Chapter 10

PRIVACY-PRESERVING POWER USAGE CONTROL IN THE SMART GRID

Chun Hu, Wei Jiang, and Bruce McMillin

Abstract In the smart grid, the power usage of households are recorded and analyzed in (near) real time by utility companies. The usage data enables a utility to manage its electric power supply to neighborhoods more efficiently and effectively. For instance, to prevent a power outage during a peak demand period, the utility can determine the power supply threshold for a neighborhood. When the total power usage of the neighborhood exceeds the threshold, certain households in the neighborhood are required to reduce their energy consumption. This type of power usage control benefits electric utilities and their consumers. However, the energy usage data collected by a utility can also be used to profile an individual's daily activities – a potentially serious breach of personal privacy. To address the problem, this paper specifies distributed, privacy-preserving energy usage control protocols that enable utilities to efficiently manage power distribution while ensuring that individual power usage data is not revealed.

Keywords: Smart grid, power usage control, privacy preservation

1. Introduction

The smart grid provides utilities and consumers with intelligent and efficient ways to manage electric power usage. To achieve this, the grid needs to collect a variety of data related to energy distribution and usage. This expanded data collection raises many privacy concerns, especially with regard to energy consumers. For example, specific appliances can be identified through their electricity usage signatures from data collected by automated meters (at a frequency much higher than the traditional monthly meter readings) [11]. Indeed, research has shown that the analysis of aggregate household energy consumption data over fifteen-minute intervals can determine the usage patterns of most

J. Butts and S. Shenoi (Eds.): Critical Infrastructure Protection VI, IFIP AICT 390, pp. 127–137, 2012.

major home appliances [4, 10]. This increases the likelihood of discovering potentially sensitive information about consumer behavior and so-called activities of daily life (ADL) [12].

Since ADL data is generally personal or private, it should be protected from access by unauthorized entities. For example, a malicious entity could analyze the usage patterns of household appliances in energy usage data, and determine when the victim is not home. The malicious entity could then plan and initiate actions without being easily exposed.

A common strategy to prevent power outages is to dynamically adjust the power consumed by households and businesses during peak demand periods. In this case, a utility may determine a threshold for each neighborhood it services. When the total power usage by a neighborhood exceeds the threshold, some households in the neighborhood are required to reduce their energy consumption based on contractual agreements with the utility.

Implementing threshold-based power usage control (TPUC) requires a utility to collect and analyze power usage data from every household in the participating neighborhoods. Consumers are generally provided with incentives such as reduced rates to encourage participation. In return, the consumers must agree to reduce their power consumption when necessary. For example, the household that consumes the most power in a neighborhood may be required to reduce its consumption to bring the total power usage of the neighborhood under the threshold.

Privacy concerns regarding the fine-granular power usage data that is required to be collected and stored by utilities is the primary obstacle to implementing TPUC in the smart grid. To address these concerns, it is important to design sophisticated TPUC protocols that preserve the privacy of both consumers and utilities. This paper describes two distributed, privacy-preserving protocols that enable utilities to efficiently manage power distribution while satisfying the privacy constraints.

2. Problem Statement

Let $A_1, \ldots, A_n$ be n participating consumers or users from a neighborhood. Furthermore, let f_{TPUC} be a privacy-preserving TPUC protocol given by:

$$f_{\text{TPUC}}\left(\{a_1, \ldots, a_n\}, t\right) \rightarrow \left(\{\delta_1, \ldots, \delta_n\}, \perp\right)$$

where $a_1, \ldots, a_n$ are the average power consumptions during a fixed time interval by consumers $A_1, \ldots, A_n$, respectively; and t is a threshold determined by the utility for the neighborhood. The protocol returns δ_i to consumer A_i and nothing to the utility. The $\delta_1, \ldots, \delta_n$ values are the required power consumption adjustments for the consumers such that $t \geq \sum_{i=1}^{n}(a_i - \delta_i)$. When $t \geq \sum_{i=1}^{n} a_i$, every δ_i is equal to zero, i.e., no power usage adjustments are required. Note that not all the consumers are required to make adjustments at a given time. In general, the specific adjustments that are made depend on the strategy agreed upon by the consumers and the utility.

