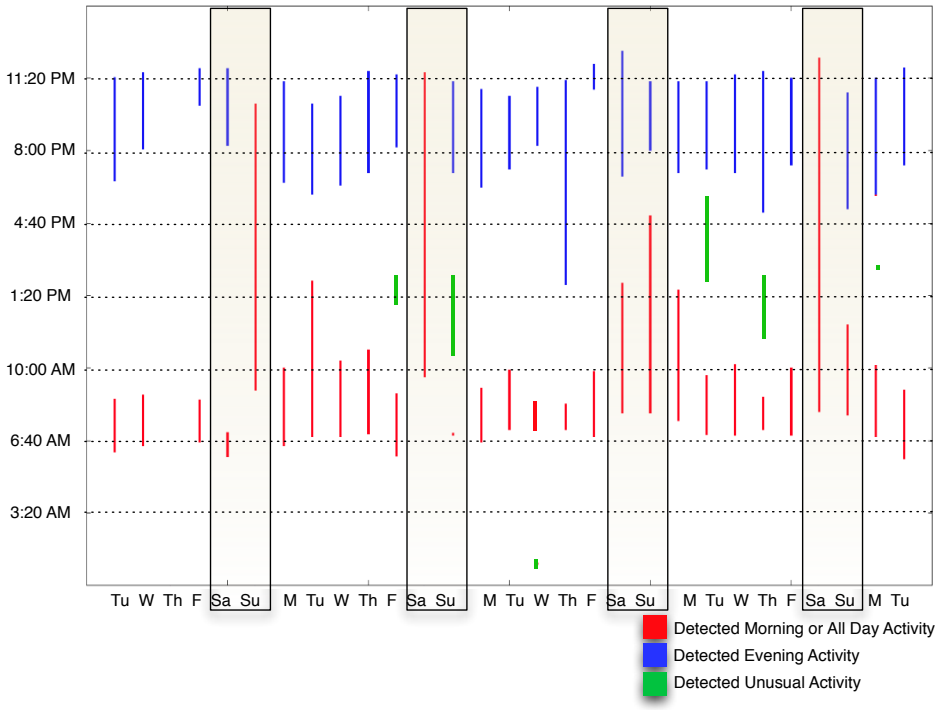

Figure 3.4. Identification of human presence with high probability for each day of the month.

3.5 Results

Label Power Events. First each power trace is pre-processed using a density-based clustering algorithm (DBSCAN [56] as implemented by WEKA [74]) to group together power tuples into *power segments*. A *power segment* is simply a collection of tuples with a particular pattern of power consumption values that are adjacent in time. Power segments often have a constant power consumption over a given time period, although this is not required. In some cases, events of the same shape are identified, such as a steep ramp-up and then leveling off. The algorithm labels the power segments such that segments with a similar pattern receive the same label. In many figures, these labels are distinguished using different colors. DBSCAN was chosen because of its simplicity. There are other potentially more sophisticated, algorithms, such as CLIQUE [12], MAFIA [65], DENCLUE [76]. However, even this simple approach is able to detect household activities with high accuracy.

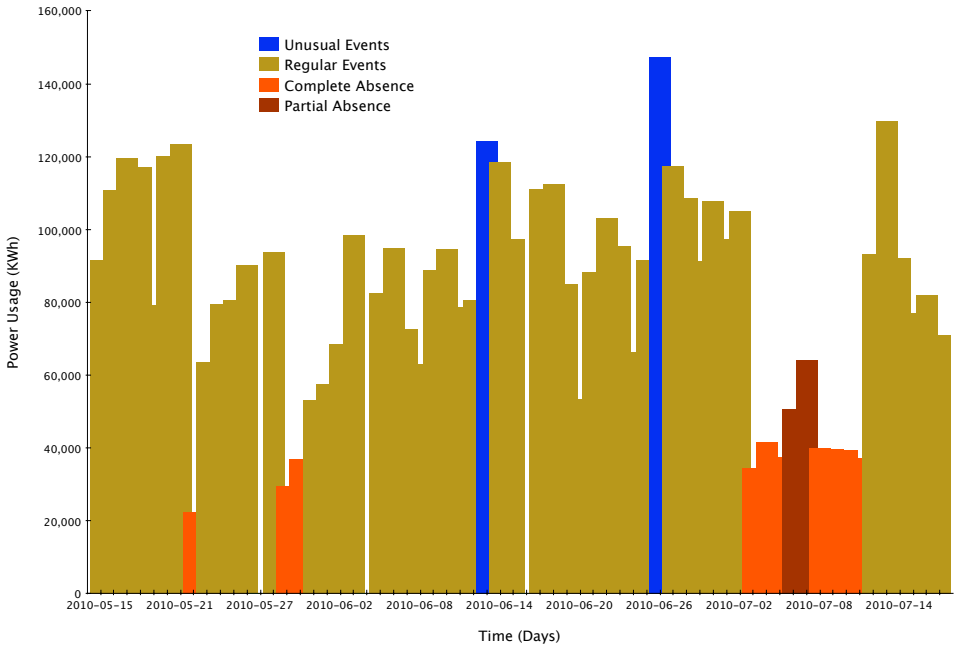

Figure 3.5. Low power periods correspond to little human activity over a two-month trace for one home.

Tag Power Events. Each power segment is appended with a few distinguishing attributes. The primary attributes are each power segment's duration and its power step, i.e, the power increase or decrease at the beginning of the segment. Power segments are also labeled with a particular shape if the power level was not constant. In this case, non-constant shapes are identified and labeled manually, although it is possible to automate the process. The result is a 6-tuple that includes the segment's label, start time, average power, duration, beginning power step, and shape label. These 6-tuples can be automatically processed to answer different types of queries on the data. For example, repetitive usage patterns are identified by filtering for power segments with the same duration, beginning power step, and shape. Figure 3.3 shows power segments (appended with labels from our activity logs) in a typical day for one of the homes. In this figure, a high variation in color corresponds to human activity, e.g., periods between 8:00 AM - 9:30 AM and 6:30 PM - midnight. Using the intuitive observation that relatively high power consumption and variation indicates human activity, Figure 3.4 reveals when people were in one of the homes over the course of a month with weekends highlighted.

Filter Automated Appliances. Figure 3.3 also demonstrates that while nearly all human-triggered power events correspond to the beginning of a power segment, there are many segments that do not correspond to any human interaction. To obtain only power segments associated with human activity, the power signatures of automated appliances, such as refrigerators, heating or air conditioning, are filtered out. The intuitive observation that periods of low power activity correlate well with periods of little human activity is leveraged to isolate signatures. Figure 3.5 illustrates this point, identifying periods of low activity in a home over the 60-day trace. Likewise, periods of high activity correlate with more people being inside the home, i.e., for a get-together or party. Figure 3.6 shows power signatures for appliances during an absence from the home. In this case, the signatures correspond to a dehumidifier that

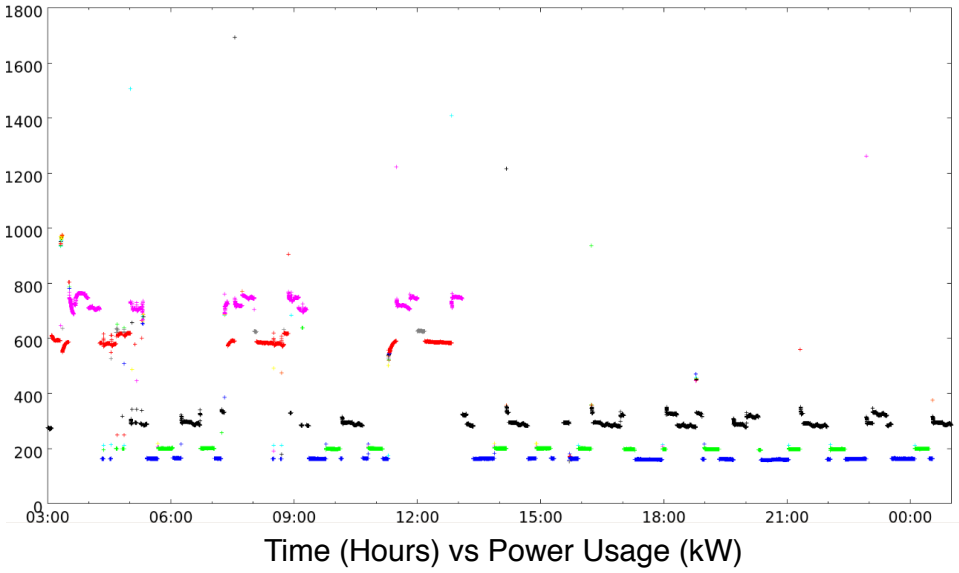

Figure 3.6. Power signatures for a dehumidifier and an air re-circulator. Note that the dehumidifier shuts off after it fills up.

runs for 2 hours every 4 hours, and an air re-circulator that runs for 20 minutes every hour.

Map Events to Real Life. After collecting and analyzing a sufficient amount of data, it is possible to identify patterns of recurring clusters according to their characteristics. Powerful data mining techniques could be applied to the obtained power segments. For example, the grouped power segments shown in Figure 3.6 could be filtered out automatically by entering them in a clustering algorithm, this time in *supervised* mode. Alternatively, tagged power segments could also be classified and matched to future occurrences. Further, pattern matching can be improved and past instances can be re-analyzed when new appliances are disambiguated. To illustrate this, Figure 3.7 shows in detail the disambiguation of power segments (identified by different colors). In this case, clustering distinguishes opaque *events* but not specific appliances or activities. An entity that had access to large amounts of data could then classify these events based on prior knowledge. In this case, knowledge from activity journals is utilized. Each home manually kept detailed power activity

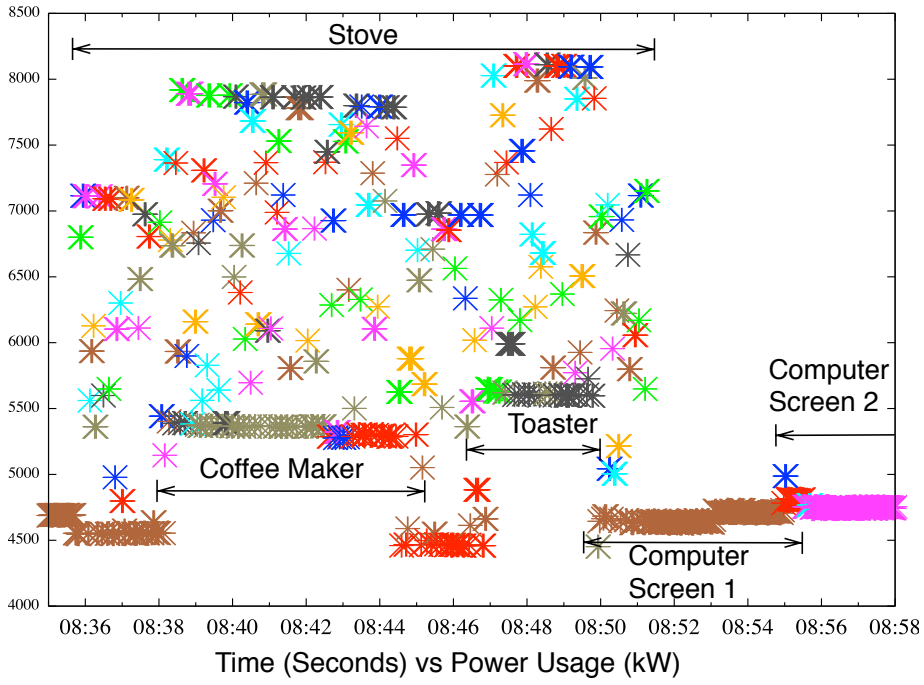

Figure 3.7. Power segments from eating breakfast. The clustering algorithm automatically generates the color scheme. The labels are from the activity logs.

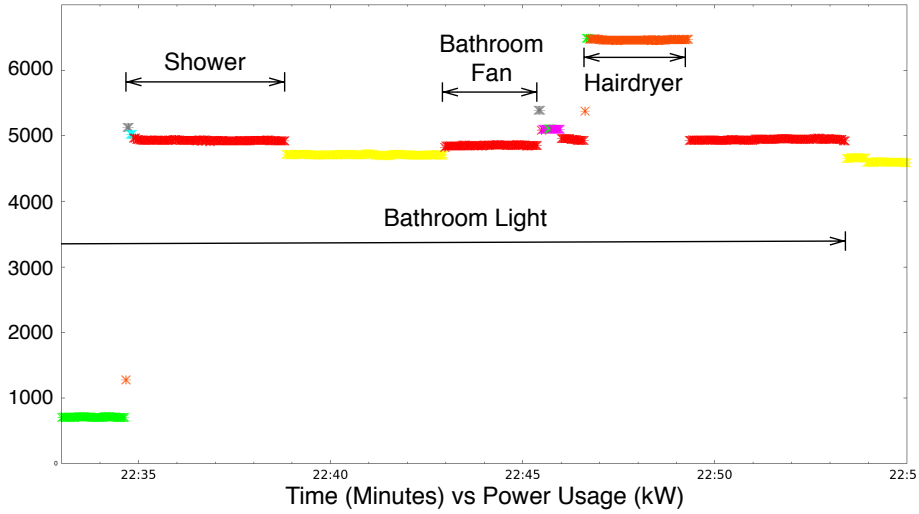

Figure 3.8. Example of the power segments from taking a shower, including labels from our activity logs.

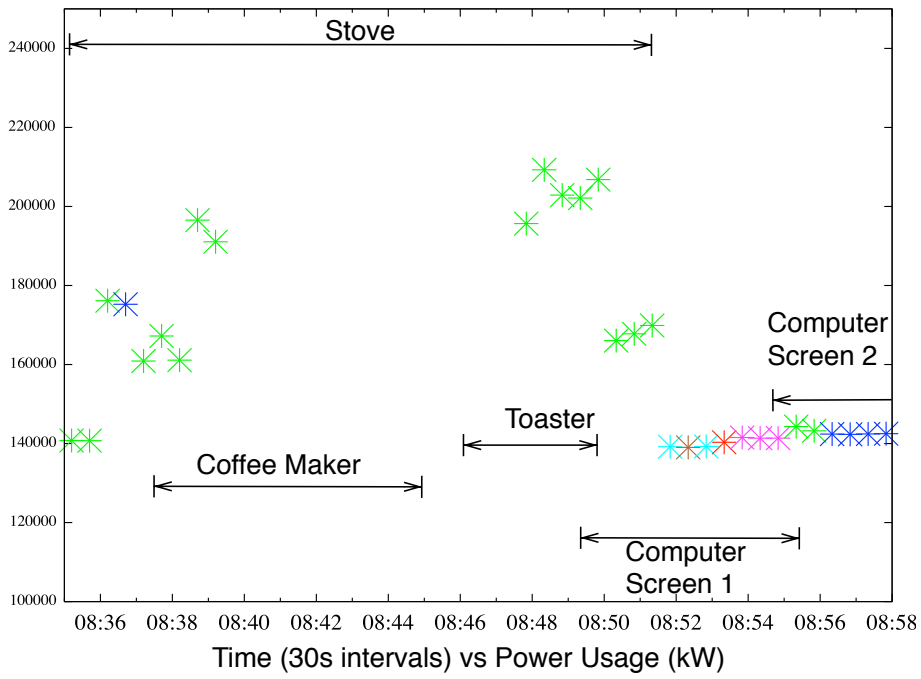

Figure 3.9. An example of the same power segments from Figure 3.7, but at a 30 second logging granularity.

journals for at least 3 days over the 60 day period to provide some knowledge of activities in the home. These journals were as accurate as possible, and recorded rough timestamps for turning on and off every light switch and appliance throughout the day. Using the data from these activity journals, the opaque power segments are mapped to specific types of real-life events. The segments in Figure 3.7 that were identified by the clustering algorithm have been marked with arrows corresponding to activities logged by the individuals living in the home. The clustering algorithm finds power segments for the stove, coffee maker, toaster and two computer screens, which is enough to answer the question in Table 3.1 about whether a person had a hot or cold breakfast that morning. Note that the algorithm is able to delimit these segments despite the simultaneous operation of other appliances. To demonstrate the importance of the logging granularity, Figure 3.9 shows the same trace as Figure 3.7,

but with a 30-second logging granularity. In this case, the pattern reveals little about the usage of each separate component.

3.6 Conclusion

This chapter highlights the issue of privacy in smart metering. A simple approach to label usage patterns using off-the-shelf statistical techniques illustrates the potentially private information that can be learned from power traces, such as when occupants are home, the number of occupants in a household, and their eating and sleeping patterns. Questions such as: Did you leave your child home alone? or Did you get a good nights sleep? can be answered by analyzing these traces. As the granularity of the measurements increases, the capability of obtaining more information will grow, especially when a centralized entity has access to data from thousands or millions of households.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 4

PRIVACY-PRESERVING SMART METERING WITH LOW-COST MICROCONTROLLERS

This chapter discusses a solution for implementing a privacy-preserving smart meter architecture that enables an electric utility to achieve its net metering goals, while respecting the privacy of its consumers. The approach leverages the notion of Zero-Knowledge proofs and provides cryptographic guarantees for the integrity, authenticity, and correctness of payments, while allowing variable pricing without revealing the power measurements gathered during a billing period.

A key impediment to the widespread adoption of privacy-preserving billing protocols is the computational and memory constraints of smart meters, which, due to cost, size, and power considerations, typically use embedded microcontrollers. Prior work does not measure these resource constraints, and, thus, implicitly assumes that meters are capable of executing protocols in a reasonable amount of time. This chapter[1] explores the economic feasibility of implementing the cryptographic techniques required for privacy-preserving smart metering and proposes a general methodology for evaluating the cost of a solution [107]. This analysis takes into account current smart meter deployments and looks at the hardware technologies utilities are adopting over both the short- and long-term. The focus is on implementing cryptographic techniques on smart meters such as those proposed by Rial et al. [126], Molina-Markham

[1]This chapter draws from previously published work: "Designing Privacy-preserving Smart Meters with Low-cost Microcontrollers" by A. Molina-Markham, G. Danezis, K. Fu, P. Shenoy, and D. Irwin. In Proceedings of the 16th International Conference on Financial Cryptography and Data Security. February 2012.

et al. [108], Kursawe et al. [92] and Jawurek et al. [82]. However, this methodology also applies to estimating the cost of similar metering systems that require privacy, including natural gas, water, and toll roads, such as the one proposed by Balasch et al. [14].

Contributions

The contributions of this chapter are:

System. This chapter describes the results of designing and implementing a privacy-friendly smart meter using low-cost microcontrollers from both the Texas Instruments MSP430 and the ARM families. It presents the first experimental results that actually measure the performance of a Camenisch-Lysyanskaya [28] (CL) based scheme using elliptic curves in constrained environments. Previous work [126] discusses and estimates, but does not include implementation results. The most comparable realization of a CL based scheme uses a Java Card [18] and does not include an elliptic curve version.

Cost Evaluation. This chapter outlines a cost evaluation strategy for implementing privacy-preserving smart meters that accounts for the special characteristics of low-cost microcontrollers and industry trends. In particular, it lists a set of system variables that designers may modify to balance security, privacy, and cost. This is the first discussion of the issues surrounding ultra-low-power implementations, which in some applications may make the difference between a meter that requires a battery replacement every few years versus every few days.

Feasibility Analysis. This chapter presents evidence to support the hypothesis that ZKP billing protocols are feasible on current deployments of smart meters and cost effective on deployments over both the short- and long-term. Because some smart meters can be remotely updated, it is plausible that a deployment may be implemented in one of these updates. In the long-term, these experimental results

may help system designers to assess the performance and cost benefits of utilizing elliptic curve primitives. This analysis takes into account the evolution of the storage and computational capabilities of low-cost microcontrollers and contrasts it to the evolution of personal computer processors.

4.1 A Zero-Knowledge Proof System for Billing

The study in this chapter focuses on the efficient implementation of the meter cryptographic components for the Rial and Danezis [126] privacy preserving smart metering protocols. Proposals by [82] can be adapted to use the same meter components.

We illustrate the protocol with an example that includes three principals, as depicted in Figure 4.1: the smart meter, the prover, and the verifier. The smart meter first measures and certifies consumer electricity readings, and then communicates them to the prover using a secure channel. The prover, a consumer-owned device, computes a bill along with a non-interactive ZKP that ensures the bill's validity. The prover sends the bill and the proof to the utility company, which verifies the bill's correctness before accepting it. Below, we describe in detail the computations the meter has to perform, and provide a brief outline of the protocols between the prover and the verifier.

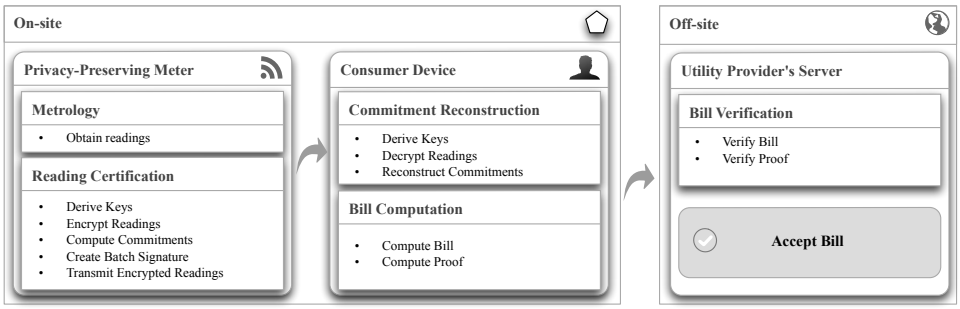

Figure 4.1. Architecture of the privacy-preserving smart metering system. A smart meter, in addition to its metrologic unit, has a microcontroller capable of encrypting and certifying its readings. The meter also has a wireless transceiver used to send encrypted readings to the consumer's device. The consumer uses the information from the meter for consumption planning, and in the computation of bills and corresponding proofs.

Smart Meter Computations. To support privacy protocols, smart meters need to perform the following computations: sensing and measuring electricity usage, deriving session keys, certifying and encrypting readings, and finally transmitting readings to the consumer.

Sensing and measuring electricity. The meter's primary function is sensing and measuring electricity usage. Thus, other computations must not interfere with this fundamental task. We denote Δt as the measurement interval, such that duration between meter readings $t_{i+1} - t_i = \Delta t$.

Deriving session keys. The protocol encrypts readings using a symmetric encryption algorithm before passing them to the user. To ensure the encrypted reading's secrecy, each reading is encrypted with a distinct session key. For every t_i the meter encrypts reading r_i using key $K_i = H_0(K, t_i)$, where H_0 is a secure hash function and K is a master symmetric key known by the consumer. Additionally, the meter derives from the master key an opening value for the commitment $o_i = H_1(K, t_i)$ where H_1 is a hash function.

Certification and encryption. After deriving K_i and o_i, the meter both encrypts the reading r_i using K_i and computes a *commitment* c_i for the reading. More formally,

the meter generates an encrypted reading $Er_i = E(K_i, r_i)$ using a symmetric encryption algorithm, and a commitment $c_i = g^{r_i} \cdot h^{o_i}$ using globally known constants g, h, and their group. The protocol also requires the meter to generate cryptographic signatures for each commitment c_i. To reduce the necessary computations, the protocol computes batch signatures Sig_j for multiple commitments $c_i, c_{i+1}, \ldots, c_{i+k}$.

Network transmission. After the meter encrypts readings and computes batches of signatures, it transmits the batches to the consumer's device (the prover) via the local network. More formally, for each batch j, the meter transmits the following tuples to the consumer: $\{\{t_i\}_j, \{Er_i\}_j, Sig_j\}$. The commitments need not be transmitted, which keeps the overheads of the protocol low.

Consumer Prover Computations. The prover computes the bill's payment and its corresponding proof of correctness. First, the prover derives the session keys $K_i = H_0(K, t_i)$ on the basis of times t_i and the master key K; decrypts the readings r_i from $Er_i = E(K_i, r_i)$, and derives the opening values from each commitment as $o_i = H_1(K, t_i)$. Then all commitments to readings can be reconstructed as $c_i = g^{r_i} \cdot h^{o_i}$ using the public parameters of the commitment scheme and the recovered readings and openings. Finally, a batch of commitments are accepted as authentic after checking the signature Sig_j. This ensures that the received encrypted readings have not been tampered with. After the readings and their signed commitments are available, an arbitrary billing function can be applied to each reading (or aggregates of readings) to establish the final bill. The prover calculates a ZKP of correctness and provides it to the verifier.