This paper considers two common power adjustment strategies:

- **Maximum Power Usage:** When the average total energy consumption by a neighborhood over a fixed time interval or round (denoted by $a = \sum_{i=1}^{n} a_i$) exceeds a predefined threshold t, then the consumer who has used the most power during previous round is asked to reduce his or her power consumption. After the next round, if the new a that is computed is still greater than t, then the newly-found maximum energy consumer is asked to reduce his or her usage. This process is repeated until $t \geq a$. Note that the a value is computed at the end of each round. During each round, the consumer who has used the most power can reduce his or her consumption without much discomfort by shutting down one or more household appliances (e.g., washer and dryer) or by adjusting the thermostat temperature setting a few degrees.

- **Individual Power Usage:** If the average total energy consumption a is over the threshold t, then the consumption of every consumer in the neighborhood is reduced based on his or her last usage a_i. The least amount of energy reduction δ_i for each user A_i is determined by the following equation:

$$\delta_i = \frac{a_i}{a}(a - t) \quad \text{and} \quad a = \sum_{i=1}^{n} a_i \tag{1}$$

  where δ_i is a lower bound on the amount of power usage that the user A_i should cut, and a is the average total power usage during the last time interval. After the adjustments, the average total power usage falls below t. Thus, under this strategy, the protocol only has only one round of execution.

Since the collection of fine-granular power usage data by a utility can compromise personal privacy, it is important to prevent the disclosure of such data. Therefore, an f_{TPUC} protocol should satisfy two privacy-preserving requirements:

- **Consumer Privacy:** The average power usage data a_i of a consumer A_i should not be disclosed to any other consumer in the neighborhood or to the utility during the execution of an f_{TPUC} protocol.

- **Utility Privacy:** The threshold t should not be disclosed to the consumers of a neighborhood during the execution of an f_{TPUC} protocol.

The utility privacy requirement must be met because an entity who knows the t values for a number of neighborhoods serviced by a utility could infer the operational capacity and the energy supply distribution of the utility. The public disclosure of this information can cause the utility to lose its competitive advantage. We adopt security definitions from the domain of secure multiparty computation [14, 15] to develop the rigorous privacy-preserving TPUC protocols described in this paper.

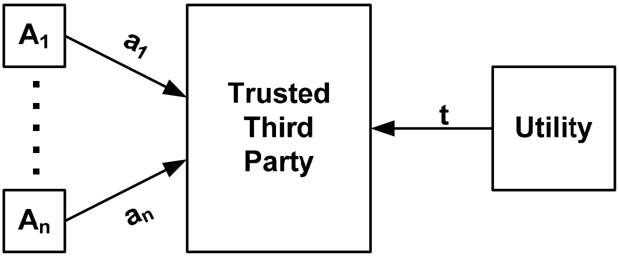

Figure 1. TTP-based f_{TPUC} protocol.

A naive – albeit secure – way to implement an f_{TPUC} protocol is to use a trusted third party (TTP). As shown in Figure 1, each consumer A_i sends his or her a_i value to a TTP while the utility sends its t value to the TTP. Having received these values, the TTP compares t with $a = \sum_{i=1}^{n}$. If $t < a$, the TTP computes each δ_i value and sends it to consumer A_i.

This TTP-based f_{TPUC} protocol easily meets the privacy-preserving requirement. However, such a TTP rarely exists in practice. Therefore, it is necessary to develop f_{TPUC} protocols that do not use a TTP while achieving a similar degree of privacy protection provided by a TTP protocol.

3. Related Work

This section briefly reviews the related work in the field. In particular, it discusses privacy issues in the smart grid, and presents key security definitions from the domain of secure multiparty computation.