The details of those computations, and families of functions that can be practically proved and verified in zero-knowledge are provided in [126] along with the detailed security proofs for the protocol. To summarize, fine-grained meter readings are only available to the consumer, while simultaneously allowing the consumer to self-calculate their bill and ensuring the utility that the consumer has not manipu-

lated or under-reported the payment. Thus, the utility has a guarantee over each bill's authenticity, and the consumer has a guarantee over their data's privacy. To resolve disputes, the meter may optionally store readings and decryption keys to permit audits by a trusted third party.

4.2 Background on Microcontrollers

The computational capabilities of low-cost and ultra-low-power microcontrollers have not developed at the same pace as high-performance microprocessors employed in servers and personal computers. System designers should, therefore, use different means to evaluate the economic feasibility of a cryptographic solution in the low-cost spectrum of embedded devices. This section presents the set of design variables of the various implementations considered in this chapter with the purpose of illustrating their effects on performance and cost.

Moore's law predicted that the number of transistors placed in an integrated circuit would double approximately every two years. This prediction, however, does not directly address two issues that are pertinent to microcontrollers. First, the production costs associated with maintaining this trend have not remained constant. Second, with the addition of more transistors, the problem of efficient power management has significantly increased [47]. As a consequence, microcontrollers that are often constrained by production costs and power budgets have not increased their computational capabilities at the same rate as microprocessors for servers and personal computers. Figure 4.2 illustrates this by showing the evolution in processing capabilities across different technologies.

4.3 Anatomy of a Smart Meter

Figure 4.3 shows the schematics of a smart meter. In general, they are equipped with an analog front end, which is part of the metrologic unit used to convert the

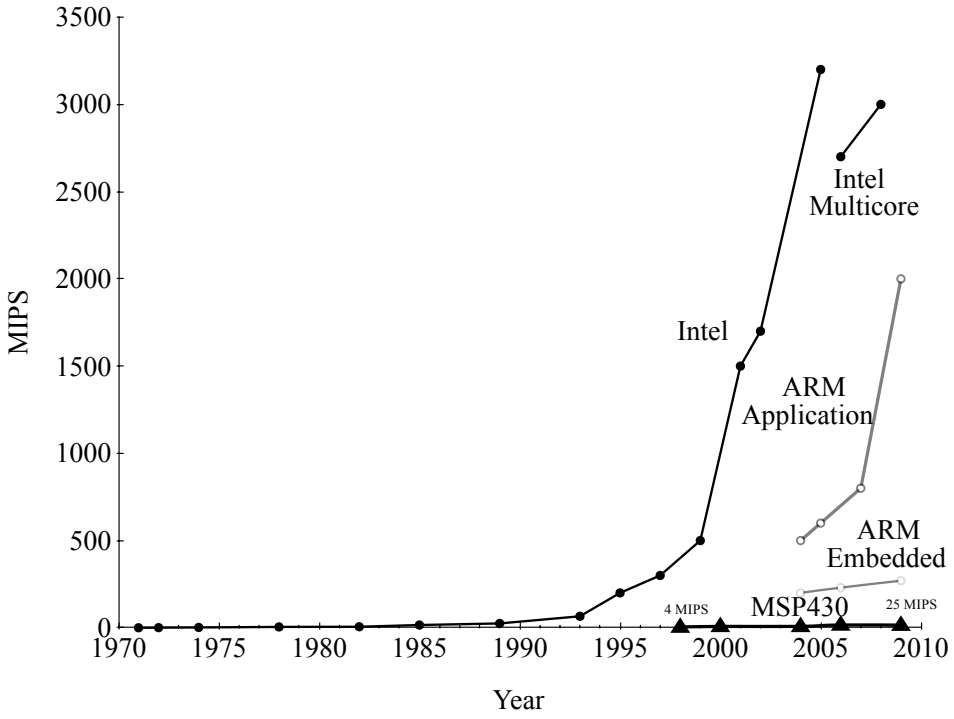

Figure 4.2. This graph provides a visual representation of performance improvements as seen across a few popular architectures. The trends in microprocessors targeting desktop computers and servers, as well as the performance improvements observed in ARM application microprocessors have followed exponential curves. However, the performance improvements observed in embedded ARM microprocessors and MSP430 microcontrollers have followed linear curves [10, 35, 80]. Note that a comparison based on microprocessors using millions of instructions per second does not capture all qualities of a microprocessor, but it helps to illustrate general trends.

data coming from the load sensors and preprocess the measurements before they are passed to the microcontroller unit. The microcontroller unit handles this stream of data as well as the general functionality of storing the data in flash memory, and driving an LCD screen. More modern microcontrollers replace the analog front end with an integrated embedded signal processor. Current deployments of smart meters use microcontrollers that run at clock speeds ranging from 8-25 MHz and have storage ranging from 32-256 KB [116].

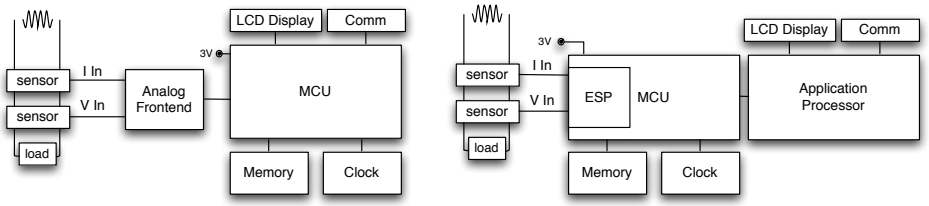

Figure 4.3. Main components of a smart meter. On the left: A simple meter with a single microcontroller unit (MCU) that controls the metrologic unit, storage and communication interfaces. On the right: A smart meter that replaces the analog front end with an embedded signal processor (ESP) and has an additional application processor that controls communication, OS, power monitoring, and analytics.

4.4 Meter Design Variables

Below is the set of design variables that are considered in the implementations. Some of these design variables correspond to features, such as qualitative privacy or security guarantees, e.g. properties of a trust or security model. Other design variables correspond to quantitative properties, for example computation performance, storage and communication requirements. The design variables in these implementations are in one of two categories, system variables or crypto variables. **System variables** include the selection of an MCU platform and a multitasking approach. **Crypto variables** include the selection of a digital signature scheme, and the selection of cryptographic primitives that rely on large integer multiplicative groups

or elliptic curve cryptography. Note that a complete analysis of the economic feasibility of a metering solution should also include a variety of **economic variables**, for example, the costs of implementation, deployment, maintenance and customer support. These economic variables are not considered explicitly in this work. The assumption is that if a solution can be implemented using microcontrollers, such as those in currently deployed meters, and those meters support software updates, then the solution is economically feasible given that it does not require a complete change in infrastructure. For example, rather than forcing millions of deployments, utility companies could offer concerned customers the option to request a meter update that implements the privacy features mentioned here.

4.5 A Privacy-Preserving Smart Meter

The meter needs to compute the algorithms $Commit$, $CLSign$ and DSA, using either large integer multiplicative groups or elliptic curve cryptography. Also a meter needs to compute a symmetric key derivation algorithm $DeriveAESKeys$ to encrypt readings with AES for on-site wireless transmission. These algorithms together produce *certified readings*. The libraries to perform integer or elliptic curve arithmetic are bnlib [121] and Miracl [32] respectively. Additionally, one of the following Real-Time Operating Systems may be used: FreeRTOS [16], SYS/BIOS [142] and MicroC/OS-III [93]. The rest of the implementation is in C, with some minimal amount of assembly code. The focus is on the MSP430 family of microcontrollers with a 16-bit RISC architecture because current deployments already include microcontrollers in this family. The implementations use the evaluation board MSP-EXP430F5438, in combination with the microcontrollers MSP430BT5190 and MSP430F5438A with the radio stack CC2567-PAN1327. The board includes an LCD screen and connectors for radio components. Both microcontrollers are from the same family (MSP430x5xx). Shared characteristics include the availability of a hardware multiplier supporting 32-

bit operations, size of flash (256 KB), frequency (25 MHz), and power consumption ($\sim$ 230 μA/MHz in active mode). The manufacturers designed the MSP430BT5190 for use with the radio stack; however, the MSP430F5438A has a larger RAM (16 KB). The evaluation compares a few ARM microcontrollers and processors. ARM ports are readily available for the arithmetic libraries mentioned above. The code is compiled using IAR Embedded Workbench for ARM version 6.30 [78]. The most significant difference is that the word size for the multi-precision arithmetic is 32 instead of 16 as in the MSP430 implementations. The other microcontroller board is the TI Stellaris Evaluation Board EKB-UCOS3-EVM. The ARM processors measured—intended for smartphone development—are capable of running full Linux distributions; nevertheless, they are measured using IAR Workbench as well.

The full ZKP based billing protocol requires the selection of various building blocks, such as commitment schemes and signatures. The security of these building blocks may depend on either the strong RSA (SRSA) assumption [27], or on the discrete logarithm (LRSW) assumption [28]. One important side-effect of the selection of these building blocks is that in order for the SRSA assumption to hold, the cryptographic operations need to be performed over multiplicative groups of integers with large moduli (1,024 to 2,048 bits in length). However, by leveraging modern Elliptic Curve Cryptography, the designer can use building blocks that rely on the discrete logarithm assumption employing considerably smaller key sizes. Therefore, for the ECC based commitments and ECDSA implementations, the NIST curves P-192 and P-224 [40] were used. For the ECC versions of the CL Signatures, the implementation uses the pairing-friendly elliptic curves $E(\mathbb{F}_{2^{379}}) : y^2 + y = x^3 + x + 1$ and $E(\mathbb{F}_p) : y^2 = x^3 + Ax + B$ with a 512-bit prime p as presented in [138].

The criteria for choosing curve parameters for ECDSA and the commitment scheme in this implementation are well known. However, choosing appropriate parameters for pairing-based cryptography is still an active area of research. That is,

using an elliptic curve implementation for CL-signatures requires an appropriate bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{H}$ that is non-degenerate and easy to compute. There is no unique way to obtain this map using elliptic curve groups $\mathbb{G}, \mathbb{H}$. While most protocols, such as signatures and identity based encryption protocols are designed using a *type-1* pairing, it is often possible to use a *type-3* pairing. The latter are typically more efficient in practice. In other words, protocols often assume the existence of a pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{H}$ (type-1). However, in some cases the designer can implement a protocol that assumes the existence of a pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{H}$ with $\mathbb{G}_1 \neq \mathbb{G}_2$ such that there is no isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$ (type-3). This implementation uses type-1 pairings on a super-singular curve defined over $GF(2^m)$ using the η_T pairing [15] and on a super-singular curve defined over $GF(p)$ using a modified Tate pairing [137].

In order for the curves to provide an adequate security guarantee, the size of the key must be large enough so that the corresponding dilogarithm problem in $\mathbb{H}$ is hard. For the purposes of the particular billing protocol described in this chapter, a smart meter needs to compute signatures and not necessarily verify them. Therefore, one would want to make operations on the curve as cheap as possible, even if that means computing more expensive parings on the consumer's device. For more details on pairings see Devegili et al. [42].

4.6 Experimental Evaluation

The experimental results here show that a range of microcontrollers are capable of generating certified readings in a few seconds. Using ECC primitives, it is possible to achieve the best performance. In the case of a Texas Instruments MSP430F5438A, a reading can be computed in under 10 seconds. The use of ECC also allows for adequate security key sizes. The larger the precision of the numbers that are used, the larger the RAM requirements; therefore due to the limitations on microcontrollers, using traditional primitives with integers with large precisions is not possible.

4.6.1 Impact of Platform Selection

The cryptographic operations *Commit* and *CLSign* are implemented using microcontrollers from two of the most popular families, specifically, a microcontroller MSP430F5438A with 256 KB flash, 16 KB RAM and a microcontroller Stellaris LM3S9B92 (ARM Cortex M) with 256 KB flash, 96 KB SRAM; and two ARM application microprocessors OMAP3 (ARM Cortex A8) and OMAP4 (ARM Cortex A9) capable of running full Linux operating systems. These two microprocessors are commonly used in smart phones. The performances of these operations on these platforms are summarized in Table 4.6.2.

4.6.2 Impact of Multitasking Approach

Meters need to be able to interrupt cryptographic computations periodically to perform measurements, logging and communication. One way of handling multitasking is with the use of an RTOS. Another way is the modeling of an application using a finite state machine and the implementation of it using timers and interrupts. Generally, the footprint of an RTOS is larger than the footprint of a state machine approach. The following three RTOS are considered here: FreeRTOS, SYS\BIOS and μC-OSIII. The configurations for each of the RTOS uses 4 KB, 16 KB and 12 KB of code size respectively. The finite state machine requires approximately 2 KB of code. RTOS have the capability of managing memory; some by reserving particular regions of the stack for different applications, and some by allowing for the use of dynamic memory allocation even with multiple heaps, such as SYS\BIOS. It is typically not a trivial engineering exercise to fit each cryptographic algorithm in RAM. Note that the system designer should probably base the decision of whether or not to use an RTOS on the necessity of additional required functionality, such as occasional tasks like secure updates, secure audits, key exchange and key revocation, etc.

	MSP430F5438A	LM3S9B92	Cortex-A8	Cortex-A9
Operating Freq	25 MHz	80 MHz	720 MHz	1 GHz
Operating Power	330 - 690 μW	333 - 524 mW	0.4 W	1.9 W
Family Price Range	\$0.25 - \$9	\$1 - \$8	\$41 - \$46	+\$50
Commitments - Key Size 1,024 bits				
Avg. Running Time	19.56 s	0.82 s	51 ms	36 ms
DSA Signatures - Key Size 1,024 bits				
Avg. Running Time	2.71 s	0.13 s	8 ms	6 ms
CL Signatures - Key Size 1,024 bits				
Avg. Running Time	43.1 s	2.3 s	150 ms	81 ms

Table 4.1. Running time of commitments and signatures across multiple platforms. The tasks are run exclusively and uninterrupted on each of the platforms. The signatures are performed on 16 bytes of data. DSA uses a 1,024-bit prime p, a 160-bit prime q, and SHA-256. The timing does not include the generation of randomness, which depends on the source. Prices are in USD (Sept., 2011).

4.6.3 Impact of ECC Utilization

The code sizes of the bnlib [121] and Miracl [32] libraries and their RAM requirements depend on which features that are included. In the experimental setting using a microcontroller MSP430F5438A, the code size of Miracl was 23 KB and the code size of bnlib was 18 KB. The performance of bnlib and Miracl on non-ECC arithmetic is comparable. The running times of the same operation using either library differed by less than 5% of the total computation time of the operation. The RAM footprint for various functions is summarized in Table 4.3. As one can see, given a security level, ECC cryptographic primitives utilize RAM more efficiently. Similarly, Table 4.2 shows that given a microcontroller and a security level, an improvement in performance of about one order of magnitud can be achieved by using elliptic curve primitives.

4.6.4 Impact of Signature Scheme Selection

Table 4.2 shows running times for performing a CLSign algorithm with four readings. Note in particular the benefit of using an elliptic curve based library. If a

Algorithm	Key Size	Library	Time
Commit	1,024	bnlib	19.9 sec
Commit	2,048	bnlib	303.0 sec
ECC Commit	192	miracl	5.6 sec
ECC Commit	224	miracl	8.3 sec
CLSign	1,024	bnlib	41.2 sec
CLSign	2,048	bnlib	313.8 sec
ECC CLSign	379	miracl	6.7 sec
ECC CLSign	512	miracl	35.6 sec
AES Key Gen	128	miracl	0.1 sec

Table 4.2. Running time of commitments (single reading) and signatures (4 reading batches) on an MSP430F5438A at 25 MHz. These times are obtained when the algorithms are running exclusively and uninterrupted. Miracl is used for the elliptic curve versions (§4.5). The key sizes are in bits.

designer uses elliptic curves, he or she can reduce a monthly batch signature with 1,440 readings (one reading every half hour) from 15.6 hours to 2.5 hours. If the designer assumes a different trust level in which zero-knowledge is not required, signatures are less expensive. On an MSP430F5438A at 25 MHz, signing a 16-byte message using regular DSA with a 1,024-bit prime p, a 160-bit prime q, and SHA-256 takes 2.71 seconds excluding the generation of randomness, which depends on the source. Signing a 16-byte message using ECDSA using a curve in $GF(p)$ for a 192-bit prime and SHA-256 takes 3.78 seconds excluding the generation of randomness. DSA signatures scale better than CL-signatures because the only overhead for a larger message would be the cost of the hash, which for the computations above is less than 0.01% of the computation.

4.7 Feasibility and Costs in Real-World Deployments

This section discusses a strategy for estimating the cost of deploying privacy preserving smart meters according to the system variables discussed previously (§4.5).

Algorithm	Key Size	RAM
Commit	1,024	5.8 KB
Commit	2,048	10.2 KB
CLSign	1,024	6.3 KB
CLSign	2,048	11.3 KB
ECC Commit	192	2.2 KB
ECC Commit	224	2.5 KB
ECC CLSign	379	3.1 KB
ECC CLSign	512	3.6 KB
AES Key Gen	128	2 KB

Table 4.3. RAM utilization for the various algorithms we implement on an MSP430F5438A all using the Miracl library. The measurements do not include RAM utilization by an RTOS, a radio stack or I/O.

4.7.1 Cost Estimation Strategy

Step 1: Determine the performance and power requirements. The first step is to determine the acceptable levels of general computational performance and the power requirements of the meter. Depending on the specific application, meter readings may need to be certified with a frequency of seconds, minutes or hours. Also, the meter may need to operate on a battery. Thus, using an ultra-low-power microcontroller may be the difference between replacing the battery every few years or every few days. For example, the performance of an LM3S9B92 MCU may seem attractive for its ratio of cost/performance. However, the power consumption is roughly three orders of magnitude greater than the MSP430 MCU. Mobile processors are still far from being ultra-low power, although their computational and storage capabilities are increasing faster than those of the MCUs.

Step 2: Determine the code and RAM requirements. Once the performance and power requirements are met by a family of microcontrollers, it is then necessary for the designer to estimate the code size and RAM requirements for the implementation of the reading certification functions in a meter, taking into account whether multitasking needs to be supported.

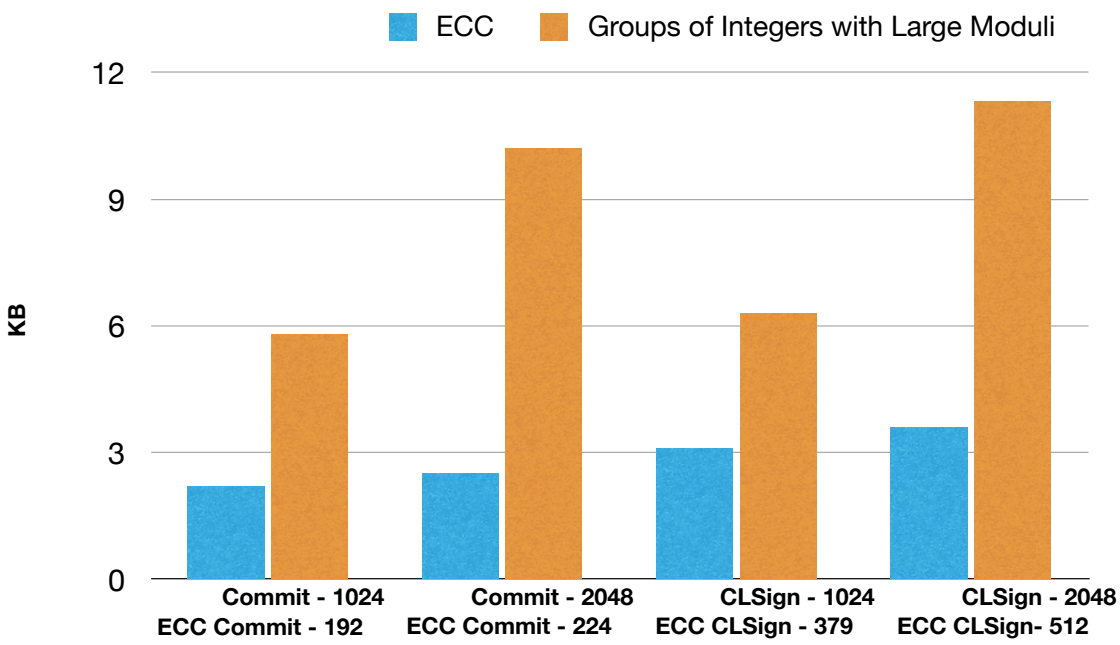

Figure 4.4. Memory requirements on an MSP430F5438A.

4.7.2 Economic Feasibility

Existing smart meters that have the ability to be remotely updated rely on microcontrollers similar to those used in the implementation in this chapter. If a microcontroller in the MSP430 family is used, it is possible to generate commitments and CL-signatures every 10 seconds when running at 25 MHz or every 28 seconds when running at a more conservative 8 MHz. Thus, a remote update that enables meters with privacy preserving functionality appears feasible.

Other metering applications may require that readings be certified at a finer granularity, for example every one or two seconds. This would require higher computational performance and larger storage than is currently available on low-cost ultra-low-power microcontrollers. For this reason, while obtaining certified readings at fine granularities is technologically feasible, it is to this date a feature that may incur a greater cost. Finally, in some circumstances, billing transactions may be required to take milliseconds. In that case, only high-end mobile processors could provide the required

performance, and thus the cost of that application would be high based on current technological trends.

While the analysis in this section does not cover all manufacturers of low cost MCUs, other leading manufacturers have similar offerings. For example *Atmel* also has AVR ultra-low-power microcontrollers, and various ARM based MCUs comparable to those discussed here. *Microchip* has the PIC microcontroller line with 8-, 16- and 32-bit MCUs. 8-bit microcontrollers are not considered because they are perhaps too constrained for the kind of crypto application described here.

Best Utilization of Resources

The best security/cost ratio can be achieved by using ECC primitives (§4.6). If current MCUs are targeted, maximizing the use of RAM can be achieved via ECC. Looking toward the future, performance will most likely regain importance due to the increasing economic feasibility of Ferroelectric RAM (FRAM), a kind of memory that enables high-performance on ultra-low-power microcontrollers, with a unified memory model. Texas Instruments has started to ship MCUs with 16 KB of FRAM ($1.20 USD), and they are already producing chips with 4 MB of FRAM [119].

4.8 Related Work

Prior work exists on distilling information about appliance usage from power traces. However, prior approaches assume knowledge of the appliances in a home or take appliance measurements in order to use *supervised learning* techniques to disambiguate events. For example, Patel et al. [118] use individual traces from USB oscilloscopes to disambiguate power traces of particular appliances. Jiang et al. [85] solve a similar problem by using a wireless sensor network to monitor building energy usage. Similarly, Lam et al. [95] classify appliances based on fine-grained load signatures. More accurate monitoring for utilities has many benefits, as prior work

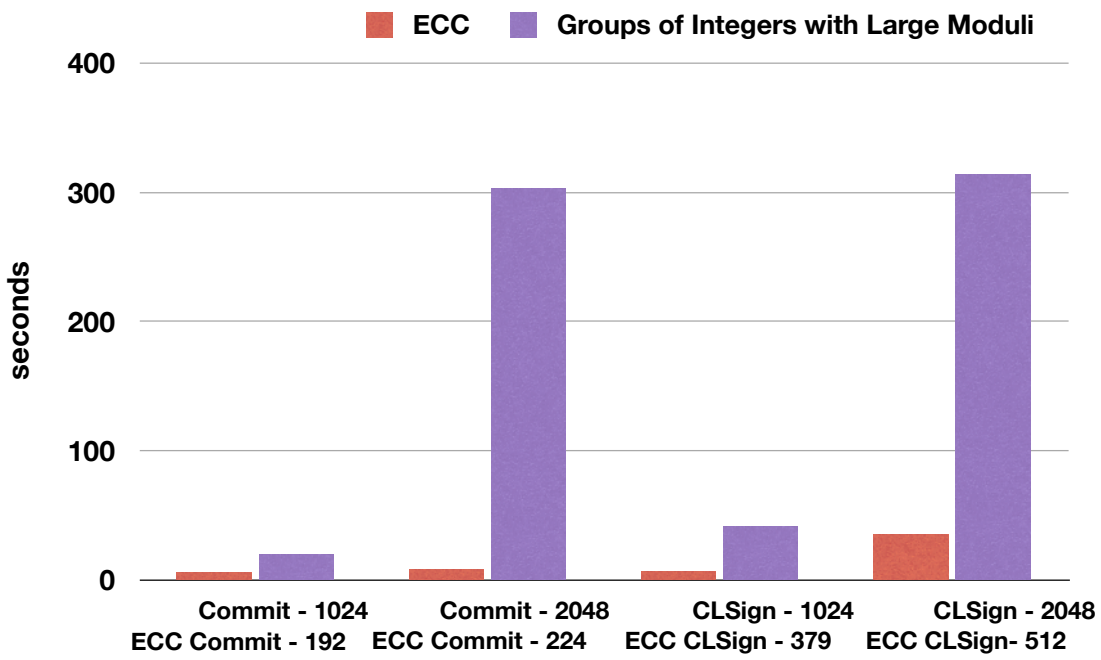

Figure 4.5. Impact of ECC on computation using an MSP430F5438A.

discusses, e.g., [102], [39], [91] but they do not propose techniques to preserve privacy. An alternative approach to protect privacy is adding noise to load signatures using rechargeable batteries [87].

Besides the cryptographic solutions described by Rial et al. [126] and Molina-Markham et al. [108], there are other proposals to provide privacy in the context of smart grids, including [82]. Methods from the field of differential privacy have been suggested to provide privacy in this setting, for example [140, 33, 37]. None of these works explores the feasibility of using low-cost or ultra-low-power microcontrollers.

4.9 Conclusion

This chapter demonstrates that ZKP based solutions to mitigate the problem of smart metering information leakage are economically feasible. Evaluating the cost of a cryptographic solution in an embedded system such as a smart meter depends first on the family of microcontrollers used, then on the storage and RAM requirements, and

finally on additional features such as communication and user interface. An empirical analysis shows that with the use of Elliptic Curve Cryptography, it is possible to reduce the RAM requirements by about 50% and obtain performance improvements of one order of magnitude, in comparison with using primitives over groups of integers with large moduli—thus obtaining a better performance/cost ratio.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 5

BAT: BACKSCATTER ANYTHING-TO-TAG COMMUNICATION

This chapter presents BAT, a networked system designed from the ground up to enable RFID applications beyond inventorying [106].[1] Computational RFID prototypes are limited by networking abstractions that impose narrow preconceptions about topologies and applications. These prototypes support programmability and integrate a wide array of sensors, which open the door to more varied applications [24, 25, 147, 125, 86]. The implementation of these applications on constrained platforms will need primitives that seamlessly support communication among tags and also with other devices. While overlays on top of existing protocols are possible, they introduce inefficiency because of packet formats designed explicitly for the tag inventory paradigm.

Supply-chain RFID technologies were developed with narrow design goals in mind, primarily inventorying or data collection. The EPC Gen 2 protocol—the de-facto standard for UHF RFID communication [55]—reflects these design goals by offering a limited set of commands not well-suited for bulk data transfers [71]. The assumption that tags will not implement any functionality locally and will instead act as simple static identifiers in most cases also imposes other restrictions. Tags must conform to

[1]This chapter draws from previously published work in:"BAT: Backscatter Anything-to-tag Communication" by A. Molina-Markham, S. S. Clark, B. Ransford and K. Fu. Chapter in Wirelessly Powered Sensor Networks and Computational RFID. Springer Signals and Communication. J. R. Smith (Ed.) December 2012. To appear.

a rigid state machine allowing no time to sense or process data. They are instead required to respond to reader commands rapidly whenever in range.

Without protocols that present appropriate abstractions for richer applications, computational RFIDs (CRFIDs) must use limited resources to shoehorn these new applications into a suboptimal paradigm [86]. Unlike supply-chain RFID tags, CRFIDs have their own microcontroller units: they are capable of running their own application logic and managing their own memory and communication links. They are, therefore, able to participate in radio protocols that are more flexible than Gen 2. Table 5.1 summarizes some key differences between CRFIDs, supply-chain tags, and battery-powered sensor nodes (motes).

The BAT protocol and software stack provides fast and flexible **b**ackscatter **a**nything-to-**t**ag communication. The key insight is that BAT separates tag applications from the networking stack by ensuring that the networking layer does not impose unnecessary constraints on abstractions. BAT separates memory management from the networking stack such that tag applications can efficiently store data in arbitrary locations without needing several interactions with the network stack. This logical detachment between the networking stack and applications does not impede secure anything-to-tag communication even using current CRFID prototypes.

In order to evaluate BAT's practicality, it was implemented on a current CRFID prototype and a relay using software radio (§5.4). BAT allows tags to send and receive data via untrusted relays with a maximum throughput of 18 Kbps. CRFIDs generate encrypted payload at a rate of 61 Kbps.

Contributions

The contributions of this chapter are:

Networking Stack. This chapter presents the design of a networking stack for backscatter devices via untrusted relays that offers more appropriate abstractions for

	Characteristics	Deployments
Supply Chain Tags	Externally powered No MCU/OS Externally managed memory Minimal network interoperability Tags do not pull data	Inventorying applications Inexpensive Low maintenance Long term Physically accessible or inaccessible Star-like topology
CRFIDs	**Externally powered** **Short power cycles** **Have an MCU** **Could run an OS** **Self-managed memory** **Networking through BAT** **Tags can pull data**	**Sensor networks** **Payment tokens** **Inexpensive** **Low maintenance** **Long term** **Physically accessible or inaccessible** **Complex topologies**
Motes	Battery powered Long power cycles Have an MCU Run an OS Self-managed memory Various IP stacks Nodes can pull data	Sensor networks More expensive Higher maintenance Shorter term Physically accessible Complex topologies

Table 5.1. CRFIDs differ from supply-chain tags in that CRFIDs can manage their own memory and application logic. CRFIDs also differ from motes in that CRFIDs have shorter power cycles and depend on RFID readers for power and communication.

tags that can manage their own memory and application logic, but depend on relays for power and communication. This networking stack allows computational RFIDs to send and receive messages to and from systems across the Internet to implement a variety of applications.

System. This chapter describes the results of implementing BAT using a UMass Moo [148] and software radio [5]. It demonstrates that a networking stack for CRFIDs using untrusted relays may be implemented while adding marginal computational cost for providing secrecy and integrity. Prior work implicitly assumes that RFID-scale devices adhere to a model in which tags gather data that is collected by a trusted reader and tags do not act as independent systems.

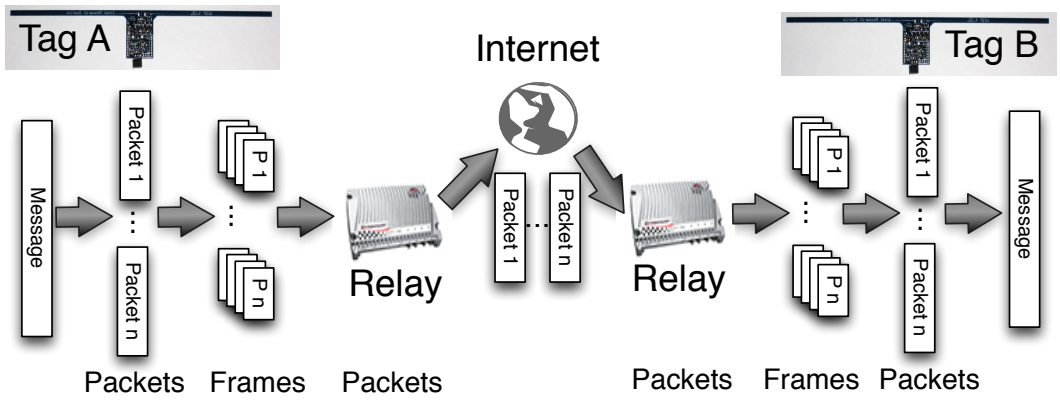

Figure 5.1. BAT Overview: Relays collect packetized messages from tags and forward them through other relays, which deliver them. Packets are split into frames locally to maximize throughput.

Key Management. This chapter shows that identity based key agreements such as the the non-interactive Sakai, Ohgishi, and Kasahara (SOK) [132] key agreement are feasible on CRFIDs. This is one of the first works that implements a key agreement suitable for RFID-scale systems that does not require the on-line verification of a public key during a key agreement.

5.1 Anything-to-Tag Communication

The prevailing protocol for supply-chain RFID systems, EPC Class 1 Gen 2 [55], is designed around the abstraction of RFID tags as remotely addressable memory. The Gen 2 commands a reader may use fall into two categories. Readers use *singulation* commands to search a tag population for a single tag; then they may issue *memory access* commands to read or write its protocol-defined memory banks.

The tags-as-memory abstraction is entirely appropriate for supply-chain applications, but it imposes several hindrances on applications running on CRFIDs. First, it restricts communications to those that can be formulated in terms of reader-to-tag memory access commands and simple tag responses. It is possible to reuse fields of existing commands to carry information—for instance, the Gen 2 *write* command takes

an arbitrarily long *WordPtr* parameter intended to specify where to start writing—but reader support for embedding arbitrary data in these fields varies. Second, the tags-as-memory abstraction as implemented in Gen 2 readers lacks the notion of a shared device-address space, so it cannot express useful mainstream networking concepts such as multicast, anycast, or peer-to-peer (tag-to-tag) communication. Finally, it offers no facility for rich tag messages. Tags respond to most commands with short fixed-length status messages and to *read* commands with the requested data. Tags cannot address messages to other tags.

In contrast to Gen 2, BAT comprises a set of abstractions designed to enable *anything-to-tag* communication, in which multiple tags can communicate with one another and with external entities such as Internet resources. Figure 5.1 depicts BAT at a block level.

BAT's increased flexibility and efficiency in comparison to Gen 2 result from several key distinctions. In terms of flexibility, tags are *addressable network nodes* rather than simple memory stores. Each tag and each relay has an address in a shared address space. In addition, tags and relays formulate their communications as explicitly addressed messages (packets) rather than implicitly addressed responses. Finally, BAT enables confidential tag-to-tag communication via untrusted *relays*.

In terms of efficiency, BAT conceptually replaces supply-chain RFID readers with BAT *relays* that maintain message queues. As in an IP network, interactions between relays and tags consist mainly of packet-exchange commands. Also, tags and relays negotiate message sizes to adapt to lower- or higher-quality links. Messages from tags and relays have the same message format; a relay can forward messages to other tags without altering them.

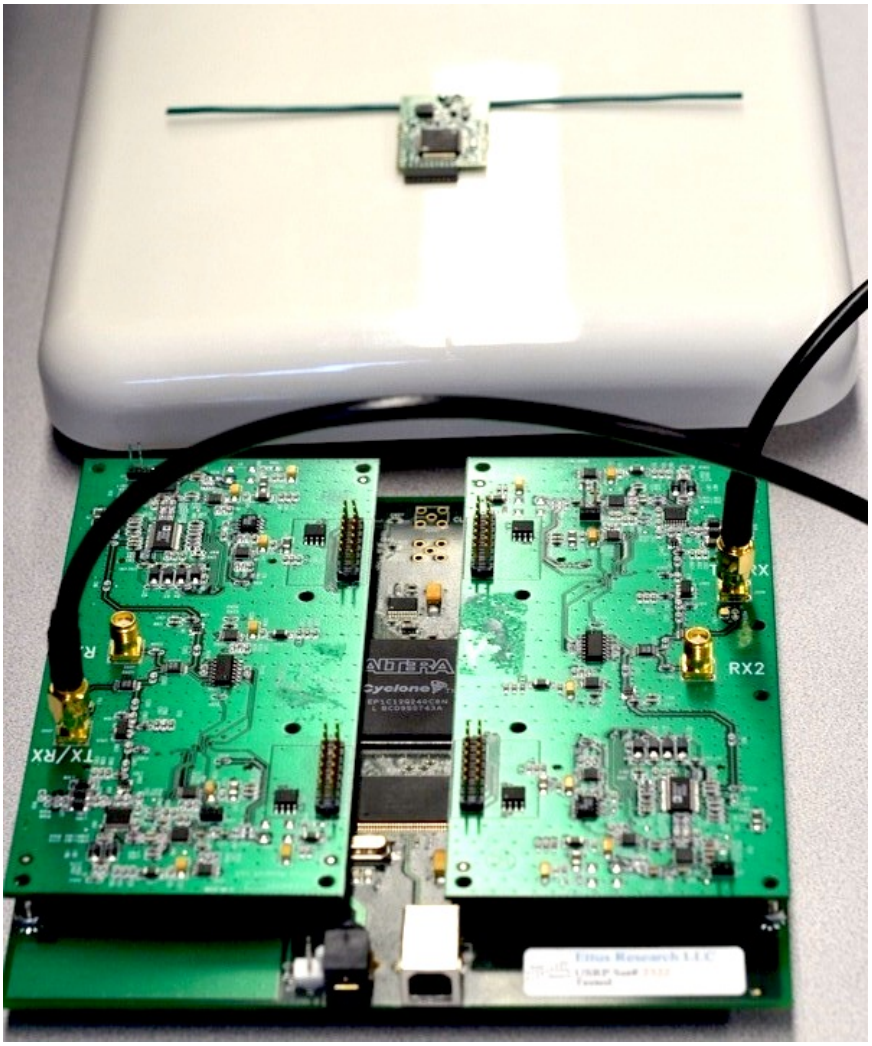

Figure 5.2. Hardware used to implement BAT. The relay (foreground) is a USRP with RFX900 daughterboards driven by GNU Radio. The UMass Moo (background) is a CRFID tag derived from the DL WISP 4.1 [135].

5.2 BAT Design Overview

BAT's basic model consists of relays that move into the vicinity of tags and offer to provide power and communication. Messages follow a path conceptually similar to mail in the U.S. postal system: items to be delivered are picked up at their sources (BAT tags) by mail carriers (BAT relays), routed through the postal system (powered networks of BAT relays), and delivered by local carriers (BAT relays) to their destinations (BAT tags).

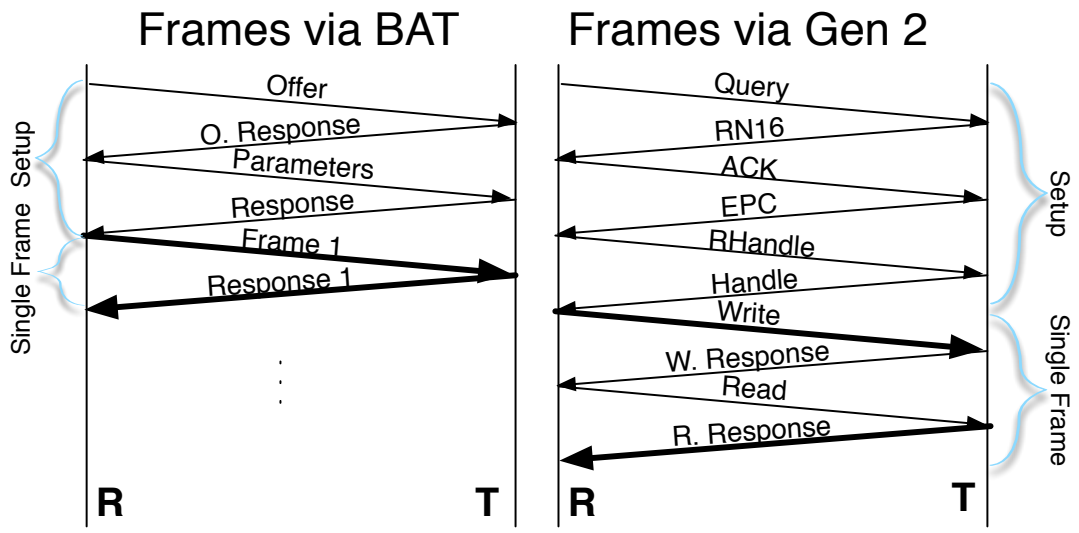

Figure 5.3. Gen 2 requires messages to achieve singulation before Read/Write commands are issued. In order to implement custom round-trip messages as in BAT ($R \rightarrow T$, $T \rightarrow R$), a Write command would have to be followed by a Read command.

A BAT system comprises a set of CRFID tags, all of which are programmed initially by a trusted tag programmer, and a set of interchangeable relays that understand common frame and packet formats. A relay is a powered device akin to the RFID readers used in supply-chain applications. It powers tags by transmitting RF energy and can exchange messages with them via BAT's link layer. Multiple relays may be connected to a centralized application controller that knows the network's topology and can coordinate message routing; relays may also independently query their surroundings for tags for which they have messages. The organization and operation of relays is application-dependent. Figure 5.5 summarizes a BAT interaction between a relay and a tag.

Experimental results demonstrate the benefits of BAT's packet-framing and frame-size-negotiation mechanisms. When there is a high-quality link between relays and tags, larger frame sizes may result in higher throughput (§5.4). However, when the link is lossy or noisy, frame sizes must be smaller in order to achieve sustained communication.

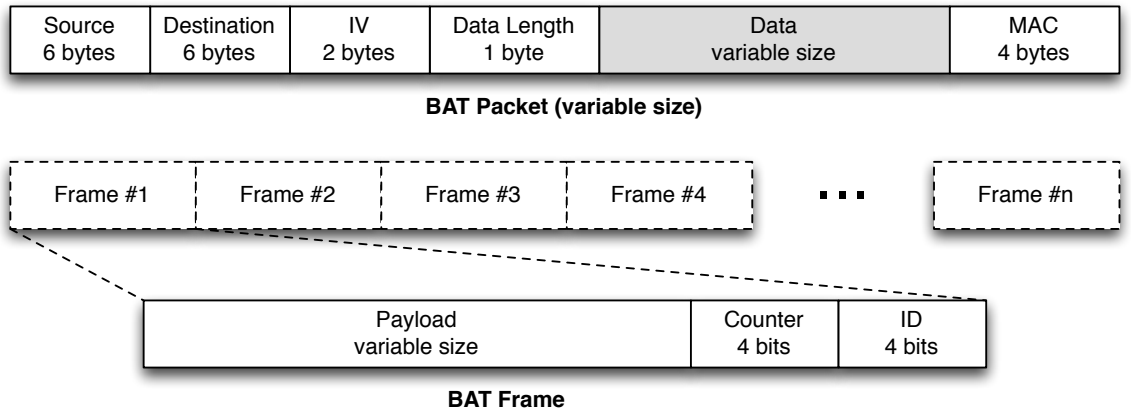

Figure 5.4. The *framed* packet format accommodates up to $2^8 = 256$ bytes of data payload per packet by breaking packets into one or more frames of variable size.

The packet format depends on the higher network layers that are used above BAT. In this prototype implementation, we chose to follow a format similar to that proposed by Karlof et al. for sensor networks [89]. Another option could be 6LoWPAN [110]. A packet is split into frames as illustrated in Figure 5.4. A frame includes a group ID, a frame-counter, and the payload. All frames with the same group ID are concatenated sequentially. The frame-counter gives the position of the frame within the packet. Once all frames are concatenated, the IV together with the source and destination fields provide a unique packet identifier. The source and destination fields consist of an *ID*, a *group* and a specified *domain* in order to facilitate routing: when a tag A has a message for a tag B, A sends the message addressed to B's ID at the specified domain. Then, B retrieves messages addressed to it opportunistically. When a tag A wants to send a message to a particular group of tags in the local domain, it can simply *post* a message to the given group at the local domain. BAT implements the retrieval and collection of messages in a given domain, and the network layer routes and delivers messages between domains.

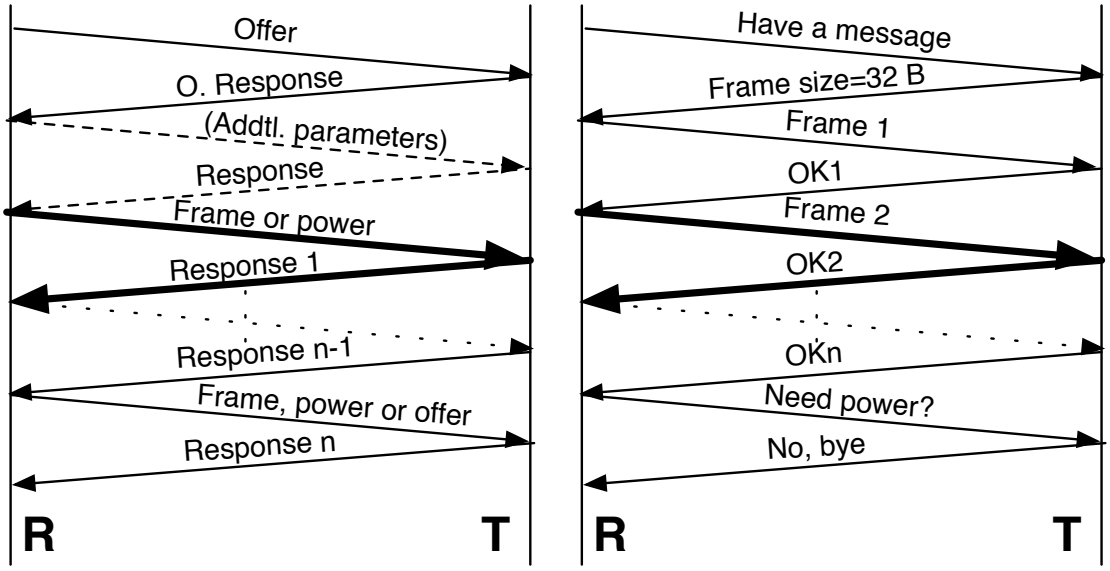

Figure 5.5. BAT messages. A tag may request a relay to: *deliver a message*, *provide power*, or *check for messages addressed to the tag*. In some cases singulation may be necessary, but not always.

In order to facilitate the use of untrusted readers, BAT encrypts message payloads. A per-packet message authentication code (MAC) allows the recipient to detect tampering or transmission errors upon receipt of a packet (§5.5).

5.3 Applications

BAT's abstractions are more closely matched to conventional networking abstractions than those of Gen 2, and they consequently enable a variety of functions. For example, multiple tags that need to collaborate toward a larger goal, such as collective time synchronization, can exchange messages with relays and tags to reach consensus using Paxos [96]. Tag-to-tag messaging could also enable tags to solve optimization problems collectively with *ant algorithms*, biologically inspired algorithms that harness the power of collective behavior in solving complex problems using simple agents [109]. Ants are known to collectively compute the shortest path between their nest and a source of food by each depositing a certain amount of pheromone

on their paths while walking, attracting other ants to follow the same path. Similar mechanisms can be generalized to allow swarm agents to solve problems collectively, such as the traveling salesman problem [46]. The coordinated action of multiple independent agents has also been studied as *amorphous computing* [11], inspired by metaphors in biology and physics.

Some applications for CRFID networks, such as those listed below, could benefit from a network stack that provides secrecy and integrity. The feasibility of using these tags to compute an aggregate in a privacy-preserving manner is also considered (§6).

Infrastructure Monitoring

Infrastructure monitoring is currently a time-consuming and largely manual task. Many infrastructure elements are only subjected to periodic spot checks that allow engineers to gauge the overall condition, but this type of information is not always sufficient. The collapse of the Mississippi River Bridge in Minnesota, which killed 13 people, is one striking example of this problem. While a 2005 inspection rated the bridge as "structurally deficient," repairs were delayed while other bridges thought to be in worse condition were fixed [143].

A persistent sensor network deployment offers a seemingly simple solution to this problem. With constant, automated monitoring, sudden changes in infrastructure condition could be detected and addressed quickly. Unfortunately, deploying wireless sensors that each operate on a battery introduces a new maintenance problem. Every monitoring point on the infrastructure will require battery replacement eventually. One way of shielding sensors from the environment would be to actually embed them in concrete at the time of construction, but the need to replace batteries precludes the possibility of embedded deployments.

CRFIDs using BAT could support fine-grained sensing without the need for regular maintenance, as they forgo batteries. The most straightforward approach would

Figure 5.6. Bridge with a monitoring system in Flint, MI. Data is collected and transmitted to a remote computer once per hour. System and photo by Fondriest Environmental [60].

be for users to collect data about a structure's condition by moving along the infrastructure with a relay, collecting data from individual tags. A more versatile system could be constructed by leveraging the tag-to-tag communications enabled by BAT. In this model, tags could exchange encrypted data among themselves and privately aggregate the data for general release. This approach would allow the tags themselves to enforce data-release policies. Any relay could query a tag and get a generic reply, such as an overall safety rating, but could not determine which tag sensed faults or the exact nature of those faults. Only trusted relays could elicit detailed reports indicating which tags have sensed structural weaknesses and what the exact sensor readings were. In this way, the public could have visibility into infrastructure state, but saboteurs interested in destroying infrastructure would still be prevented from gaining any sensitive information. To be more specific, we could use BAT to implement the protocol proposed by Shi et al. [140], which allows untrusted aggregators to compute privacy-preserving aggregation using data generated by CRFID tags (§6).

Body Area Networks.

BAT could benefit Body Area Networks (BANs) by allowing implanted sensors and actuators to intercommunicate via an untrusted relay, rather than expending their limited battery reserves to communicate with one another directly. Currently, implantable medical devices, such as pacemakers, leverage the same non-rechargeable battery responsible for lifesaving operations to communicate with device programmers. Replacing the battery in an implantable device requires the surgical replacement of the whole device, so reducing the load on the battery is a major goal for implantable medical devices [75].

A glucose monitor and injection system for diabetics is one example of a non-implantable BAN device that could benefit from CRFIDs using BAT. Rather than using a single large, heavy piece of hardware to sense, report, and inject insulin, it

would be desirable to decompose the system into smaller, discrete sensor and actuator devices. A CRFID could serve as the insulin sensor, allowing for more discrete placement. The sensor could continuously report readings to a mobile phone capable of presenting them to the user in a digestible manner. The phone would also act as a router, triggering the discrete insulin pump to deliver medication when necessary. While battery-powered sensors also allow the glucose monitoring system to be decomposed in this manner, creating two devices that require batteries exacerbates the issue of battery replacement. CRFIDs running BAT would allow the lightweight, flexible decomposition of this BAN device.

BAT can provide both confidentiality and integrity for BANs, but, depending on the application requirements, confidentiality may be optional. When the functions performed by the tags are vital, confidentiality may be a priority. In some situations, however, the relay could also display information to the user, such as readings from one of the sensors. BAT can also support this model, which could allow partially trusted relays to decrypt packets, but not to create packets that can be successfully verified by a tag.

Roadway Monitoring

Real-time data on road safety can prevent many accidents. Approaching cars could be alerted of a dangerous pothole or piece of detritus on the highway so that drivers have the opportunity to react appropriately. The standard monitoring technique for road safety is a combination of opportunistic checks by municipal workers and notifications from the public. When those responsible for maintaining roads eventually become aware of problems, drivers must wait until a worker can be dispatched to put up a warning sign before they are finally notified of obstacles in advance.

A network of CRFIDs using BAT could provide real-time data directly to passing vehicles and also report conditions to the local highway department or other respon-

sible parties. Tags embedded directly in the roadway could monitor road conditions via vibration sensors, for example, and report data to passing vehicles without any human intervention. The passing vehicles would provide the tags with both power and routing. Vehicles could also send notifications back to tags as they move along the road to propagate warnings along the roadway so that future vehicles receive earlier warnings. This would allow the tags to not only notify drivers of obstacles, but to forward messages back to a trusted endpoint like the local highway department. The opportunistic use of vehicles as routers would obviate the need for a large and expensive deployment of dedicated RFID relays to act as routers.

While road condition information is not generally considered private, authentication and integrity would be necessary for this application. Without both properties, malicious parties could falsify packets to overload the network or misdirect drivers and repair workers. BAT's use of CMAC to ensure integrity guarantees that adversaries cannot forge arbitrary messages. Because BAT uses keys shared only by a pair of tags, a receiver tag would be able to verify that the message received was sent by one other tag. The identity based key exchange allows for the specification of an expiration date.

5.4 BAT Evaluation

The practicality of BAT is evaluated via a prototype implementation using a UMass Moo. The implementation shows that BAT's networking abstractions do not add significant communication overhead in comparison to supply-chain RFID networking, and BAT's mechanisms to enable the negotiation of optimal frame size generally result in an increased bidirectional throughput.

Figure 5.7. Roadway monitoring via video cameras [141]. The use of CRFIDs could allow for the monitoring of larger areas with less maintenance and infrastructure than video monitoring.

5.4.1 Prototype Implementation

The relay implementation uses the GNU Radio software-defined radio platform to drive Universal Software Radio Peripheral (USRP) hardware [5, 57]. This hardware and software combination allows complete specification of message structure in both directions. It builds upon Buettner's RFID reader implementation [26]. Tags are implemented using the UMass Moo [148], see Figure 5.2. This prototype has an MSP430F2618 with 8 KB of RAM and a maximum operating frequency of 4 MHz under harvested power. The tag prototype fixes some transmission options to specific values for simplicity of implementation; in particular, it uses phase-reversal amplitude shift keying (PR-ASK) modulation with pulse interval encoding (PIE) from relay to tag, and phase-shift keying (PSK) modulation with Miller-4 encoding from tag to relay. Tags are implemented so that facilitation of power requests and rate adaptation are transparent to applications; individual applications on tags do not need to be aware of physical conditions or power budgets.

5.4.2 BAT's Throughput

BAT negotiates optimal frame size at the beginning of an interaction to maximize throughput. When a tag needs to receive more than a few bytes of data on a regular basis, using a fixed, conservative frame size adds significant overhead. The optimal frame size may vary considerably depending on operating conditions, the tag's internal memory speed, or the particular combination of reader and tag. Experiments show that the throughput increases linearly with the frame size to achieve the highest throughput of 18 Kbps with a frame size of 112 bytes. With larger frame sizes, the error rate increases, and, therefore, the goodput decreases. Figure 5.8 illustrates the average throughput for this prototype using BAT when sending data split into frames of various sizes.

Because there is not yet a wide variety of CRFID prototypes, one can use high-capacity Gen 2 tags and readers to further illustrate the need for variable frame sizes. In this experiment, the payload of the Gen 2 BlockWrite command carries data from a reader to a tag. Two commercial tags were used: the Xerafy Sky-ID and the Ramtron MaxArias, which can store up to 8 KB and 2 KB of user data respectively. The size of the payload was varied and the throughput was recorded. The readers in the experiment are the ThingMagic reader M5e and a Ramtron MaxReader Development Kit.

When using the *User* memory bank of the Xerafy Sky-ID, the number of Block-Write commands necessary to fully write the payload value increases linearly with frame size, except when the tag is placed less than 3 mm from the reader's antenna. In order to achieve a high successful-write rate, a BlockWrite command should carry at most 4 bytes of data. MaxArias tags would achieve the best throughput when using BlockWrite commands with 56 bytes of data at a time. In order to implement BAT-style communication atop Gen 2, a BlockWrite command would have to be fol-

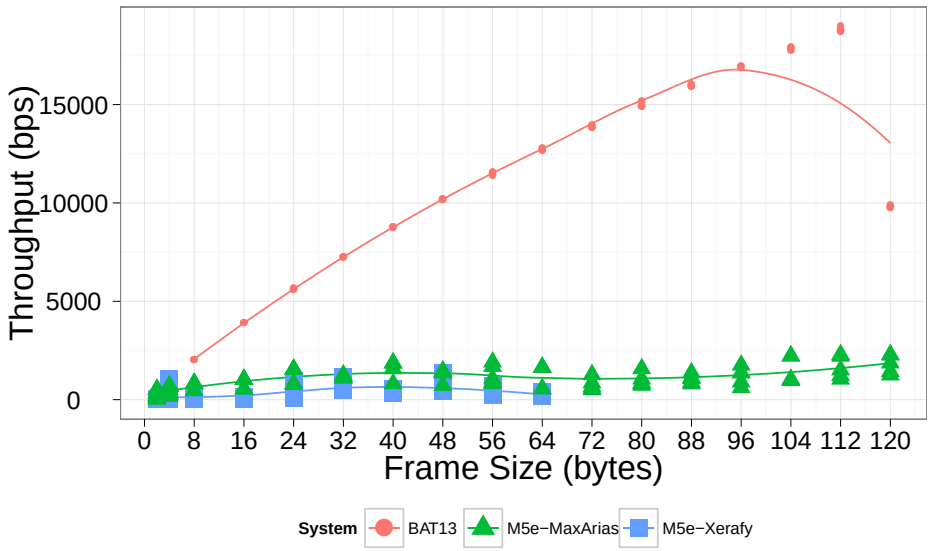

Figure 5.8. The throughput of BAT depends on the size of frames. Shorter frames result in an increased number of round trips, overhead in frame responses, and inter-frame processing. Larger frames are more likely to result in a corrupted frame. The figure shows the average throughput accounting for retransmissions using different frame sizes. The maximum throughput observed in the current prototype is 18 Kbps, achieved when using a 112-byte frame. Larger frames significantly reduce throughput due to larger error rates. BAT uses Tari = 13 μs. Frame size does not significantly affect throughput when a Gen 2 M5e reader is used with two different Gen 2 high-capacity tags because of their low success rate—the number of attempts required for a write increases proportionally with the frame size. The M5e reader uses Tari = 12.5 μs. Tags are placed approximately an inch from the antenna.

lowed by a Read command (with $\sim$ 4 bytes of data) in order to allow a tag to reply with a non-Gen 2 protocol response as illustrated in Figure 5.3.

BAT's abstractions do not add a communication overhead. Its frame-size negotiation happens only once per transmission and its overhead is approximately 50 ms, roughly the time it takes to perform a Gen 2 read. Gen 2 does not negotiate optimal frame size, but there is still some overhead associated with transmission, as illustrated in Figure 5.3. For example, Gen 2 requires tag singulation because it is meant to be used in tag-dense environments. However, BAT may or may not need this singulation. Gen 2's overhead typically involves three initiating messages before each command is

sent to a tag. In addition, Gen 2 Read and BlockWrite commands have fields that are unnecessary from the standpoint of a protocol designed to facilitate anything-to-tag communication.

The throughput of BAT will likely scale with improvements in prototypes and the physical layer. For example, one of the parameters of the physical layer, the Tari (the time interval to encode a bit in the $R \rightarrow T$ direction), significantly impacts throughput. In experiments, no errors were found when the Tari was greater or equal to 13 μs. When Tari was equal to 11 μs, only 2 of 100 128-bit frames contained no errors. In this case, errors consisted of missed bits and not bit flips. When Tari was less than 11 μs, all frames had missed bits and flipped bits. The limiting factor hindering communication with Tari values smaller than 13 μs is the MCU in the UMass Moo. The MSP430F2618 in the UMass Moo can process an interrupt no more than every 7 μs regardless of the operating clock speed, and the time required to process the input is around 5 μs. The demodulator, however, supports higher rates. In contrast, other non-CRFID tags such as the MaxArias use a 6 μs Tari.

5.5 Using Untrusted Relays

CRFIDs are well-suited for network applications that require long-term deployments because of their low maintenance requirements. In many cases it may be important to provide message secrecy and integrity for such deployments (§5.1). The ability to use untrusted relays for message routing minimizes the size of the trusted computing base and simplifies key management problems. Untrusted relays are only required to understand how to route packets in the network.

In order to support the use of untrusted relays, BAT encrypts packet payloads using AES. To ensure packet integrity, each packet includes a 32-bit message authentication code (MAC) to prevent tampering. Karlof et al. [89] provide arguments for the sufficiency of this MAC size for sensor network applications. Preliminary results

Operation	Cycles	Time (calculated)
AES-128 setup	144,677	36.1 ms
AES-128 enc (per block)	7,795	1.9 ms
AES-128 dec (per block)	8,003	2.0 ms
CMAC (AES-128 80 B)	39,669	9.9 ms

Table 5.2. The cycle count for each security operation was measured using a hardware debugger and a UMass Moo. The reported time is calculated assuming a typical sending clock speed of 4 MHz.

show that CRFID prototypes are capable of performing state-of-the-art cryptographic operations to support encrypted communication at the link layer. The most expensive operations correspond to the public-key cryptography necessary when shared keys are not distributed in advance. Once these keys have been established, providing secrecy and integrity is relatively inexpensive, as shown in Table 5.2. Opportunistic encryption allows for 61 Kbps of throughput, higher than the maximum 18 Kbps link speed achieved in experiments. Once a shared key has been constructed, tags must encrypt the packet payload using a symmetric block cipher (AES-128) and then calculate a MAC (we use CMAC [53]) over an initialization vector (IV) and ciphertext. This sort of computation is also possible because BAT allows tags to request additional power to encrypt/decrypt as needed.

5.5.1 Performance on Future CRFID Prototypes

The ratio of the computational performance of cryptographic operations over the overall throughput of BAT's communication will continue to decrease with improved microcontroller capabilities, such as power-efficiency and higher clock-speeds. This implementation was based on a current CRFID prototype with an MSP430F2618 microcontroller, the UMass Moo [148]. In the near future, we may see other CRFID prototypes with ARM microcontrollers or other MSP430 microcontrollers with higher clock speeds, which will enhance the performance of BAT's cryptographic operations.

There are MCUs in the MSP430F5xx series that would allow a CRFID to operate at 8 MHz using harvested energy or at a maximum of 25 MHz using an alternative source of energy. There are also several MCUs in the ARM Cortex-M0 and ARM Cortex-M0+ families that are even more power-efficient than MCUs in the MSP430 family, while providing higher clock speeds. For example, the ARM Cortex-M0+ may consume as little as 11.2 μW/MHz compared to 803 μW/MHz consumed by the MSP430F2618. An ARM Cortex-M0 may consume 16 μW/MHz in its lowest-power setting. Table 5.3 forecasts BAT's performance on these MCUs.

Primitive	MCU	Clock Freq.	Time (calculated)
AES Setup	MSP43F5310	8 MHz	18 ms
AES Setup	Cortex-M0+	30 MHz	1.2 ms
AES Setup	Cortex-M0	50 MHz	0.7 ms
AES Enc	MSP43F5310	8 MHz	978 μ s
AES Enc	Cortex-M0+	30 MHz	74 μ s
AES Enc	Cortex-M0	50 MHz	44 μ s
AES Dec	MSP43F5310	8 MHz	1004 μ s
AES Dec	Cortex-M0+	30 MHz	80 μ s
AES Dec	Cortex-M0	50 MHz	48 μ s

Table 5.3. Computation times for cryptographic operations on the following MCUs: (1) MSP430F5310 @ 8 MHz; (2) ARM Cortex-M0+ @ 30 MHz; (3) ARM Cortex-M0 @ 50 MHz.

5.5.2 Shared-Key Generation

Shared-key generation is the most expensive security operation required by BAT. However, micro-benchmarks show that identity-based key agreements are feasible on CRFID tags. The Smart-Chen-Kudla (SCK) key agreement [36] is less computationally expensive than a traditional Diffie-Hellman (DH) approach. However, the non-interactive Sakai, Ohgishi, and Kasahara (SOK) construction [132] is only marginally more expensive than DH.

Identity-based key-agreement schemes allow for the creation of private–public key pairs, such that the public key is an arbitrary string and only a trusted entity—a private-key generator (PKG)—can compute the corresponding private key. Thus, for example, it would be easy for a tag to encrypt a message so that only a tag with serial number x could obtain the corresponding decryption key from the PKG. Additionally, the sending tag can include an expiration time in its public key. SOK key exchange is also desirable because DH is vulnerable to man-in-the-middle attacks unless a third party authenticates protocol participants. This limitation is usually addressed by the addition of a Certificate Authority (CA), but key management is difficult in CA-based systems [58]. Table 5.4 lists the times necessary to compute a shared key using the approaches described above and the elliptic curve pairing implementations by the Miracl Crypto SDK [32].

Key Agreement	Security	Time (calculated)
Diffie-Hellman	2048-bit	208 s
Diffie-Hellman	1024-bit	50 s
Diffie-Hellman	640-bit	21 s
ECC Diffie-Hellman	$\sim$ DL 1024-bit	27 s
SOK with Type-1	$\sim$ DL 1024-bit	53 s
SOK with Type-1	$\sim$ DL 640-bit	30 s

Table 5.4. Computation times for key agreements on the MSP430F2618 @ 4 MHz. All multi-precision arithmetic, EC arithmetic, and pairings are implemented by Miracl [32]. The Diffie-Hellman key exchange based on the ECDLP uses the NIST curve P-192. The Type-1 pairings use supersingular curves $E(\mathbb{F}_p)$ for a 512-bit prime and $E(\mathbb{F}_{2^{379}})$.

It should be noted that tags must be re-keyed periodically for revocation of expired devices. In a solution without certificates, like the one discussed in this chapter, the public key of a tag may be an arbitrary string. This is one important simplification that comes from using identity-based encryption. Therefore, when a tag A wants to talk to a tag B, there is no need for tag B to request the verification of the public

key of tag A. Despite this advantage, identity based key agreements still require that a trusted authority distributes private keys. There are elegant solutions based on hierarchical identity-based encryption (HIBE) that allow for the implementation of this re-keying process in a distributed fashion [64]. The discussion of these schemes is outside of the scope of this chapter. Obtaining private keys and computing shared keys for pairwise communication is relatively expensive. A tag must be provided with power for $t_k \approx 53$ seconds to generate one of these shared keys. However, a traditional approach is not considerably less expensive. If the number of tags $|\mathbf{T}|$ is small, tags can also be instructed to precompute every pairwise key they can use to communicate with other tags.

5.6 Related Work

Networks of RFID tags [24] or CRFIDs have been of interest in the last few years, and other authors have proposed applications including building instrumentation [147] and user activity inference [25]. This past work assumes that tags are not capable of communicating with one another and that the network is actually composed of readers that report data to a central database where all computations occur. Other work has focused on enhancing supply-chain RFID-style backscattering. For example, Wang et al. [146] describe how to reduce singulation overhead by allowing multiple tags to communicate simultaneously in spite of collisions.

Reynolds has suggested the use of *semi-passive* tags capable of intercommunication, but these tags abandon backscatter in favor of traditional radio technology, so the network model is much different from that in this chapter [125]. Application-specific communication between backscattering RFID tags has been explored previously by Juels [86]. Nikitin et al. [114] propose tag-to-tag communication by allowing passive tags to listen to one another's transmissions. Their approach is restricted by requiring close proximity. BAT could be implemented atop this physical layer to improve

throughput. CCCP [133] supports tag-to-tag communication of a sort, but it is limited to storage and retrieval of data from a single tag to an untrusted reader. It does not give tags the ability to exchange data.

5.7 Conclusion

BAT allows CRFIDs to communicate with computer systems as well as other tags using relays that can be easily replaced and do not need to be aware of application information. A preliminary implementation demonstrates that BAT's abstractions are better suited for RFID-scale sensor networks than communication protocols designed for supply-chain tags because it eliminates the need for relays to manage tags' internal memory and facilitates bidirectional communication that treats the up-link and the down-link with equal priority. Micro-benchmarks show that offering a secure network layer that uses untrusted relays to exchange information does not add a significant overhead, which is attractive for applications that require tags to be left exposed and unattended.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 6

PRIVACY-PRESERVING AGGREGATION USING CONSTRAINED DEVICES

This chapter explores the problem of performing distributed privacy-preserving aggregation using constrained devices. In applications such as infrastructure monitoring, it may be beneficial to allow an untrusted entity to learn an aggregate value without revealing the individual entries that contributed to the aggregate. A traditional way of allowing an untrusted party to learn these aggregates is through a trusted aggregator that does not expose the individual entries. In some cases, such as in the event of a disaster, it may be important not to rely on a trusted aggregator.

The model of aggregation presented here, initially developed by Shi et al. [140] for large-scale systems, does not require a trusted aggregator. To achieve distributed differential privacy, a set of devices coordinates to collectively perturbate the output of an aggregation. This chapter shows that Shi et al.'s cryptographic construction for achieving distributed differential privacy is feasible using computational RFID devices by implementing it on a UMass Moo. It also discusses limitations of extensions of Shi et al.'s model designed to provide fault tolerance [33]. A Moo needs to make a 17 s computation to contribute to an aggregate. However, if fault tolerance is implemented, a Moo needs to perform a 2-minute calculation when 100 other devices are involved or a 4-minute calculation when 10,000 other devices are involved.

Contribution

This chapter demonstrates the feasibility of distributed CRFID systems for the collective computation of aggregates without relying on a trusted aggregator. This

chapter presents the implementation of Shi et al.'s [140] protocol on a CRFID prototype and describes the limitations of techniques in current CRFIDs that allow fault tolerance. The computational overhead of one such approach by Chan et al. [33] is estimated. These results provide a basis for measuring the performance of new distributed privacy-preserving systems for CRFIDs that do not require a trusted aggregator. Prior work in this area has restricted attention to implementations on full-fledged computer systems.

6.1 Privacy-Preserving Aggregation with an Oblivious Aggregator

In the example discussed earlier (§5), a bridge may be monitored by a series of devices equipped with sensors such as vibration, temperature, pressure or other sensors. While a typical approach for data collection in this example involves a centralized entity that collects data and analyzes it, in some situations, it could be desirable that non-trusted entities be able to query the system to obtain aggregate information *without* going through the centralized entity. For instance, the link to the centralized entity may not exist—e.g. in the case of a disaster area or a battle zone—or it may be useful to allow the public to verify the condition of bridges and request that a particular one be serviced because the aggregate shows that the bridge is not structurally sound. In either situation, it may be important to protect the individual entries of an aggregation. These could expose details about a vulnerability that should not be made public.

Another example that may need the kind of privacy-preserving aggregation discussed here is untrusted inventory inspections. For example, when dealing with pharmaceutical shipments, it may be important to know that no more than a certain percentage of medications has been exposed to undesirable conditions. An untrusted

inspector should not be able to gain any knowledge about the specific inventory of a shipment.

6.2 A Mechanism for Privacy-Preserving Aggregation

Let us consider the problem of computing a sum $\sum_{i=1}^{n} x_i$ over n values from n client devices while providing differential privacy. A solution to this problem was proposed by Shi et al. [140]. Their general technique is illustrated in Figure 6.1, and the assumptions for the simpler case are described below.

6.2.1 Simple Model Assumptions

Listed below are Shi et al.'s assumptions for a distributed system that enables the computation of sums using data from multiple *clients*. Each client is assumed to have a *device* that contains data that belongs to the client and that is under his or her control. Subsequently, possible ways to relax some of these assumptions are discussed.

The problem is to compute a sum $\sum_{i=1}^{n} x_i$ over n values, each the result of computing a statistical query over data that resides on each client's device. Clients are semi-honest, but they do not trust others with their data; thus, they would like to be guaranteed differential privacy. While clients may want to participate in the collaborative computation of a sum that includes their data, they do not wish to reveal their individual values. They are not expected to falsely report information or willingly avoid participation. In particular, they are assumed to always respond when queried by a designated aggregator. However, clients may collude with the aggregator by revealing their individual values. The designated aggregator is oblivious, and it is assumed to have access to auxiliary information. The aggregator adheres to the protocol and is trusted to learn the answer to the sum, but not the individual entries from each client's device. Aggregators may gain additional knowledge because they

may have other means of knowing particular individuals' values or because they have access to a related database. Clients and aggregators that participate in the protocol have previously obtained their private keys from a trusted entity. It is assumed that there is no churn. That is, there are no dynamic joins or leaves. All participants are known beforehand, and they all need to participate in order to successfully compute a sum. Finally, note that the model does not address security properties related to availability. It is assumed that there is always a communication channel between the aggregator and each of the devices.

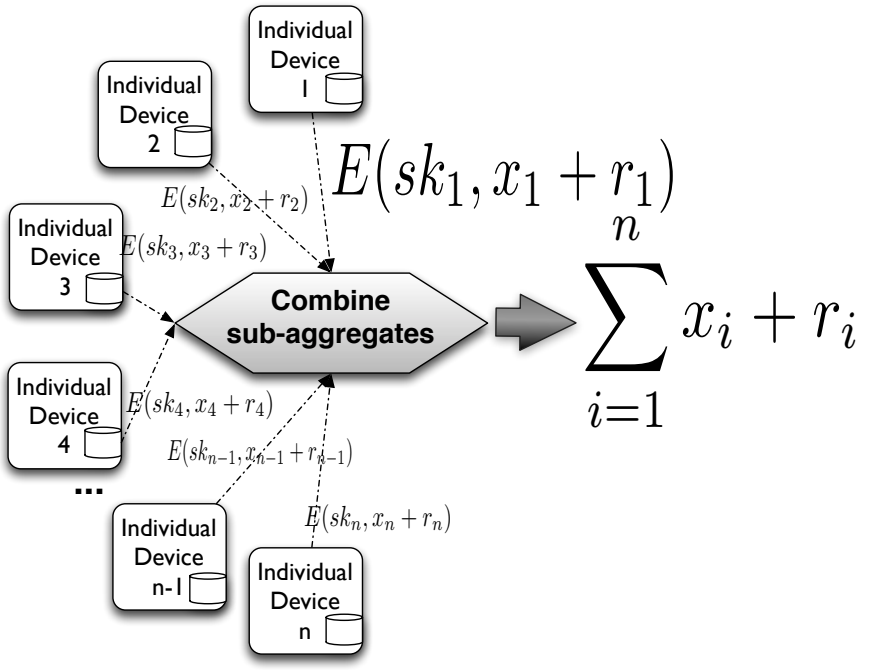

Figure 6.1. Computing a sum via a distributed model to achieve differential privacy.

6.2.2 Basic Construction

Shi et al.'s basic construction can be described as follows. The reader is referred to [140] for additional details, proofs of security, and bounds on utility. Given $\mathbb{G}$ a cyclic group of prime order p for which the Decisional Diffie-Hellman (DDH) assumption holds, and $\mathbb{H} : \mathbb{Z} \to \mathbb{G}$ a hash function modeled as a random oracle, the

computation of the aggregate takes place in three steps. First, in a setup step, a trusted dealer selects a public parameter and encryption keys for each of the participants and the aggregator. Then, each of the participants uses the public parameter and his or her key to encrypt his or her individual value previously perturbed with some noise drawn from a geometric distribution. Finally, when an aggregator receives all the encrypted pieces, he or she combines them to obtain the final aggregate without obtaining the individual contributions in the process. More explicitly the three steps correspond to the following three procedures:

1. $Setup(1^\lambda)$: The trusted dealer selects a generator $g \in_R \mathbb{G}$ as a public parameter and secret keys $sk_0, sk_1, \ldots, sk_n \in_R \mathbb{Z}_p$ such that $\sum_{i=0}^{n} sk_i = 0$. sk_0 is the secret key for the aggregator, and sk_i is the secret key for participant i.

2. $NoisyEnc(g, sk_i, t, \hat{x_i})$: At each time step t, the participant i with value x_i adds noise r_i from a geometric distribution, such that $\hat{x_i} = x_i + r_i \mod p$. Then the participant computes $c_i = g^{\hat{x_i}} \cdot \mathbb{H}(t)^{sk_i}$.

3. $AggrDec(g, sk_0, t, c_1, \ldots, c_n)$. The aggregator computes $V = \mathbb{H}(t)^{sk_0} \prod_{i=1}^{n} c_i$ and then S, the discrete log of V base g.

Note that $V = \left(\prod_{i=0}^{n} \mathbb{H}(t)^{sk_i}\right) g^{\sum_{i=1}^{n} \hat{x_i}}$ and because $\sum_{i=0}^{n} sk_i = 0$ then $V = g^{\sum_{i=1}^{n} \hat{x_i}}$ and therefore $S = \sum_{i=1}^{n} \hat{x_i}$.

To provide (ε, δ)-differential privacy, the noise r_i is chosen as follows:

$$r_i = \begin{cases} \text{Geom}(\alpha) & \text{with probability } \beta; \\ 0 & \text{with probability } 1 - \beta. \end{cases}$$

Where $\alpha = \exp(\frac{\varepsilon}{\delta})$, $\beta = \frac{1}{n} \log \frac{1}{\delta}$, and $\text{Geom}(\alpha)$ is the symmetric geometric distribution that takes integer values such that the probability mass function for the support k is $\frac{\alpha-1}{\alpha+1} \cdot \alpha^{|k|}$. Therefore, when V is decrypted, the resulting value will contain roughly one

copy of geometric noise in addition to the aggregate in order to achieve differential privacy. The proof that this scheme is computationally (ε, δ)-differentially private against compromised users can be found in Shi et al.'s paper [140]. Also note that the communication is $O(n)$, the total computation for a client is $O(1)$, and the error in the aggregate is also $O(1)$.

6.3 Implementation Using RFID-Scale Devices

The feasibility of implementing a distributed system to perform privacy-preserving aggregation of time-series of data using RFID-scale devices is explored here. The approach applies the system proposed by Shi et al. [140], which is then used by Chan et al. [33] as a building block for an alternate construction that allows for dynamic node joins and leaves.

As described above, in Shi et al.'s basic construction, each participant's device needs to compute a hash, two modular exponentiations, and a multiplication in a Diffie-Hellman group. Decrypting the aggregate using this construction requires the computation of a dilogarithm, and therefore, significantly more computational power. Thus this chapter shows that this model can be realized even when the individual devices are constrained, provided that the aggregator is sufficiently powerful.

6.3.1 Implementation Details

An implementation of the cryptographic algorithms with the UMass Moo [148] is used to evaluate the feasibility of implementing a system such as Shi et al.'s with RFID-scale devices. This CRFID prototype has an MSP430F2618 microcontroller unit with 8KB of RAM and runs at 4 MHz using harvested power and up to 16 MHz using an additional source of power, such as a solar panel or a vibration harvester.

The implementation of Shi et al.'s basic algorithms uses the Miracl [32] library for multi-precision arithmetic and elliptic curve arithmetic. The code was compiled

using the IAR Embedded Workbench for TI MSP430 [6] version 5.40.7 for measuring running times, memory footprint, and code size. Miracl was configured to use static memory allocation, i.e. functions such as *malloc* were not used in this implementation. The code is in C and does not use assembly optimizations.

For comparison, six DH groups were used: DH $\mathbb{G}_1$ is a classic Diffie-Hellman group with a 640-bit prime; DH $\mathbb{G}_2$ is a classic Diffie-Hellman group with a 1024-bit prime; and EC $\mathbb{G}_3, \mathbb{G}_4, \mathbb{G}_5, \mathbb{G}_6$ are the elliptic curve groups associated to the NIST curves P-192, P-224, P-256, P-384 respectively [23].

The implementation was also integrated in Contiki [48] to explore issues of integration with an RTOS and a micro database such as Antelope [144]. Some of the issues considered are RAM utilization and additional code size. This integration was evaluated using Cooja [115], a Java based simulator for MSP430 devices.

6.3.2 Measurements

Group	Comparable Strength	Cycles	Time (calculated) @ 4Mhz	Time (calculated) @ 16Mhz	RAM
DH $\mathbb{G}_1$	–	13,4225,440	33.5 s	8.3 s	3379 B
DH $\mathbb{G}_2$	SKIPJACK	333,209,546	83.3 s	20.8 s	4675 B
EC $\mathbb{G}_3$	SKIPJACK	67,849,396	16.9 s	4.2 s	3463 B
EC $\mathbb{G}_4$	Triple-DES	89,254,793	22.3 s	5.5 s	3571 B
EC $\mathbb{G}_5$	AES-128	115,877,960	28.9 s	7.2 s	3679 B
EC $\mathbb{G}_6$	AES-192	244,316,066	61.0 s	15.2 s	4111 B

Table 6.1. The cycle count for encrypting a noisy value ($c_i = g^{\hat{x}_i} \cdot \mathbb{H}(t)^{sk_i}$) measured using a hardware debugger and a UMass Moo. The underlying hash function was derived using SHA256. As a reference, the second column lists a block cipher with comparable strength based on the key size [112, 23]. The reported time is calculated assuming a clock speed of 4 MHz, typical when using harvested energy, and 16 MHz, the maximum when an additional source of power is available. The RAM requirements do not include the networking stack or any other additional application logic and are measured using IAR's IDE and compiler tools V.5.40.7.

Table 6.1 shows the time it would take a CRFID tag to produce an encrypted noisy value (§6). As we can see, it is possible to encrypt a noisy value on a Moo in under 17 s using an acceptable level of security. For higher levels of security, using classic Diffie-Hellman groups has a prohibitive cost, not only computationally, but the memory requirements are greater than those of the current UMass Moo prototype. An implementation for an MSP430 using a classic Diffie-Hellman group with a 2048-bit prime requires 13 KB of RAM, more than the 8 KB of RAM of the current UMass Moo prototype. This total amount of RAM has to be shared with the networking stack and other application logic.

6.3.3 Additional System Considerations

It is important to note that in addition to being able to generate a noisy encrypted version of a local query result, a device should be able to store its data locally on a system that resembles a relational database. One example of such a system for constrained devices is Antelope [144]. This system allows for the computation of queries such as:

```
>> SELECT MEAN(humidity), MAX(humidity) FROM samples
WHERE year = 2010 AND month >= 6 AND month <= 8;
```

or

```
>> contacts <- JOIN device, rendezvous ON device_id
PROJECT address, year, mon;
SELECT COUNT(*) FROM contacts WHERE year = 2011 AND mon = 4
AND address = aaaa::1;
```

It is also necessary to allow some basic multitasking to coordinate communication, computation, sensing, and possibly actuating. Over the years multiple RTOSes have been developed for sensor nodes, such as TinyOS [98] or Contiki [48]. Currently,

there is no operating system with comparable functionality for RFID-scale devices. Instead multitasking is implemented using finite state machines. Switching between states happens using timers and interrupts. One reason contributing to the non-existence of real-time operating systems for RFID-scale devices may be their highly erratic power cycles. An RTOS suitable for these devices would have to have faster suspend and resume functionality than RTOSes for battery-powered systems, which for the most part assume longer power cycles.

With this in mind, it is important to ensure that the code sizes of an RTOS, database system, and cryptographic implementation do not exceed the total size of ROM available. The implementation of Shi et al.'s generation of noisy encryptions (§6.3.1) was integrated with Contiki and Antelope to illustrate that the total integration of the system may fit within the 116 KB available for code in an MSP430F2618. The cryptographic components require approximately 31 KB depending on the DH group selected, and Contiki and Antelope require 4 KB and 17 KB respectively in their minimum configurations. Finally, there needs to be sufficient code space to implement networking and drivers for additional storage (i.e. external flash). In the case of this prototype, these require under 4 KB; however, other networking stacks such as ip or ipv6 may have significantly higher code size requirements.

In practice, a greater challenge is to accommodate RAM requirements. Because the implementation of the cryptographic components requires approximately 4KB of RAM and a database system such as Antelope requires just over 4KB of RAM, there is barely enough room to fit both of these components in the 8 KB of total RAM available. To ameliorate this issue, a custom implementation of the crypto components was developed from scratch using EC $\mathbb{G}_3$ and only including the arithmetic operations needed to perform arithmetic on the NIST curve P-192 [23]. This implementation requires approximately 11 KB of code size and approximately 2 KB of RAM.

6.4 Fault Tolerance

Shi et al.'s basic approach suffers from the requirement that each participant must take part in the aggregation. It is easy to see that if a single encrypted entry $c_i = g^{\hat{x_i}} \cdot \mathbb{H}(t)^{sk_i}$ is missing, then $V' = \left(\prod_{i \in S} \mathbb{H}(t)^{sk_i}\right) g^{\sum_{i \in S} \hat{x_i}}$ will not decrypt to the sum $\sum_{i \in S} \hat{x_i}$ of all the remaining values.

Chan et al. [33] built on Shi et al.'s basic approach to create a solution that provides fault tolerance and dynamic joins and leaves by using binary trees. While this solution may be appealing when using traditional computer systems such as workstations or servers, it may not always be practical on constrained devices. The main reason is that a client in this system has to perform $O(\log n)$ encryptions rather than a single encryption. For example, a device needs to perform 14 encryptions when the number of devices is approximately 10,000 or 7 encryptions when the number of devices is approximately 100. Thus, in practice, computing a noisy aggregate using constrained devices such as CRFIDs will require minutes rather than seconds. In some applications, however, this penalty may be acceptable.

6.4.1 Utility

Shi et al.'s approach produces an answer with an error of size $O(1)$. However, Chan et al.'s produces an error[1] of size $\tilde{O}((\log n)^{3/2})$ in order to deal with missing participants [33]. These bounds do not take into account the issue of requiring privacy budgets for multiple related queries.

6.5 Related Work

Related work in networking and distributed systems has aimed to improve the performance of aggregations. For example, Chen et al [37] propose a system for achieving

[1]The authors use the simpler $\tilde{O}(\cdot)$ notation to hide a $\log \log n$ factor and other factors that depend on the number of failed users and the differential privacy factors.

distributed differential privacy using the Goldwasser-Micali bit-cryptosystem [67]. In such a system, encrypting a bit requires a squaring and a multiplication. When the local answer to a query is small, this encryption system is significantly more efficient. A drawback of their approach is that they assume the existence of a proxy that is honest but curious between the analysts and the participants. This proxy controls which clients participate in a given aggregation. Chen et al. discuss a way to deal with malicious proxies through the use of trusted hardware, such as Trusted Platform Modules (TPMs).

Another system that guarantees differential privacy using a distributed model of a sort is GUPT [104]. However, Mohan et al.'s distributed model has the goal of achieving parallelism to improve performance without compromising the utility guarantees. This parallelism also allows for the transparent division into computation blocks that can be independently calculated. The purpose of this is to enhance practicality even when programs are not written with privacy in mind. In GUPT the aggregator is trusted and the private output is the result of computing an answer to an aggregate and adding Laplacian noise afterwards.

McSherry [103] developed PINQ, a system that provides a programing interface to perform queries on a database obtaining responses that provide differential privacy. McSherry shows how to deal with the issue of composing multiple queries and how this composition affects both utility and privacy. As a result, he develops a *privacy calculus* such that an *agent* sets and verifies the proper differential privacy parameters for each analyst according to a policy and the various queries made by that agent.

6.6 Conclusion

This chapter shows that techniques to achieve differential privacy can be implemented using CRFIDs, opening the door to a variety of privacy-aware applications. However, there are important issues that need to be addressed to improve the prac-

ticality of large deployments. The issue of allowing fault tolerance may require a considerable number of additional expensive encryptions, which may cross the line of acceptable performance. Other system aspects that require further development include the design and implementation of supporting system components such as a real-time operating system for RFID-scale devices. From the theoretical standpoint, it is important to develop more practical models that address malicious clients, privacy budgets, and high utility guarantees without relying on trusted aggregators.

CHAPTER 7

PRIVACY-PRESERVING AGGREGATION OF MEDICAL TELEMETRY

Medical devices are increasingly collecting telemetry to monitor patients' well-being as well as device functioning. The collection and storage of this telemetry are often performed by device manufacturers, who then make aggregate information available to caregivers. The information gathered from this telemetry could also be used to study the effectiveness of devices and their reliability. However, a challenge lies in being able to compute statistical analyses on telemetry from multiple manufacturers and institutions such that the privacy of individual patients is protected.

An alternative approach is one in which caregivers collect and store telemetry from medical devices and participate in multi-institutional protocols to perform statistical analyses [105].[1] Homomorphic encryption schemes can be used to facilitate the calculation of statistical functions while preventing the disclosure of individual entries, thus allowing for the computation of analyses using data from multiple institutions and manufacturers.

The feasibility of this approach is judged on the basis of *how well* an analysis is performed rather than *how fast* an answer is computed. The rational is that the computation of a privacy-preserving counting query in a distributed fashion can be done sufficiently fast when performing an analysis over a period of several weeks.

---

[1] This chapter draws from previously published work: "HICCUPS: Health Information Collaborative Collection Using Privacy and Security" by A. Molina, M. Salajegheh, and K. Fu. In Proceedings of the ACM Workshop on Security and Privacy in Medical and Home-Care Systems. November 2009.

This chapter shows that a number of useful statistical analyses may be performed via this approach, including counting queries, averages, sums, maxima, minima and linear regressions; which allow for the evaluation of the effectiveness and reliability of medical devices.

Contributions

The contributions of this chapter are:

Model. This chapter proposes an alternative model that does not rely on device manufacturers as mediators between the collection of telemetry and analysts. In the resulting trusted computing base, only caregivers have full access to patients' data. Because patients in a single institution may have devices from multiple manufacturers, institutions will be in the position to perform analyses across manufacturers. Multiple institutions could then collaborate in the computation of analyses.

Design. This chapter designs Health Information Collaborative Collection Using Privacy and Security (HICCUPS), a system that could allow for the collaborative computation of aggregates across institutions via homomorphic encryption.

Definition. This chapter defines a metric for evaluating a system that computes statistical aggregates while preserving patients' privacy. A suitable metric for a system should be based on the utility of analyses rather than the computational overhead required to protect privacy, provided that a computation is done sufficiently fast.

7.1 Case for a Distributed Model for Aggregating Medical Telemetry

Preserving the privacy of aggregated medical data currently focuses on manual removal of personal information or rigid processing of data by highly trusted information brokers. While such systems can work well on a small scale, drawbacks include the ad-hoc nature of manual redactions and the placing of power in the hands of a

single entity that could be compromised. An alternative approach to secure aggregation could instead protect the privacy of individuals using a distributed model. Aggregate information could be used to identify trends and system-wide causes of disease, procedural mistakes, or device malfunction.

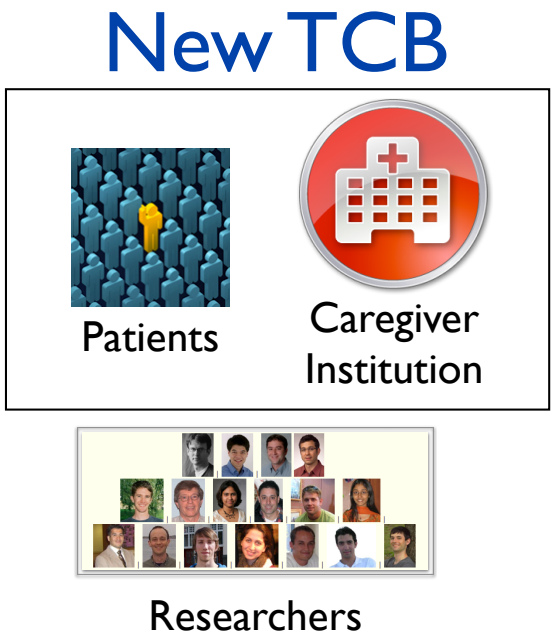

Figure 7.1. An alternative trust model would place researchers—including device manufacturers—in a position where they can submit privacy preserving queries. Only patients and their caregivers have full access to patients' telemetry.

Medical telemetry generated by home monitoring is an example of an instance in which patient privacy could be at risk due to current practices that do not limit the amount of data given to device manufacturers. The trend toward remote monitoring allows patients with implanted devices to ensure proper functioning without the inconvenience of having to travel frequently to a clinical office. The current mechanism employed to make this remote monitoring possible involves sending all of the medical telemetry *directly to the manufacturers*, which then make it available to a patient's caregiver remotely (Figure 7.4). As of 2012, patients with medical implants such as implantable cardio-defibrillators and pacemakers do not have access to the informa-

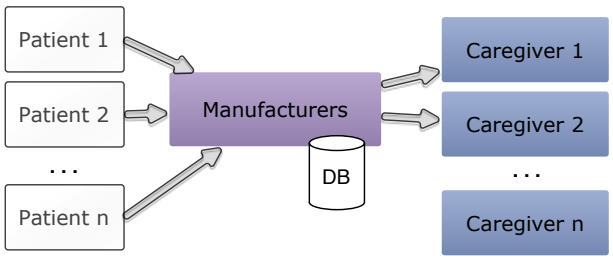

Figure 7.2. Data flow that prevails currently. Patients upload their telemetry to the manufacturer's servers, from which it is made available to caregivers for analysis.

tion that is collected from their devices and manufacturers treat these data as their own [29]. There is also no formal infrastructure to allow clinical researchers to gain access to key pieces of data of a statistical nature.

Solutions such as making it easier for researchers to access de-identified data, as proposed by Gostin and Nass [113], may still pose unanticipated problems. For example, Biel et al. [19] show that it is possible to identify an individual in a predetermined group by using ECG data. Moreover, centralized locations that even briefly have access to unencrypted data create single points of failure where entire national databases could be compromised by clever hackers or conspiring insiders. As an alternative, this chapter proposes that both device manufacturers and clinical researchers be able to obtain the information that they need about the data via computing aggregate functions on encrypted data.

Medical Implant Reliability Studies

In 2006, cardiologist Dr. William Maisel [100] published a meta analysis of pacemaker and ICD registries to assess the rates of pacemaker and ICD malfunctions in order to be able to identify trends in the reliability of these devices. Another similar study also aimed at counting the malfunctions of pacemakers and ICDs examined data from multiple years included in the annual reports of the Federal Drug Administration of the U.S. [101]. The authors of this study pointed out that the database

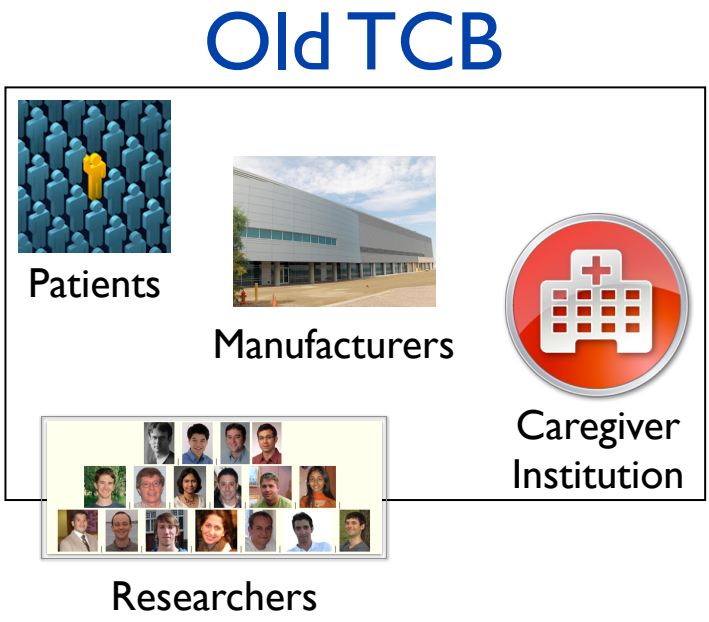

Figure 7.3. The current trust model requires that a manufacturer gather all patient data and deliver it to their corresponding caregivers. Patients do not currently have access to their data.

registries that monitor the performance of these devices are limited primarily by their small size or by their voluntary nature. Both of the studies cited above concluded that ongoing surveillance of pacemaker and ICD performance should be required. A system like the one described in this chapter could allow institutions such as the FDA to conduct reports that provide a more detailed aggregation of device malfunctions including device model numbers and types of malfunctions. The data-sets made available using this model could be larger, and the aggregation could potentially be computed more frequently and in an automated way. If manufacturers published their parameters for identifying a malfunctioning device without extracting them from the patient, then not only the caregiver, but also the FDA, would be able to react to reliability or safety issues.

Identifying the Impact of Low Income on Preterm Birth Risk

Developments in the implementation of distributed methods for analysis may go beyond medical devices. In 2009 the Center for Democracy and Technology in the U.S. published a document to encourage the use of de-identified and anonymized health data and to rethink the protection of this data by regulations such as the HIPAA Privacy Rule [127]. This document notes that one of the common uses for this kind of data is research. In particular, the document mentions as an example that de-identified data has been used to perform research on the prevention of premature births. One such research study, performed by DeFranco et al. studied the effect of living in a socioeconomically deprived area on the risk of preterm births [41]. In this study, a number of counties in Missouri were identified as being below the U.S. poverty line based on census information. These counties were then classified according to various levels of poverty. The number of pregnancies that resulted in various periods of preterm birth were counted using de-identified records, and the aggregates were analyzed. The study concluded that above other underlying risk factors, women residing in socioeconomically deprived areas are at an increased risk of having a preterm birth. A distributed system for aggregation could help in the validation or extension of DeFranco et al.'s analysis to regions all over the nation. Homomorphic encryption could allow researchers to aggregate device malfunctions or calculate the number of preterm births for women living below the U.S. poverty line in various counties. These aggregates could also be computed for different preterm birth periods and then analyzed to determine the impact of low income conditions.

7.2 HICCUPS

This section presents Health Information Collaborative Collection Using Privacy and Security (HICCUPS), a distributed system that uses homomorphic encryption to allow only caregivers to have unrestricted access to patients' records and at the same

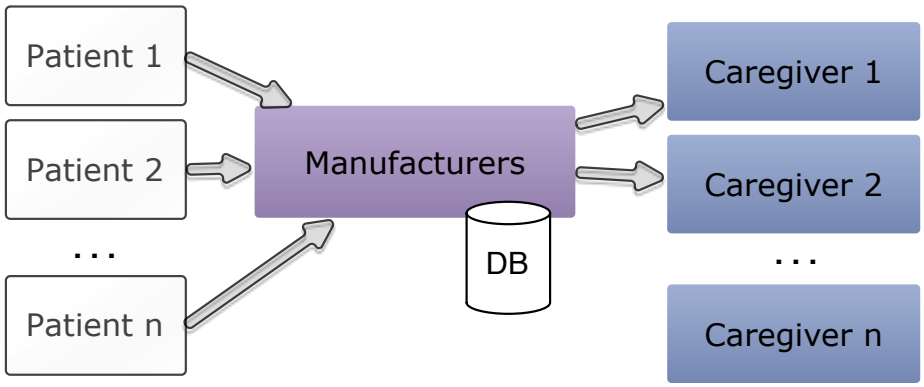

Figure 7.4. The figure illustrates the data flow that prevails currently. Patients upload their telemetry to the manufacturer's servers, from which it is made available to caregivers for analysis.

time enable researchers to compute statistical values and aggregate functions across different patients and caregivers.

7.2.1 Background on Homomorphic Encryption

Modern cryptography allows for computation on *encrypted* values through a technique developed in the 1970s known as homomorphic encryption [128], which became popular in the 1990s for secure tallying of encrypted votes [117, 63]. Algorithms from number theory allow arithmetic on encrypted data.

Homomorphic encryption allows an information aggregator not to be trusted. That is, the aggregator could not easily violate individual patient privacy in a mathematically provable sense. The scheme prevents rogue insiders from violating privacy and prevents accidental leakage of private information. The tradeoff is that homomorphic encryption requires sophisticated computation on a modern computer, which is feasible on commodity hardware for workloads common to medical telemetry.

The potential of computing on encrypted data has promising theoretical results, especially after the recent findings of Gentry [63] that prove fully homomorphic encryption schemes can be implemented using lattices. This discovery suggests that the

research community may be close to extending the capabilities of this technique to essentially allow arbitrary computations on encrypted data.

7.2.2 Access Model

The assumption is that direct caregivers need access to all of a patient's medical data in order to perform proper treatment. Under this assumption, there is an unrestricted data flow between patients and their direct caregivers. In such a model, multiparty computation techniques can be applied to allow distinct caregivers to compute collective answers to queries posed by clinical researchers and manufacturers (Figure 7.5).

For simplicity, it is also assumed that a patient does not have multiple caregivers in this model. Multiple associations could lead to false aggregates due to duplication. However, such a situation can be addressed by requesting that each patient has only one caregiver that reports data on his behalf.

The queries required by researchers and manufacturers can be classified into the following types:

1. **Selective individual disclosure: Queries to obtain a list of records.** This type of query would present the problem of returning a set of records that match a set of criteria from data distributed across the set of caregivers $\{\mathcal{C}_1, \mathcal{C}_2, \ldots \mathcal{C}_n\}$. The result of such a query can be seen as a matrix, the rows of which are the union of the rows of the sub-matrices that each of the caregivers returns to the query posted. For example, a query could request the age, gender and number of critical events in a given month for all the patients that have an ICD and that use a home monitor nationwide. Then each caregiver C_i would return a matrix (query table) with three columns and as many rows as patients are being treated by C_i. The final result to the query should be a union of all these records. The result can be thought of as a matrix with the same three

columns and with as many rows as the total number of relevant records found nationwide.

2. **Queries to obtain aggregated disclosures.** This type of query would present the problem of computing an aggregate function on data distributed across the set of caregivers $\{\mathcal{C}_1, \mathcal{C}_2, \ldots \mathcal{C}_k\}$ while preserving the privacy of the data between any two different caregivers.

The main focus of this chapter is to design a system using homomorphic encryption to address the queries of the second type (aggregation), which are posed when clinical researchers and device manufacturers need to compute aggregates and other similar statistical information. The objectives for obtaining data may differ between clinical researchers and device manufacturers, and it is important to note that this model requires that both researchers and manufacturers specify clearly and openly the kind of information that they require.

7.2.3 Threat Model

The goal of *HICCUPS* is to prevent unnecessary disclosure of large collections of medical telemetry. It is assumed that locations with transient access to plaintext records (e.g., caregivers) are relatively small collections such that a compromise will result in only localized disclosure of information rather than system-wide disclosure. The threat model for *HICCUPS* considers both external and internal adversaries. For instance, an external hacker who gains unauthorized access to a machine should cause at worst a localized disclosure of patient information. An external entity should not be able to cause a catastrophic disclosure of the entire collection. Potential insiders include medical researchers and aggregators. When these players follow the established protocol, it is expected that they will have access to aggregated information. However, if the players act maliciously they should not be able to compromise the entire system, but at worst delay the availability of information. In this model,

caregivers are fully trusted by their corresponding patients. That is, caregivers are not considered potential insiders but are potential targets of external adversaries.

7.2.4 Desired Properties

In order to answer queries for aggregated information, it is necessary to solve the problem of computing aggregates from encrypted values using the public key $\mathcal{R}_p$ of a researcher $\mathcal{R}$ by caregivers $\{\mathcal{C}_1, \mathcal{C}_2, \ldots \mathcal{C}_k\}$. Desirable properties for such a solution include:

1. **Anonymity of Data Provider**. Given a set of ciphertexts

$$\{\mathrm{Enc}_{R_p}(a_1), \mathrm{Enc}_{R_p}(a_2), \ldots, \mathrm{Enc}_{R_p}(a_k)\},$$

   provided by a set of entities $\{\mathcal{C}_j\}$, the probability of determining that $\mathrm{Enc}_{R_p}(a_i)$, for a given i, was computed by $\mathcal{C}_j$ for some j should differ by a negligible quantity from guessing this association.

2. **Distributivity**. It is important to emphasize that a proposed solution should be implemented using a distributed model as opposed to a centralized model. A centralized approach would give too much power to the holder and would create a single point of failure. As discussed by Jefferson et al. [83] centralized systems introduce several privacy risks in voting systems [84] for example.

3. **Semantic security**. A final implementation of *HICCUPS* should be done using an encryption system that is semantically secure to avoid chosen plaintext attacks (§7.5).

7.2.5 Design

This section shows how having an encryption scheme with the homomorphic property may provide an answer to the problem of computing aggregate functions with

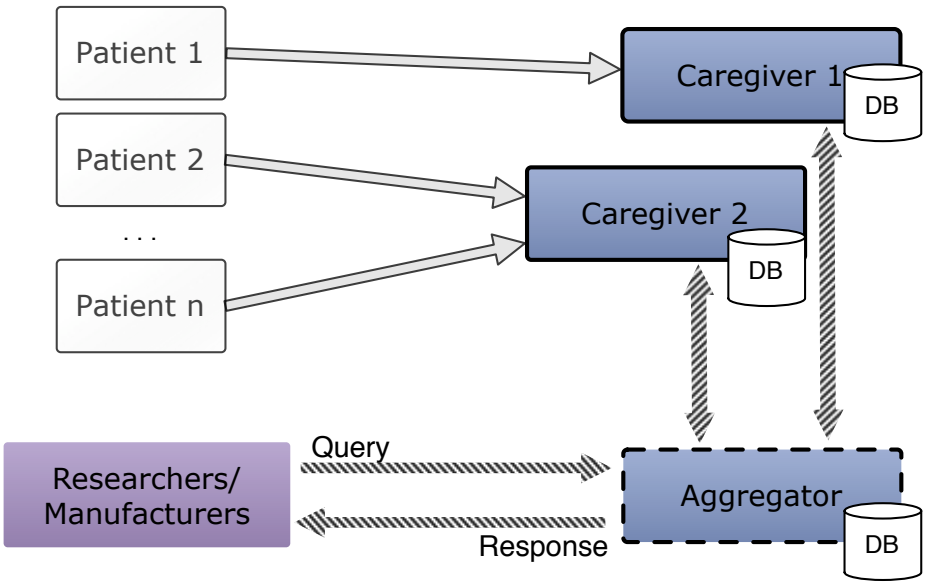

Figure 7.5. A researcher or manufacturer needs to compute an aggregate function from data across various caregivers. The query can be handled by an aggregator chosen among the caregivers that computes on encrypted data.

data that is distributed among various caregivers. Sample aggregation functions that can be computed using the homomorphic property are presented (§7.3). The number of functions that can be computed using this technique is dramatically extended if a fully homomorphic encryption is used.

Unless stated otherwise, the existence of a public key infrastructure with a semantically secure homomorphic encryption scheme with key generation algorithm, encryption algorithm, and decryption algorithm (Gen, Enc, Dec respectively) is assumed.

The model proposed here is one in which patients give all of their medical data to their caregivers. In this model, various patients' data clusters around caregivers. The tasks of computing aggregates across caregivers in order to answer the questions of manufacturers and researchers could then be thought of as a multiparty computation (Figure 7.6).

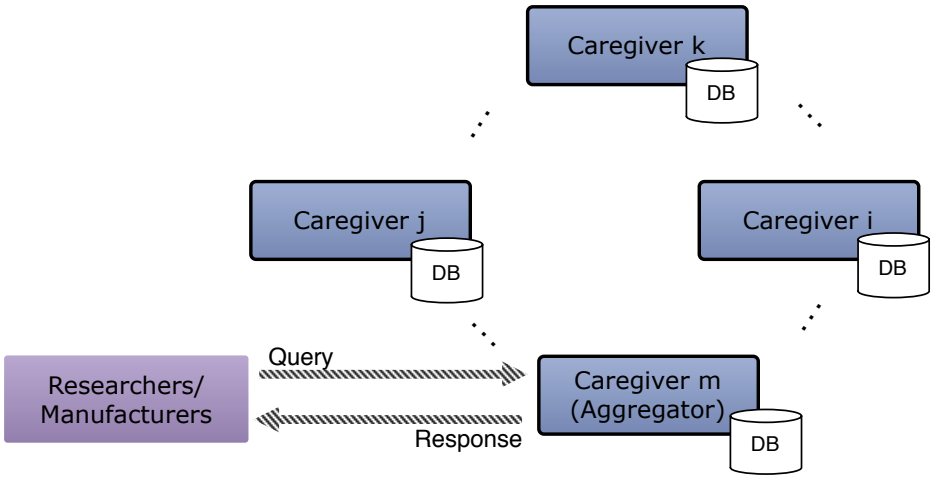

Figure 7.6. The aggregator is chosen at random to eliminate the probability of a compromised aggregator systematically leaking data. The rest of the caregivers compute sub-aggregates, which can be combined by the aggregator to produce a total aggregate for the manufacturers and researchers.

Consider the problem of computing a publicly known aggregate function f for a variable x with data distributed among a set of caregivers $\{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_k\}$ from sub-aggregates $a_1, a_2, ..., a_n$, pre-computed by the caregivers $\mathcal{C}_i$. That is, $f(a_1, a_2, ..., a_n)$ is the aggregate value needed for the publicly known function f. Aggregates can be combined to compute functions such as sample mean, sample variance, maxima, linear regression, and sample correlation (§7.3).

A global aggregate could be computed as follows:

1. **Request for an aggregate**. A researcher $\mathcal{R}$ interested in computing a global aggregate submits the request specifying one of the possible aggregate functions f. For example, the function could be a simple aggregate sum $f(x_{ij}) = \sum_{i,j} x_{ij}$, over a defined set of values x_{ij} distributed among all the set of participant caregivers $\{\mathcal{C}_i\}$.

2. **Selection of an aggregator**. All the caregivers run a distributed algorithm to determine a random aggregator within the set of participant caregivers $\{\mathcal{C}_i\}$.

At the end of this step, one of the participant caregivers is designated the aggregator for the request. This aggregator is denoted $\mathcal{A}$. Note that having a fixed aggregator could lead to an attack in which the privacy guarantees could be reduced to essentially those of handing over the sub-aggregates per caregiver directly to the researcher $\mathcal{R}$. This may be undesirable, for example, if a query is asking for the age of the oldest patient in a group distributed nationwide. By exposing sub-aggregates, the researcher would obtain not only the age, but also potentially the name of the patient's caregiver. The selection of an aggregator randomly can be done using distributed techniques similar to those proposed by Kapron et al. [88].

3. **Computation of sub-aggregates**. Each caregiver $\mathcal{C}_i$ receives the request and computes its corresponding sub-aggregate a_i and encrypts it first using the public key of the researcher $\mathcal{R}_p$, and then using the public key of the aggregator $\mathcal{A}_p$. The caregiver $\mathcal{C}_i$ can then use a bulletin board style protocol to share the encrypted result $\text{Enc}_{\mathcal{A}_p}(\text{Enc}_{\mathcal{R}_p}(a_i))$. Coming back to the example, the caregivers would create the a_is such that $\sum_i a_i = \sum_{i,j} x_{ij}$, but would not send the a_is unencrypted.

4. **Unwrapping**. The caregiver $\mathcal{A}$ chosen to compute the aggregate obtains the encrypted values $\{\text{Enc}_{\mathcal{R}_p}(a_i)\}$ by decrypting the first layer of $\text{Enc}_{\mathcal{A}_p}(\text{Enc}_{\mathcal{R}_p}(a_i))$ for each i. This is needed to ensure that only the designed aggregator $\mathcal{A}$ is able to compute the aggregate. A complimentary technique can be used to ensure participation; for example, the outer wrap can be signed by the corresponding caregiver.

5. **Aggregation**. The caregiver $\mathcal{A}$ computes $f^*(\text{Enc}_{\mathcal{R}_p}(a_i))$, where f^* can be obtained from f using the homomorphic property. Subsequently $\mathcal{A}$ destroys each individual $\text{Enc}_{\mathcal{R}_p}(a_i)$. Implicitly, this as-

sumes that the majority of the caregivers are honest. If that were not the case, and the majority of the caregivers were willing to forward the individual $\text{Enc}_{\mathcal{R}_p}(a_i)$, then the probability of falling victim to an attack like the one in the case of having a fixed aggregator could not be considered negligible. Also note that the aggregator was not in the position to learn anything about the sub-aggregates from the other caregivers since the sub-aggregates were encrypted using the researcher's public key. In the example f^* corresponds to adding encrypted aggregates with the operation $\oplus$, while f corresponds to the addition of plaintexts, thus $f^*(\text{Enc}_{\mathcal{R}_p}(a_i)) = \oplus_i \text{Enc}_{\mathcal{R}_p}(a_i)$.

6. **Handing over**. The aggregator $\mathcal{A}$ returns the encrypted aggregate value $\text{Enc}_{\mathcal{R}_p}(f(a_i))$ to the researcher $\mathcal{R}$, which can be decrypted using its corresponding secret key $\mathcal{R}_s$.

Note that in an alternative approach this aggregation could be done by encrypting each of the values $\{\text{Enc}_{\mathcal{R}_p}(x_{i,j})\}$ distributed among caregivers $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_k$, where $x_{i,j}$ is a value known to $\mathcal{C}_i$ and j ranges from $1, \ldots, n_i$ the sample size in $\mathcal{C}_i$. However, in the case of computing aggregates for medical data nationwide this could potentially involve transmitting millions of values to the aggregator $\mathcal{A}$, instead of sending only one sub-aggregate a_i per caregiver. Thus, the proposed approach could potentially avoid a large unnecessary overhead.

7.3 Computing Aggregates through Counting Queries

Multiple aggregates can be combined to calculate common statistical functions, although there are some challenges in generalizing the examples below to complete solutions for privacy-preserving aggregation (§7.5). The ability to extend the techniques described here will depend on whether a singly homomorphic encryption scheme or a fully homomorphic scheme is used. That is, whether one or two operations (addi-

tion and/or multiplication) on ciphertexts with the homomorphic property may be performed.

Consider a homomorphic encryption scheme that is IND-CPA secure with key generation algorithm, encryption algorithm, and decryption algorithm (Gen, Enc, Dec respectively) and with semigroups of plaintexts and ciphertexts $\mathbf{M}$ and $\mathbf{C}$ respectively. That is, the homomorphic property is such that for $m_1, m_2 \in \mathbf{M}$ and a pair of public and secrete keys $\mathcal{A}_p, \mathcal{A}_s$

$$\mathrm{Enc}_{\mathcal{A}_p}(m_1 + m_2) =_p \mathrm{Enc}_{\mathcal{A}_p}(m_1) \oplus \mathrm{Enc}_{\mathcal{A}_p}(m_2)$$

where $\oplus$ is the corresponding operation to $+$ on $\mathbf{C}$ and $=_p$ denotes indistinguishability of distributions. An adversary would not be able to tell that $\mathrm{Enc}_{\mathcal{A}_p}(m_1 + m_2)$ and $\mathrm{Enc}_{\mathcal{A}_p}(m_1) \oplus \mathrm{Enc}_{\mathcal{A}_p}(m_2)$ correspond to the same plaintext. However

$$\mathrm{Dec}_{\mathcal{A}_s}(\mathrm{Enc}_{\mathcal{A}_p}(m_1 + m_2)) = \mathrm{Dec}_{\mathcal{A}_s}(\mathrm{Enc}_{\mathcal{A}_p}(m_1) \oplus \mathrm{Enc}_{\mathcal{A}_p}(m_2)).$$

A randomly selected caregiver computes an aggregate such as the sample mean and variance of a sample of values $\{x_{i,j}\}$ distributed among caregivers $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_k$, where $x_{i,j}$ is a value known to $\mathcal{C}_i$ and j ranges from $1, \ldots, n_i$, the sample size in $\mathcal{C}_i$. This procedure does not require the actual knowledge of the values $\{x_{i,j}\}$. Instead the knowledge of appropriate encrypted sub-aggregate values suffices.

In this case, the mean of $x_{i,j}$ over all $\mathcal{C}_i$s is given by

$$\bar{x} = \frac{\sum_{i=1}^{k} \sum_{j=1}^{n_i} x_{i,j}}{\sum_{i=1}^{k} n_i}$$

And the variance of the sample is given by

$$s^2 = \frac{\sum_{i=1}^{k} \sum_{j=1}^{n_i} (x_{i,j})^2 - \frac{(\sum_{i=1}^{k} \sum_{j=1}^{n_i} x_{i,j})^2}{\sum_{i=1}^{k} n_i}}{(\sum_{i=1}^{k} n_i) - 1}$$

Now, denote $a_i = \sum_{j=1}^{n_i} x_{i,j}$, and

$b_i = \sum_{j=1}^{n_i} (x_{i,j})^2$ then the formulas can be rewritten as:

$$\bar{x} = \frac{\sum_{i=1}^{k} a_i}{\sum_{i=1}^{k} n_i}$$

and

$$s^2 = \frac{\sum_{i=1}^{k} b_i - \frac{(\sum_{i=1}^{k} a_i)^2}{\sum_{i=1}^{k} n_i}}{(\sum_{i=1}^{k} n_i) - 1}$$

Thus, given the additive homomorphic scheme, researcher $\mathcal{R}$ can compute these aggregates preserving privacy as follows: In order to compute aggregates such as the mean and the variance, the aggregator $\mathcal{A}$ first collects the encrypted values of a_i, b_i, n_i (using the public key, $\mathcal{R}_p$ of the researcher $\mathcal{R}$) from the caregivers. More explicitly, if each of the $\mathcal{C}_i$s provides $\text{Enc}_{\mathcal{R}_p}(a_i)$, $\text{Enc}_{\mathcal{R}_p}(b_i)$, and $\text{Enc}_{\mathcal{R}_p}(n_i)$, then the aggregator $\mathcal{A}$ computes $a = \bigoplus_{i=1}^{k} \text{Enc}_{\mathcal{R}_p}(a_i)$, $b = \bigoplus_{i=1}^{k} \text{Enc}_{\mathcal{R}_p}(b_i)$, and $n = \bigoplus_{i=1}^{k} \text{Enc}_{\mathcal{R}_p}(n_i)$ and sends these three values to researcher $\mathcal{R}$. Finally, the researcher $\mathcal{R}$, using the corresponding secret key $\mathcal{R}_s$, would simply compute:

$$\bar{x} = \frac{\text{Dec}_{\mathcal{R}_s}(a)}{\text{Dec}_{\mathcal{R}_s}(n)}$$

since

$$\bigoplus_{i=1}^{k} \text{Enc}_{\mathcal{R}_p}(a_i) =_p \text{Enc}_{\mathcal{R}_p}(\sum_{i=1}^{k} a_i).$$

Similarly,

$$s^2 = \frac{\text{Dec}_{\mathcal{R}_s}(b) - \frac{\text{Dec}_{\mathcal{R}_s}(a)}{\text{Dec}_{\mathcal{R}_s}(n)}}{\text{Dec}_{\mathcal{R}_s}(n) - 1}.$$

These ideas can be extended to compute a linear regression $y = \beta_0 + \beta_1 x$ and a sample correlation r_{xy}. The problem of estimating β_0, and β_1 can be obtained from the sums $\sum x, \sum x^2, \sum y, \sum y^2$, and $\sum xy$ as follows:

$$\beta_1 = \frac{\sum y \cdot \sum x - n \cdot \sum xy}{(\sum x)^2 - n \cdot (\sum x^2)}$$

and

$$\beta_0 = \frac{\sum x \cdot \sum xy - \sum y \sum x^2}{(\sum x)^2 - n \cdot x^2}$$

and

$$r_{xy} = \frac{n \cdot \sum xy - \sum x \sum y}{\sqrt{n \cdot \sum x^2 - (\sum x)^2}\sqrt{n \cdot \sum y^2 - (\sum y)^2}}.$$

It may also be useful to compute maxima or minima. The following shows how to compute maxima; the computation of minima is completely analogous. While this described method is not optimal, it serves to illustrate the feasibility of computing such functions using aggregates.

In order to compute a global maximum for a variable among a set of caregivers $\{C_i\}$, the problem must first be redefined in a convenient way. In particular, it will be necessary to have an idea of the range $(r_{\min}, r_{\max})$ and precision d for these values. For example, if it were required to find the maximum temperature for all the patients meeting certain conditions, one could specify the range to be between 35 and 48 degrees Celsius. Precision may need to be obtained up to one decimal place.

Given these two values, one can define a vector with l entries where l is equal to the number of intervals of size d in the interval $(r_{\max}, r_{\min})$, or simply $l = \frac{r_{\max} - r_{\min}}{d}$ if $r_{\min}, r_{\max} \in \mathbb{Z}$. Then each of the entities in $\{\mathcal{C}_i\}$ can return a vector

$$v_i = (\mathrm{Enc}_{\mathcal{R}_p}(c_0), \ldots, \mathrm{Enc}_{\mathcal{R}_p}(c_j), \ldots, \mathrm{Enc}_{\mathcal{R}_p}(c_{l-1}))$$

where $c_j = 1$, if $r_{\min} + c_j$ is the maximum value within the caregiver $\mathcal{C}_i$'s data, and $c_j = 0$ otherwise.

Adding the vectors $\{v_i\}$ across caregivers produces a vector $v = \sum_i v_i$ such that the last non-zero entry of the decrypted vector $\mathrm{Dec}_{\mathcal{R}_s}(v)$ (decrypted entry by entry) corresponds to the global maximum.

This approach would require the computation of l sum aggregations, and, therefore, the complexity of this algorithm increases linearly with the number of possible values for the maximum. This complexity can be greatly reduced by using a typical tree-like approach to determine if the maximum is on the *left* or on the *right* of a given interval. That is, one can run this aggregation scheme with only two possible values for the maximum, implementing it so that the participant caregivers return either one or the other as being closer to the maximum. After one side has been chosen, the algorithm would be used recursively on this selected subinterval to again determine if the maximum is on the *left* or on the *right* side of the interval until the desired precision has been achieved. This approach would require a logarithmic number of sum aggregations on l, but there would be an overhead in the communication caused by multiple protocol interactions.

Now consider a fully homomorphic encryption scheme that is IND-CPA secure with key generation algorithm, encryption algorithm, and decryption algorithm (Gen, Enc, Dec respectively) and with sets of plaintexts and ciphertexts $\mathbf{M}$ and $\mathbf{C}$ respectively with ring structures. Then, the homomorphic properties are as follows: For $m_1, m_2 \in \mathbf{M}$ and a pair of public and secret keys $\mathcal{A}_p, \mathcal{A}_s$,

$$\mathrm{Enc}_{\mathcal{A}_p}(m_1 + m_2) =_p \mathrm{Enc}_{\mathcal{A}_p}(m_1) \oplus \mathrm{Enc}_{\mathcal{A}_p}(m_2)$$

and

$$\mathrm{Enc}_{\mathcal{A}_p}(m_1 \cdot m_2) =_p \mathrm{Enc}_{\mathcal{A}_p}(m_1) \odot \mathrm{Enc}_{\mathcal{A}_p}(m_2),$$

where $(+, \cdot)$ are the operations in $\mathbf{M}$ and $(\oplus, \odot)$ are the corresponding operations on $\mathbf{C}$. $=_p$ denotes indistinguishability of distributions.

This structure dramatically increases the type of functions that can be computed. In particular, one can see that matrix multiplication with encrypted data is easy to compute. This would allow the computation of multiple regression models, for example. The challenge, however, is to eliminate the need for communicating the large matrices to an aggregator, and instead, think of subdividing the problem into smaller problems that each of the caregivers can compute separately.

7.4 Defining Evaluation Metrics

This work is a preliminary attempt to learn the problems and limitations of employing homomorphic encryption techniques to aggregate medical telemetry. This section argues for a reconsideration of the *metrics* for evaluating a solution to the problem of privacy-preserving aggregation of medical telemetry.

The Case for Expressibility

In biomedical image analysis, processing a single fMRI brain image may take a few hours or more if processing the image requires tasks such as manual skull stripping or manual selection of areas of interest by an expert. Gathering the appropriate medical data to perform research may involve contacting multiple institutions and having different IT departments gather the data according to multiple parameters—a task that may take days. Therefore, running time of homomorphic encryption likely should not serve as the primary metric of quality given that instantaneous results are not expected. If gathering medical telemetry for research purposes using homomorphic techniques does not require days of processing, then computation time may not be the only performance metric, or even an important one.

Determining the extent to which a system like *HICCUPS* is useful should involve measuring its *expressibility*. That is, if a system allows the computation of only one type of aggregate, then the capability to express an aggregate problem is limited.

On the other hand, if a system allows the computation of *most* types of aggregates, or even further, *most* types of functions on aggregates, this would mean that the capability of expressing an aggregate problem is high.

The Case against Strict CPU Metrics

Future systems designed with purposes similar to *HICUPPS* should not solely be evaluated with computational performance metrics. Note that the overhead added by computing aggregates on encrypted data is minimal in comparison to other aspects such as communication costs. Compare the following scenarios for computing aggregates on medical telemetry.

Current Practice. With the current infrastructure, researchers and manufacturers acquire statistical data by accessing patients' medical data directly. This scenario requires that *all* relevant data be gathered and aggregated under the management of one institution. Manufacturers query a patient frequently and collect all of the patients' data. Researchers must submit their requests to manufacturers or doctors to obtain the statistical data—potentially taking months to access aggregated data.

Distributed aggregation without privacy. An improvement to the above system would result from the distribution of the workload among caregivers. A manufacturer or researcher submits a request for an aggregate to a designated aggregator. This aggregator broadcasts the manufacturer's request to all of the caregivers. Each caregiver computes the sub-aggregate of his patients' medical data and sends back one single value to the aggregator. Finally, the aggregator combines these sub-aggregates into one aggregate value that will be returned to the manufacturer or researcher. The designated aggregator could be the manufacturer itself, one of the caregivers, or an external entity.

An estimate of the time that is required to compute an aggregate under this scenario is made up of: the time needed to submit the request to the aggregator;

the time needed to broadcast the request to the caregivers; the time caregivers need to compute the sub-aggregate value; the time needed to transmit and process the data from all the caregivers; and finally, the time needed to compute and return the aggregate value to the manufacturer or researcher. If caregivers work in parallel, then the time required to compute the sub-aggregates can be assumed to be the maximum time needed by any one caregiver.

HICCUPS. In practical terms, the hypothetical performance of *HICCUPS* differs from the distributed aggregation scenario mentioned above in that each sub-aggregate computed by the caregivers is encrypted using, first the researcher's or manufacturer's public key and then using the aggregator's public key. Additionally, the aggregator would have to be chosen randomly for each request.

In order to estimate the computation overhead added by *HICCUPS*, it is possible to compare the time overhead of this protocol to the second scenario. There are essentially four pieces of overhead: the time that it takes to perform two encryptions of a single value; the time that is needed by the aggregator to decrypt the first layer of encryption; and the time that is needed to perform k operations on encrypted data, where N is the number of caregivers. Operations on encrypted data include, for example, additions on a group of elliptic curve points. The performance of *HICCUPS* can be estimated by calculating the encryption overhead and adding it to the performance time of the distributed aggregation scenario above.

HICCUPS could be implemented using a variety of encryption schemes. For the purposes of this argument, only two of the more commonly used encryption schemes that have the homomorphic property are considered. These schemes are RSA and ElGamal based on ECC.

Using data provided by Gupta et al. [73], it is possible to estimate the time overhead of *HICCUPS* due to security. As Table 7.1 shows, a manufacturer or researcher would need to tolerate only a delay on the order of 100 milliseconds in order to

protect the security and privacy of patients' data. The addition operation is about 0.59 μseconds and 0.71 μseconds for ECC-160 and ECC-224 bits respectively [38].

Protocol	RSA-1024	ECC-160	RSA-2048	ECC-224
Time Overhead	901.309 ms	380.66 ms	5786.611 ms	527.431 ms

Table 7.1. Estimated overhead added by *HICCUPS* for performing a simple aggregation with 100 caregivers with 1000 records each. The table shows the overheads using four different primitives: securely equivalent RSA-1024 and ECC-160, as well as RSA-2048 and ECC-224.

7.5 Related Work

Related work on secure aggregation includes applications of homomorphic encryption to electronic voting and advances in understanding the theoretical limits of homomorphic encryption.

Homomorphic Encryption Applications

Homomorphic encryption has been highly studied since its introduction by Rivest et al. [128]. Their paper proposed the idea of being able to compute on encrypted data without the need to decrypt. The desired property was to be able to construct an encryption scheme such that you would obtain the same result by *multiplying* two plaintexts and then encrypting the result, or, by first encrypting two plaintexts and then multiplying their corresponding ciphertexts.

Castelluccia et al. have shown that it is possible to compute aggregates such as averages, variances, and standard deviations in a scenario similar to the one described, using only a single homomorphic operation [30]. However, due to the resource constraints of sensor networks, their work uses symmetric key cryptography which imposes a different set of requirements than those in this chapter. For instance, aggregation in sensor networks is hierarchical as opposed to the one-layer aggregator in *HICCUPS*.

The research problem addressed by *HICCUPS* shares similar goals with electronic voting designs. Both systems aim to protect user (patient vs. voter) privacy while providing aggregated result of the private data (statistical function vs. count). However, the solutions for voting systems cannot be easily applied to the telemetry systems because of two major differences in the problems: (1) The constraints of voting systems (voter privacy and result verifiability) are believed to be self-contradictory. This is not the case in telemetry systems. (2) While the voting system requires only one fixed query (counting), it is not feasible to require medical researchers to completely fix their queries. Homomorphic encryption schemes have been successfully applied to voting [117]. However, *HICCUPS*' desired properties and conditions pose different problems. For example, the problem of generating ballots, aggregating votes individually and allowing individual verifications may impose a non-negligible overhead for computing queries on a regular basis. Also, homomorphic encryption has been used in the implementation of universal re-encryption for mixnets [70].

There have been attempts to provide privacy-preserving systems for sharing medical data. Au and Croll propose a privacy-preserving centralized e-health system to provide access to health record data from medical databases distributed across various clinics and hospitals [13]. However, Jefferson et al. [83] exposed some of the privacy risks introduced by a centralized system, such as the voting system studied in their work [84]. Furthermore, Sahai suggested that the existence of an efficient and practical semantically secure public key encryption scheme that is also algebraically homomorphic, would enable minimally interactive distributed data-mining and secure computation [131].

Theoretical Advancements

Goldwasser and Micali introduced the term *semantic security* when they were defining the first probabilistic cryptosystem to formalize the fact that determinis-

tic cryptosystems are not secure against chosen ciphertext attacks [66]. That is, if a deterministic encryption scheme is used, then it is possible that an eavesdropper observing several messages may be able to detect ciphertexts coming from identical messages. This first probabilistic system, proposed by Goldwasser and Micali [67] was a homomorphic encryption system that, while impractical, served as the basis for many other homomorphic encryption systems. This implies that the highest security level cannot be reached by a deterministic homomorphic cryptosystem. Even further, Boneh and Lipton showed that any deterministic algebraically homomorphic cryptosystem can be broken in sub-exponential time [22].

Homomorphic encryption does not provide the *nonmalleability* security requirement. For a cryptosystem to be *nonmalleable* it is necessary that given a ciphertext $c = E(m)$, it should be hard for an adversary to create a ciphertext $c' = E(m')$ such that a relationship between m' and m can be established. It is clear that homomorphic encryption schemes do not satisfy this property since a relationship between m' and m would be given by the homomorphic property [44, 45]. Bellare et al. [17] showed that if a cryptosystem does not provide the *nonmalleability* security requirement, then chosen plaintext indistinguishability IND-CPA is the strongest requirement that may be satisfied by it. In fact, there are homomorphic encryption schemes that satisfy IND-CPA, for example Elgamal [54] and Pallier [117] cryptosystems.

The question of the existence of a fully homomorphic cryptosystem, i.e. one that commutes with both addition and multiplication efficiently, was an open problem until recently. Craig Gentry proved that it is possible to create a fully homomorphic encryption using lattices [63]. The work of Boneh et al. [21] presents a homomorphic encryption scheme that allows the evaluation of 2-DNF formulas on encrypted boolean variables. The defined encryption function supports addition and one multiplication. The same technique can be used in this chapter to enhance the system capabilities for researchers and manufacturers. The assumption of having honest majority of

caregivers made in this chapter can be relaxed by applying the techniques from the work of Ishai et al. [81]. Their work proposes several solutions to perform secure arithmetic computation with no honest majority.

Achieving Differential Privacy with Adequate Utility

A challenge in computing distributed counting queries is that when computing exact answers, it is possible to infer an individual entry by either using a collection of queries or by using external knowledge. For example, consider the case when an analyst requests an answer to the computation of a sum $\sum_{i=1}^{n} a_i$, where a_i is the entry associated to individual i. Now, if somehow the analyst is able to request the computation $(\sum_{i=1}^{k-1} a_i) + (\sum_{i=k+1}^{n} a_i)$, i.e. because a *SELECT* query would exclude individual k, then the analyst would be able to infer the value a_k. Differential privacy gives a formal definition that prevents an adversary from manipulating a system in this way.

Because achieving differential privacy requires that an answer has a small distortion, one of the main challenges is to ensure that systems that provide differential privacy also provide *good answers*; that is, answers that are very close to an exact answer. Dinur et al. [43] provide bounds for the minimum amount of noise that must be added in statistical databases in order to achieve differential privacy. Also, Gupta et al. [72] provide a bound on the minimum number of statistical queries that are needed to answer all queries in a given class. However, the problem is more complicated in practice because analyses can be decomposed into simple queries in a variety of ways. As a result, the final answer to an analysis that is computed with a set of simpler queries, each answered providing differential privacy, may have different utility than the same analysis computed using a different set of simpler queries. In other words: "it depends on how you ask." The issue of decomposing a complex analysis

into the best set of queries to achieve maximum utility when queries are answered via a system that provides differential privacy is an open problem.

7.6 Conclusion

This chapter proposes an alternative model for collecting medical telemetry from medical devices to allow privacy-preserving analyses across institutions. This approach may be used for monitoring for malfunctions and enabling multi-institutional research. This chapter also argues that given the time-line of medical studies, systems to perform multi-institutional analyses should be evaluated on the basis of the usefulness of the information that can be learned and the privacy that they provide rather than the computational overhead that may be incurred.

CHAPTER 8

FUTURE WORK

Recent theoretical advancements have provided robust definitions for characterizing privacy. There is still more work needed to determine how to attain privacy with high utility. The notion of utility is to some extent application dependent; for example, in the context of statistical databases, Rastogi et al. [124] consider a definition of utility that relates to the accuracy of computing counting queries. However, there is ongoing research towards quantifying utility in a more general way, independent of the application. For instance, Sankar et al. [136] study the problem of quantifying the privacy-utility tradeoff using rate distortion theory. Kasiviswanathan, et al. address the question of identifying what can be privately learned [90]. Thus, there are a number of rich intellectual problems whose solutions would improve the current practice of data collection and analysis.

Additional problems may be of particular interest for systems that are deployed more ubiquitously while still relying on constrained devices. For example, an interesting issue is computing privacy preserving aggregates in the presence of high *churning* rates. In the case of CRFIDs, this could involve calculating aggregates when *most* tags do not participate in a given aggregation. This would allow, for instance, a transportation authority to compute an aggregate query involving only the transportation tokens currently present on a given bus. Chan et al.[33] propose a solution to perform privacy-preserving aggregation with fault tolerance to address the issue of aggregation with nodes that occasionally join and leave. Not only is this solution

somewhat impractical on CRFIDS, but it also assumes that a majority of the tags will be present in any given aggregation.

Another compelling question relates to determining the extent to which partial computations can be *off-loaded* from constrained devices to more powerful computing systems without leaking potentially private data. Lauter et al. [111] show promising results with regards to the efficient implementation of fully homomorphic encryption from ring-LWE. More work is needed to explore what such techniques would make possible in the near future on highly constrained devices. For example, if the computation of certain functions could be privately outsourced from constrained devices to more powerful systems, it may be possible to better design distributed systems that limit data leakage while enabling high computational workloads.

At a lower level, there is still more research needed to improve general system support for some particularly constrained devices. In the case of CRFIDs there is room for designing operating systems, distributed or otherwise, that are adequate in highly transient power conditions. It is important to continue to improve application support for these devices, particularly to ensure that they operate securely. Future work could also affect other related areas in computer systems. Personal devices such as smart phones, tablet computers, or other handheld devices could potentially improve the ways in which they collect and aggregate data across individuals. Developments in distributed techniques to provide privacy in aggregations offer an alternative model in which individuals do not have to trust providers with their data.

8.1 Privacy-Preserving Dynamic Queries for Smart Metering

Further work is needed on privacy-preserving smart meters to enable flexible queries. It is possible to use zero-knowledge proofs to compute sophisticated billing and other aggregates while providing privacy. However, this approach is limited in its ability to offer flexibility to request different aggregate functions over time. An

electric utility company may be interested in knowing: How many people in a given neighborhood charge an electric car during the day? or How many people in a given neighborhood have underperforming appliances? These questions may be answered in a privacy-preserving manner using an approach similar to that discussed in Chapter 6.

A challenge is limiting the knowledge that any given analyst may obtain. A query preprocessor needs to implement a policy that applies privacy parameters according to the history of queries that a given analyst has requested. Such a technique has been applied in systems such as GUPT [104]. A related issue may be in implementing a system that allows individuals to have an input in deciding the extent to which they would like to participate in such aggregates. For example, some individuals may opt to enroll in a benefit program where they are required to participate often in this kind of query. Others may choose to decline all participation.

8.2 Smart Phones and Other Personal Data Relays

Using networking stacks such as the one described in Chapter 5, smart phones could take on the role of data relays, obtaining data from CRFIDs that are either attached to a person or a place. A key aspect of providing privacy within the model of this dissertation is that the raw data remains with the individual; both the individual and the service provider can benefit from the data without the service provider having direct access to them. Thus, in this case, a highly constrained device, such as a medical device, may communicate its data to a smart phone, which would then respond to a query for an aggregation. Also, this smart phone would potentially obtain an aggregate from the medical implants of other individuals. In this way, smart phones could take on the role of personal systems that actually interface with service providers and devices from different individuals. Therefore, the techniques discussed in this dissertation may be further extended by the additional capabilities of smart phones. Note that smart phones are members of a more general class of data relays.

For example, Chapter 4 suggests that in the near future each household will have small devices that can control all appliances in the home, e.g. lighting, temperature, vehicle charging, washing appliances, etc. This kind of device would also be a candidate for mediating aggregation between service providers and other individuals.

Thus, it is possible that the work in this dissertation will motivate further research with the purpose of developing techniques that allow an individual's data to stay with him or her as much as possible to protect location data, participation in social networks, preferences about places, movies, books, etc.

8.3 Transportation Payments Using CRFIDs

E-cash protocols are related to the techniques discussed in Chapter 4. These protocols typically involve three parties: banks, users, and merchants and are meant to enable financial transactions that provide privacy guarantees to users and security guarantees to banks and merchants. Thus, users can be guaranteed anonymity or unlinkability in transactions. Banks and merchants would like to attain strong guarantees that ensure that transactions are properly backed by real money.

Many of the cryptographic blocks required for e-cash schemes share commonalities with the cryptographic blocks used in the billing protocol in Chapter 4, such as cryptographic commitments, CL-signatures and ZKPs. However, the protocols differ slightly. Typically, users have accounts with banks and withdraw e-coins, which can be given to merchants in exchange for goods or services. Then merchants, who also have accounts with banks, can deposit the obtained e-coins to obtain financial credit for the coins.

An important feature of e-cash is that transactions between users and merchants may happen *off-line*. That is, e-coins can be spent without requiring that merchants communicate to the bank to verify the validity of the e-coins. E-cash protocols instead discourage double spending by ensuring that when a user double spends an e-coin, he

or she reveals his or her identity in the process. Thus, when a user creates an account with a bank, the bank knows the identity of the user but not how he or she spends e-coins.

The work in this dissertation may serve as a foundation for using CRFIDs to implement various payment tokens. Hinterwälder et al. [77] have explored the possibility of implementing an e-cash scheme using a UMass Moo. Hinterwälder et al. do not discuss the networking aspects of the protocol. The networking stack for CRFIDs discussed in this dissertation may be complementary to the implementation of an e-cash scheme that could be used in transportation systems.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 9

CONCLUSIONS

The work in this dissertation gives evidence to support the thesis that a model for data collection in which individuals keep most of their data and provide only aggregates to service providers is practical in applications that rely on low-cost or ultra-low-power microcontrollers. This dissertation shows that the current model of relying on a trusted data aggregator poses privacy concerns for individuals, in particular in the case of smart metering. This application is only one example of the many data collection systems that are becoming more prevalent. Economic and technological trends suggest that the issue of creating privacy-aware solutions that can be implemented on constrained devices will continue to grow in importance because data collection mechanisms will rely more and more on constrained devices. There are systems that need to be implemented with low-cost devices, and new systems may also benefit from emerging energy harvesting technologies that allow for long-term and low-maintenance deployments. This dissertation also discusses limitations and challenges that remain open—either related to techniques for achieving privacy with high utility or regarding improving the practicality of implementations using constrained systems.

The hope is that this work motivates the development of techniques that place individuals in control of their own private data. The overall vision of this dissertation is to contribute to the goal of providing privacy to individuals, which is ultimately a right that should be preserved not despite achievements in computer science, but because of them.

THIS PAGE INTENTIONALLY LEFT BLANK

BIBLIOGRAPHY

[1] 16-Bit Ultra-Low Power MSP430 Microcontrollers. http://www.ti.com/lsds/ti/microcontroller/16-bit_msp430/product_search.page. Last Viewed: September 29, 2012.

[2] ARM Cortex-M0 core MCUs. http://www.nxp.com/products/microcontrollers/cortex_m0/. Last Viewed: September 29, 2012.

[3] Elster REX2 Smart Meter Teardown. http://www.ifixit.com/Teardown/Elster-REX2-Smart-Meter-Teardown/5710/1. Last Viewed: September 29, 2012.

[4] Embedded Processors. http://www.ti.com/lsds/ti/dsp/embedded_processor.page. Last Viewed: September 29, 2012.

[5] GNU Radio. http://gnuradio.org/. Last Viewed: September 29, 2012.

[6] IAR Embedded Workbench for TI MSP430. http://www.iar.com/en/Products/IAR-Embedded-Workbench/TI-MSP430/. Last Viewed: September 29, 2012.

[7] Kinetis L Series MCUs. http://www.freescale.com/webapp/sps/site/taxonomy.jsp?code=KINETIS_L_SERIES. Last Viewed: September 29, 2012.

[8] The Energy Detective. http://www.theenergydetective.com/. Last Viewed: July 16, 2010.

[9] Functional Specification for an Advanced Metering Infrastructure Version 2, 2007. http://www.mei.gov.on.ca/en/pdf/electricity/smartmeters/AMI%2520Specifications%2520July%25202007.pdf. Last Viewed: July 22, 2010.

[10] ARM Company Milestones, 2011. http://www.arm.com/about/company-profile/milestones.php. Last Viewed: September 29, 2012.

[11] Abelson, H., Allen, D., Coore, D., Hanson, C., Homsy, G., Knight, Jr., T. F., Nagpal, R., Rauch, E., Sussman, G. J., and Weiss, R. Amorphous Computing. *Communications of the ACM 43*, 5 (May 2000).

[12] Agrawal, R., Gehrke, J., and Gunopulos, D. Automatic subspace clustering of high dimensional data. *Data Mining and Knowledge Discovery*, 11 (2005), 5–33.

[13] Au, R., and Croll, P. Consumer-centric and privacy-preserving identity management for distributed e-health systems. In *Hawaii International Conference on System Sciences, Proceedings of the 41st Annual* (Jan. 2008), pp. 234–234.

[14] Balasch, J., Rial, A., Troncoso, C., Geuens, C., Preneel, B., and Verbauwhede, I. PrETP: Privacy-Preserving Electronic Toll Pricing. In *USENIX Security* (2010).

[15] Barreto, P., Galbraith, S., Ó hÉigeartaigh, C., and Scott, M. Efficient pairing computation on supersingular abelian varieties. *Designs, Codes and Cryptography 42* (2007), 239–271. 10.1007/s10623-006-9033-6.

[16] Barry, R. FreeRTOS-a free RTOS for small embedded real time systems. `http://www.freertos.org`. Last Viewed: September 29, 2012.

[17] Bellare, M., Desai, A., Pointcheval, D., and Rogaway, P. Relations among notions of security for public-key encryption schemes. In *Advances in Cryptology—CRYPTO* (1998), pp. 26–45.

[18] Bichsel, P., Camenisch, J., Groß, T., and Shoup, V. Anonymous Credentials on a Standard Java Card. In *Proceedings of the 16th ACM Conference on Computer and Communications Security* (2009).

[19] Biel, L., Pettersson, O., Philipson, L., and Wide, P. ECG Analysis: A New Approach in Human Identification. In *IEEE Transaction on Instrumentation and Measurement* (June 2001), pp. 808–812.

[20] Bishop, M. *Computer Security: Art and Science*. Addison-Wesley Professional, 2003.

[21] Boneh, D., Goh, E.J., and Nissim, K. Evaluating 2-DNF formulas on ciphertexts. *Theory of Cryptography* (2005), 325–341.

[22] Boneh, D., and Lipton, R. Algorithms for black box fields and their applications to cryptography. *Advances in Cryptology* (1996), 223–238.

[23] Brown, M., Hankerson, D., López, J., and Menezes, A. Software implementation of the NIST elliptic curves over prime fields. *Topics in Cryptology* (2001), 250–265.

[24] Buettner, M., Greenstein, B., and Sample, A. Revisiting Smart Dust with RFID Sensor Networks. *Proceedings of the 7th ACM Workshop on Hot Topics in Networks* (Jan 2008).

[25] Buettner, M., Prasad, R., Philipose, M., and Wetherall, D. Recognizing Daily Activities with RFID-Based Sensors. In *Proc. 11th International Conference on Ubiquitous Computing (UbiComp)* (2009).

[26] Buettner, M., and Wetherall, D. A Software Radio-based UHF RFID Reader for PHY/MAC Experimentation. In *RFID, IEEE International Conference on* (2011).

[27] Camenisch, J., and Lysyanskaya, A. A Signature Scheme with Efficient Protocols. In *Security in Communication Networks*. 2003.

[28] Camenisch, J., and Lysyanskaya, A. Signature Schemes and Anonymous Credentials from Bilinear Maps. In *Advances in Cryptology*. 2004.

[29] Campos, H. Open access to my hearts data, Aug. 2012. Invited Talk. ACM Workshop on Medical Communication Systems (MedCOMM).

[30] Castelluccia, C., Mykletun, E., and Tsudik, G. Efficient aggregation of encrypted data in wireless sensor networks. *IEEE Mobiquitous* (2005).

[31] Cavoukian, A. Privacy by design... take the challenge. *Information and Privacy Commissioner of Ontario, Canada* (2009).

[32] CertiVox. MIRACL Crypto SDK. `http://certivox.com/index.php/solutions/miracl-crypto-sdk/`. Last Viewed: September 29, 2012.

[33] Chan, T.H.H., Shi, E., and Song, D. Privacy-preserving stream aggregation with fault tolerance. Tech. rep., Full online technical report, http://eprint.iacr.org/2011/722.pdf, 2011.

[34] Chatterjee, S., and Menezes, A. On cryptographic protocols employing asymmetric pairings—the role of ψ revisited. *Discrete Applied Mathematics 159*, 13 (2011), 1311–1322.

[35] Chen, J. MSP430 Overview and Key Applications. In *TI Developer Conference* (2008).

[36] Chen, L., and Kudla, C. Identity based authenticated key agreement protocols from pairings. In *Computer Security Foundations Workshop, 2003. Proceedings. 16th IEEE* (2003).

[37] Chen, R., Reznichenko, A., Francis, P., and Gehrke, J. Towards statistical queries over distributed private user data. In *Proceedings of the 9th USENIX Symposium on Networked Systems Design and Implementation. USENIX* (2012).

[38] Cohen, H., Miyaji, A., and Takatoshi, O. Efficient elliptic curve exponentiation using mixed coordinates. In *ASIACRYPT '98: Proceedings of the International Conference on the Theory and Applications of Cryptology and Information Security* (London, UK, 1998), Springer-Verlag, pp. 51–65.

[39] Cohn, G., Gupta, S., Froehlich, J., Larson, E., and Patel, S. GasSense: Appliance-level, single-point sensing of gas activity in the home. *Pervasive Computing 6030* (2010), 265–282.

[40] Daley, W. M. Digital Signature Standard (DSS). Tech. rep., Defense Technical Information Center, 2000. http://www.dtic.mil/cgi-bin/GetTRDoc?Location=U2&doc=GetTRDoc.pdf&AD=ADA399987. Last Viewed: September 29, 2012.

[41] DeFranco, E. A., Lian, M., Muglia, L. A., and Schootman, M. Area-level poverty and preterm birth risk: A population-based multilevel analysis. *BioMed Central Public Health 8*, 316 (2008), doi:10.1186/1471–2458–8–316.

[42] Devegili, A., Scott, M., and Dahab, R. Implementing Cryptographic Pairings over Barreto-Naehrig Curves. In *Pairing-Based Cryptography*. 2007.

[43] Dinur, I., and Nissim, K. Revealing information while preserving privacy. In *Proceedings of the 22nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems* (New York, NY, USA, 2003), PODS '03, ACM, pp. 202–210.

[44] Dolev, D., Dwork, C., and Naor, M. Non-malleable cryptography. *Proceedings of the 23rd Annual ACM Sympusium on the Theory of Computing* (1991), 542–552.

[45] Dolev, D., Dwork, C., and Naor, M. Non-malleable cryptography. *SIAM Journal of Computing 30*, 2 (2000), 391–437.

[46] Dorigo, M., Caro, G.D., and Gambardella, L.M. Ant Algorithms for Discrete Optimization. *Artificial Life* (1999).

[47] Dreslinski, R.G., Wieckowski, M., Blaauw, D., Sylvester, D., and Mudge, T. Near-Threshold Computing: Reclaiming Moore's Law Through Energy Efficient Integrated Circuits. *Proceedings of the IEEE* (2010).

[48] Dunkels, A., Gronvall, B., and Voigt, T. Contiki-a Lightweight and Flexible Operating System for Tiny Networked Sensors. In *Local Computer Networks. 29th Annual IEEE International Conference on* (2004), pp. 455–462.

[49] Dwork, C. A Firm Foundation for Private Data Analysis. *Communications of the ACM 54*, 1 (2011), 86–95.

[50] Dwork, C., Kenthapadi, K., McSherry, F., Mironov, I., and Naor, M. Our data, ourselves: Privacy via distributed noise generation. *Advances in Cryptology-EUROCRYPT* (2006), 486–503.

[51] Dwork, C., McSherry, F., Nissim, K., and Smith, A. Calibrating noise to sensitivity in private data analysis. *Theory of Cryptography* (2006), 265–284.

[52] Dwork, C., and Nissim, K. Privacy-preserving datamining on vertically partitioned databases. In *Advances in Cryptology–CRYPTO* (2004), Springer, pp. 134–138.

[53] Dworkin, M., of Standards, Information Technology Laboratory (National Institute, and Division, Technology). Computer Security. *Recommendation for block cipher modes of operation: The CMAC mode for authentication*. US Department of Commerce, National Institute of Standards and Technology, 2005.

[54] Elgamal, T. A public key cryptosystem and a signature scheme based on discrete logarithms. *Information Theory, IEEE Transactions on 31*, 4 (Jul 1985), 469–472.

[55] EPCglobal Inc. EPCglobal Class 1 Generation 2 Air Interface. V. 1.0.9, January 2005.

[56] Ester, M., Kriegel, H., Sander, J., and Xu, X. A density-based algorithm for discovering clusters in large spatial databases with noise. *Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining* (1996).

[57] Ettus Research. Universal software radio peripheral. `http://ettus.com/`. Last Viewed: September 29, 2012.

[58] Ferguson, N., Schneier, B., and Kohno, T. *Cryptography Engineering: design principles and practical applications*. Wiley Publishing, 2010.

[59] Fiat, A., and Shamir, A. How To Prove Yourself: Practical Solutions to Identification and Signature Problems. In *Advances in Cryptology*. 1987.

[60] Fondriest Environmental, 2008. `http://www.fondriest.com`. Last Viewed: September 29, 2012.

[61] Fujisaki, E., and Okamoto, T. Statistical Zero Knowledge Protocols to Prove Modular Polynomial Relations. In *Advances in Cryptology*. 1997.

[62] Galbraith, S. D., Paterson, K. G., and Smart, N. P. Pairings for cryptographers. *Discrete Applied Mathematics 156*, 16 (2008), 3113–3121.

[63] Gentry, C. Fully homomorphic encryption using ideal lattices. In *STOC: Proceedings of the 41st Annual ACM Symposium on Theory of Computing* (New York, NY, USA, 2009), ACM, pp. 169–178.

[64] Gentry, C., and Silverberg, A. Hierarchical ID-based cryptography. *Advances in Cryptology—ASIACRYPT* (2002), 149–155.

[65] Goil, S., Nagesh, H., and Choudhary, A. MAFIA: efficient and scalable subspace clustering for very large data sets. *Technical Report CPDC-TR-9906-010, Northwestern University* (1999).

[66] Goldwasser, S., and Micali, S. Probabilistic encryption & how to play mental poker keeping secret all partial information. *Proceedings of the 14th Annual ACM Symposium on the Theory of Computing* (1982), 365–377.

[67] Goldwasser, S., and Micali, S. Probabilistic encryption. *Journal of Computer and System Sciences 28*, 2 (1984), 270–299.

[68] Goldwasser, S., Micali, S., and Rackoff, C. The knowledge complexity of interactive proof-systems. In *Proceedings of the seventeenth annual ACM symposium on Theory of computing* (1985), ACM, pp. 291–304.

[69] Goldwasser, S., Micali, S., and Rackoff, C. The knowledge complexity of interactive proof-systems. *SIAM Journal of Computing 18* (1989), 186–208.

[70] Golle, P., Jakobson, M., Juels, A., and Syverson, P. Universal Re-encryption for Mixnets. *Topics in Cryptology* (2004).

[71] Gummeson, J., Zhang, P., and Ganesan, D. Flit: A Bulk Transmission Protocol for RFID-Scale Sensors. In *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services* (2012), ACM.

[72] Gupta, A., Hardt, M., Roth, A., and Ullman, J. Privately Releasing Conjunctions and the Statistical Query Barrier. In *Proceedings of the 43rd annual ACM symposium on Theory of computing* (New York, NY, USA, 2011), ACM, pp. 803–812.

[73] Gupta, V., Stebila, D., and Shantz, S.C. Integrating elliptic curve cryptography into the web's security infrastructure. In *Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters* (2004), ACM, pp. 402–403.

[74] Hall, M., Frank, E., Holmes, G., and Pfahringer, B. The weka data mining software: An update. *ACM SIGKDD* (Jan 2009).

[75] Halperin, D., Heydt-Benjamin, T. S., Ransford, B., Clark, S. S., Defend, B., Morgan, W., Fu, K., Kohno, T., and Maisel, W. H. Pacemakers and Implantable Cardiac Defibrillators: Software Radio Attacks and Zero-Power Defenses. In *Proceedings of the 29th Annual IEEE Symposium on Security and Privacy* (May 2008).

[76] Hinneburg, A., and Gabriel, H. Denclue 2.0: Fast clustering based on kernel density estimation. *Advances in Intelligent Data Analysis VII* (2007).

[77] Hinterwälder, G., Paar, C., and Burleson, W.P. Privacy Preserving Payments on Computational RFID Devices with Application in Intelligent Transportation Systems. In *Proceedings of the 8th Workshop on RFID Security and Privacy* (2012).

[78] IAR Systems. IAR Embedded Workbench, 2011. `http://www.iar.com/ewarm`. Last Viewed: September 29, 2012.

[79] Institute for Electric Efficiency. Utility-Scale Smart Meter Deployments. Plans and Proposals, 2012. http://www.edisonfoundation.net/iee/Documents/IEE_SmartMeterRollouts_0512.pdf. Last Viewed: September 29, 2012.

[80] Intel. Corporate Timeline, 2011. http://www.intel.com/about/companyinfo/mu-seum/archives/timeline.htm. Last Viewed: September 29, 2012.

[81] Ishai, Y., Prabhakaran, M., and Sahai, A. Secure arithmetic computation with no honest majority. Cryptology ePrint Archive, Report 2008/465, 2008. http://eprint.iacr.org.

[82] Jawurek, M., Johns, M., and Kerschbaum, F. Plug-in Privacy for Smart Metering Billing. In *Privacy Enhancing Technologies*. 2011.

[83] Jefferson, D., Rubin, A. D., Simons, B., and Wagner, D. Analyzing internet voting security. *Commun. ACM 47*, 10 (2004), 59–64.

[84] Jefferson, D.R., Rubin, D.R., Simons, B., and Wagner, D. A. Security analysis of the secure electronic registration and voting experiment (SERVE). Tech. rep., 2004. www.servesecurityreport.org.

[85] Jiang, X., Dawson-Haggerty, S., and Dutta, P. Design and implementation of a high-fidelity AC metering network. *Conference on Information Processing in Sensor Networks* (2009).

[86] Juels, A. Yoking-Proofs for RFID Tags. In *First International Workshop on Pervasive Computing and Communication Security* (Mar. 2004), R. Sandhu and R. Thomas, Eds.

[87] Kalogridis, G., Efthymiou, C., Denic, S.Z., Lewis, T.A., and Cepeda, R. Privacy for smart meters: Towards undetectable appliance load signatures. In *Smart Grid Communications (SmartGridComm), First IEEE International Conference on* (2010), pp. 232 –237.

[88] Kapron, B., Kempe, D., King, V., Saia, J., and Sanwalani, V. Fast asynchronous byzantine agreement and leader election with full information. In *SODA '08: Proceedings of the nineteenth annual ACM-SIAM Symposium on Discrete Algorithms* (Philadelphia, PA, USA, 2008), Society for Industrial and Applied Mathematics, pp. 1038–1047.

[89] Karlof, C., Sastry, N., and Wagner, D. TinySec: A Link Layer Security Architecture for Wireless Sensor Networks. In *Proceedings of ACM International Conference on Embedded Networked Sensor Systems* (2004).

[90] Kasiviswanathan, S.P., Lee, H.K., Nissim, K., Raskhodnikova, S., and Smith, A. What can we learn privately? In *Foundations of Computer Science. FOCS. IEEE 49th Annual IEEE Symposium on* (2008), pp. 531 –540.

[91] Kim, Y., Schmid, T., Srivastava, M. B., and Wang, Y. Challenges in resource monitoring for residential spaces. In *Proceedings of the First ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings* (New York, NY, USA, 2009), ACM, pp. 1–6.

[92] Kursawe, K., Danezis, G., and Kohlweiss, M. Privacy-friendly Aggregation for the Smart-grid. In *Privacy Enhancing Technologies*. 2011.

[93] Labrosse, J. J. *MicroC/OS-III: The Real-Time Kernel*. Micriμm Press, 2010.

[94] LaCommare, K., and Marnay, C. Microgrids and heterogeneous power quality and reliability. *International Journal of Distributed Energy Resources* (2007).

[95] Lam, H., Fung, G., and Lee, W. A novel method to construct taxonomy of electrical appliances based on load signatures. *Consumer Electronics, IEEE Transactions on 53*, 2 (2007), 653 – 660.

[96] Lamport, L. The part-time parliament. *ACM Transactions on Computer Systems 16*, 2 (May 1998).

[97] Laughman, C., Lee, D., Cox, R., Shaw, S., Leeb, S., Norford, L., and Armstrong, P. Advanced non-intrusive monitoring of electric loads. *IEEE Power and Energy Magazine* (2003), 56–63.

[98] Levis, P., Madden, S., Polastre, J., Szewczyk, R., Whitehouse, K., Woo, A., Gay, D., Hill, J., Welsh, M., Brewer, E., et al. TinyOS: An Operating System for Sensor Networks. *Ambient intelligence 35* (2005).

[99] Lysyanskaya, A., Rivest, R., Sahai, A., and Wolf, S. Pseudonym Systems. In *Selected Areas in Cryptography*. 2000.

[100] Maisel, W. H. Pacemaker and ICD Generator Reliability: Meta-analysis of Device Registries. *JAMA 295*, 16 (2006), 1929–1934.

[101] Maisel, W. H., Moynahan, M., Zuckerman, B. D., Gross, T. P., Tovar, O. H., Tillman, D., and Schultz, D. B. Pacemaker and ICD Generator Malfunctions: Analysis of Food and Drug Administration Annual Reports. *JAMA 295*, 16 (2006), 1901–1906.

[102] Mattern, F., Staake, T., and Weiss, M. ICT for green: how computers can help us to conserve energy. *Conference on Energy-Efficient Computing and Networking* (Jan 2010).

[103] McSherry, F. D. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *Proceedings of the 35th SIGMOD international conference on Management of data* (New York, NY, USA, 2009), SIGMOD, ACM, pp. 19–30.

[104] Mohan, P., Thakurta, A., Shi, E., Song, D., and Culler, D. GUPT: Privacy Preserving Data Analysis Made Easy. In *Proceedings of the International Conference on Management of Data* (2012), ACM, pp. 349–360.

[105] Molina, A., Salajegheh, M., and Fu, K. HICCUPS: Health Information Collaborative Collection Using Privacy and Security. In *Proceedings of the Workshop on Security and Privacy in Medical and Home-Care Systems (SPIMACS)* (November 2009), ACM Press, pp. 21–30.

[106] Molina-Markham, A., Clark, S., Ransford, B., and Fu, K. BAT: Backscatter Anything-to-tag Communication. In *Wirelessly Powered Sensor Networks and Computational RFID* (2012), Joshua R. Smith, Ed. To appear.

[107] Molina-Markham, A., Danezis, G., Fu, K., Shenoy, P., and Irwin, D. Designing privacy-preserving smart meters with low-cost microcontrollers. In *Proceedings of the 16th International Conference on Financial Cryptography and Data Security* (February 2012).

[108] Molina-Markham, A., Shenoy, P., Fu, K., Cecchet, E., and Irwin, D. Private Memoirs of a Smart Meter. In *Proceedings of the 2nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Building* (2010).

[109] Mullen, R.J., Monekosso, D., Barman, S., and Remagnino, P. A review of ant algorithms. *Expert Systems with Applications* (2009).

[110] Mulligan, G. The 6lowpan architecture. In *Proceedings of the 4th workshop on Embedded networked sensors* (New York, NY, USA, 2007), EmNets, ACM, pp. 78–82.

[111] Naehrig, M., Lauter, K., and Vaikuntanathan, V. Can homomorphic encryption be practical? In *Proceedings of the 3rd ACM workshop on Cloud computing security workshop* (2011), ACM, pp. 113–124.

[112] National Security Agency, USA. The Case for Elliptic Curve Cryptography. `http://www.nsa.gov/business/programs/elliptic_curve.shtml`. Last Viewed: September 29, 2012.

[113] Ness, R. B. Influence of the HIPAA privacy rule on health research. *JAMA 18*, 298 (2007), 2164–2170.

[114] Nikitin, P.V., Ramamurthy, S., Martinez, R., and Rao, K.V.S. Passive Tag-to-Tag Communication. In *RFID, IEEE International Conference on* (April 2012).

[115] Osterlind, F., Dunkels, A., Eriksson, J., Finne, N., and Voigt, T. Cross-level Sensor Network Simulation with Cooja. In *Local Computer Networks, Proceedings 2006 31st IEEE Conference on* (2006), pp. 641–648.

[116] Pacific Gas and Electric Company. Smart Meters by the Numbers. http://www.pge.com/myhome/customerservice/smartmeter/installation/. Last Viewed: September 29, 2012.

[117] Paillier, P. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In *EUROCRYPT: Proceedings of Advances in Cryptology* (1999), J. Stern, Ed., vol. 1592, pp. 223–238.

[118] Patel, S., Robertson, T., Kientz, J., and Reynolds, M. At the Flick of a Switch: Detecting and Classifying Unique Electrical Events on the Residential Power Line. *UbiComp* (2007).

[119] Pearson, J., and Moise, T. The Advantages of FRAM-Based Smart ICs for Next Generation Government Electronic IDs, 2007.

[120] Pedersen, T. Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing. In *Advances in Cryptology*. 1992.

[121] Plumb, C., and Zimmermann, P. bnlib: Extended Precision Integer Math Library. http://philzimmermann.com/EN/bnlib/index.html, Last Viewed: September 29, 2012.

[122] Quinn, E. Smart metering and privacy: Existing law and competing policies. *A Report for the Colorado Public Utilities Commision* (2009).

[123] Rastogi, V., and Nath, S. Differentially Private Aggregation of Distributed Time-Series with Transformation and Encryption. In *Proceedings of the ACM SIGMOD International Conference on Management of Data* (2010), pp. 735–746.

[124] Rastogi, V., Suciu, D., and Hong, S. The Boundary Between Privacy and Utility in Data Publishing. In *Proceedings of the 33rd International Conference on Very Large Databases* (2007), VLDB Endowment, pp. 531–542.

[125] Reynolds, M. Beyond RFID: Peer to Peer, Semi-Active RFID Tags. NSF Workshop on Animal Tracking and Physiological Monitoring, 2007. Presentation.

[126] Rial, A., and Danezis, G. Privacy-Preserving Smart Metering. In *Workshop on Privacy in the Electronic Society* (2011).

[127] Ricciardi, L., and Rubel, A. Encouraging the Use of, and Rethinking Protections for De-Identified (and "Anonymized") Health Data. http://www.cdt.org/healthprivacy/20090625_deidentify.pdf. Last Viewed: August 24, 2009.

[128] Rivest, R., Adleman, L., and Dertouzos, M. On data banks and privacy homomorphisms. *Foundations of Secure Computation* (1978).

[129] Rosenberg, B. *Handbook of Financial Cryptography and Security.* Chapman & Hall/CRC, 2010.

[130] Sadasivan, Shyam. An Introduction to the ARM Cortex-M3 Processor. `http://www.arm.com/files/pdf/IntroToCortex-M3.pdf`. Last Viewed: September 29, 2012.

[131] Sahai, A. Computing on encrypted data. *Information Systems Security* (2008), 148–153.

[132] Sakai, R., Ohgishi, K., and Kasahara, M. Cryptosystems based on pairing. In *Proceedings of the Symposium on Cryptography and Information Security* (January 2000), pp. 135–148.

[133] Salajegheh, M., Clark, S. S., Ransford, B., Fu, K., and Juels, A. CCCP: secure remote storage for computational RFIDs. In *Proceedings of the 18th USENIX Security Symposium* (Montreal, Canada, Aug. 2009), pp. 215–230.

[134] Saltzer, J. H., Reed, D. P., and Clark, D. D. End-to-end arguments in system design. In *Proceedings of the Second International Conference on Distributed Computing Systems* (Apr. 1981), ICDCS '81, pp. 509–512.

[135] Sample, A.P., Yeager, D.J., Powledge, P. S., Mamishev, A. V., and Smith, J. R. Design of an RFID-based battery-free programmable sensing platform. *IEEE Transactions on Instrumentation and Measurement* (Nov. 2008).

[136] Sankar, L., Rajagopalan, S.R., and Poor, H.V. A Theory of Utility and Privacy of Data Sources. In *Information Theory Proceedings (ISIT), 2010 IEEE International Symposium on* (June 2010), pp. 2642 –2646.

[137] Scott, M. Implementing Cryptographic Pairings. In *Pairing-Based Cryptography.* 2007.

[138] Scott, M., Costigan, N., and Abdulwahab, W. Implementing Cryptographic Pairings on Smartcards. In *Cryptographic Hardware and Embedded Systems.* 2006.

[139] Sensus. FlexNet AMI System. `http://www.sensus.com/Module/Catalog/ElectricCategory?id=48`. Last Viewed: July 26, 2010.

[140] Shi, E., Chan, T.H.H., Rieffel, E., Chow, R., and Song, D. Privacy-preserving Aggregation of Time-series Data. In *Proceedings of NDSS* (2011).

[141] Spinoff NASA. Intelligent Highway System. `http://http://spinoff.nasa.gov/spinoff1996/36.html`. Last Viewed: September 29, 2012.

[142] Texas Instruments. SYS/BIOS Real-Time Operating System. `http://www.ti.com/tool/sysbios`. Last Viewed: September 29, 2012.

[143] The Economist. Superstructures, December 9, 2010. http://www.economist.com/node/17647603. Last Viewed: April 04, 2011.

[144] Tsiftes, N., and Dunkels, A. A Database in Every Sensor. In *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems* (2011), ACM, pp. 316–332.

[145] Vaikuntanathan, V. Computing Blindfolded: New Developments in Fully Homomorphic Encryption. In *Foundations of Computer Science (FOCS), IEEE 52nd Annual Symposium on* (2011), pp. 5–16.

[146] Wang, J., Hassanieh, H., Katabi, D., and Indyk, P. Efficient and Reliable Low-Power Backscatter Networks. In *Proceedings of the ACM SIGCOMM Conference* (2012).

[147] Welbourne, E., Koscher, K., Soroush, E., Balazinska, M., and Borriello, G. Longitudinal Study of a Building-Scale RFID Ecosystem. In *Proceedings of the International Conference on Mobile Systems, Applications, and Services* (June 2009).

[148] Zhang, H., Gummeson, J., Ransford, B., and Fu, K. Moo: A Batteryless Computational RFID and Sensing Platform. Tech. Rep. UM-CS-2011-020, Department of Computer Science, University of Massachusetts Amherst, Amherst, MA, June 2011.

BIOGRAPHICAL NOTE

Andres David Molina-Markham grew up in México City, México. He obtained his Bachelor's and Master's in Mathematics at the Universidad Nacional Autónoma de México (UNAM). He attended the University of Pennsylvania where he earned a Master's in Mathematics and in Computer and Information Science in 2004 and 2006 respectively. He worked for two years in the Section of Biomedical Image Analysis at the University of Pennsylvania before he started his Ph.D. at the University of Massachusetts Amherst in 2008. Andres was a teaching assistant for Math courses at UNAM and the University of Pennsylvania and for CS466, an Applied Cryptography course for undergraduates at the University of Massachusetts Amherst. Andres is married to Elizabeth Molina-Markham, and when he's not in the lab, he enjoys playing tennis and taking walks with his dog, Luca.