Privacy issues in the smart grid are highlighted in [12]. Our work primarily focuses on one of these issues, namely, protecting the release of fine-granular energy usage data in a smart grid environment. Quinn [11] has observed that power consumption data collected at relatively long intervals (e.g., every fifteen or thirty minutes) can be used to identify the use of most major household appliances. Indeed, data collected at fifteen-minute intervals can be used to identify major home appliances with accuracy rates of more than 90 percent [10]. Furthermore, the successful identification rate is near perfect for large two-state household appliances such as dryers, refrigerators, air conditioners, water heaters and well pumps [4]. Lisovich, *et al.* [8] describe the various types of information that can be inferred from fine-granular energy usage data.

In this paper, privacy is closely related to the amount of information disclosed during the execution of a protocol. Information disclosure can be defined in several ways. We adopt the definitions from the domain of secure computation, which were first introduced by Yao [14, 15]. The definitions were subsequently extended to multiparty computation by Goldreich, *et al.* [6].

We assume that the protocol participants are "semi-honest." A semi-honest participant follows the rules of a protocol using the correct inputs. However, the participant is free to later use what he or she sees during the execution of the protocol to compromise privacy (or security). Interested readers are referred to [5] for detailed definitions and models.

The following definition formalizes the notion of a privacy-preserving protocol with semi-honest participants.

Definition. Let T_i be the input of party i, $\prod_i(\pi)$ be i's execution image of the protocol π and s be the result computed from π. π is secure if $\prod_i(\pi)$ can be simulated from $\langle T_i, s\rangle$ and the distribution of the simulated image is computationally indistinguishable from $\prod_i(\pi)$.

Informally, a protocol is privacy-preserving if the information exchanged during its execution does not leak any knowledge regarding the private inputs of any participants.

4. Privacy-Preserving Protocols

We specify two privacy-preserving TPUC protocols: f^1_{TPUC} and f^2_{TPUC} for the maximum power usage strategy and the individual power usage strategy, respectively. We adopt the same notation as before: $A_1, \ldots, A_n$ denote n utility consumers in a participating neighborhood, and $a_1, \ldots, a_n$ denote the average power usage during a fixed time interval set by utility C. Additionally, $a = \sum_{i=1}^{n} a_i$ and $a^m \in \{a_1, \ldots, a_n\}$ denotes the maximum individual energy usage of consumer $A^m \in \{A_1, \ldots, A_n\}$. Without loss of generality, we assume that a^m is unique and $a_1, \ldots, a_n$ are integer values. Since $a_1, \ldots, a_n$ can be fractional values in the real world, the values have to be scaled up to the nearest integers before the protocols can be used. After the results are returned by the protocols, they are adjusted by the appropriate scaling factors to obtain the final values.

The privacy-preserving requirements (consumer privacy and utility privacy) described above are difficult to achieve without using a trusted third party. Consequently, we relax the privacy-preserving requirements slightly in defining the protocols. In particular, the two privacy-preserving requirements are specified as follows:

- **Maximum Power Usage:** Only a and a^m can be disclosed to $A_1, \ldots,$ A_n.
- **Individual Power Usage:** Only a can be disclosed to $A_1, \ldots,$ A_n.

Note that these relaxed requirements permit the design of efficient protocols.

The f^1_{TPUC} and f^2_{TPUC} protocols require several primitive protocols as subroutines. These primitive protocols are defined as follows:

- *Secure_Sum*$(a_1, \ldots, a_n) \to a$
  This protocol has n (at least three) participants. Each participant A_i has an a_i value, which is a protocol input. At the end of the protocol, a is known only to A_1.
- *Secure_Max*$(a_1, \ldots, a_n) \to a^m$
  This protocol has n participants. Each participant A_i has an a_i value,

1. A_1 randomly selects $r \in \{0, N-1\}$, computes $s_1 = a_1 + r \mod N$ and sends s_1 to A_2
2. A_i $(1 < i < n)$ receives s_{i-1}, computes $s_i = s_{i-1} + a_i \mod N$ and sends s_i to A_{i+1}
3. A_n receives s_{n-1}, computes $s_n = s_{n-1} + a_n \mod N$ and sends s_n to A_1
4. A_1 receives s_n and computes $a = s_n - r \mod N$

Figure 2. Secure_Sum protocol.

which is a protocol input. At the end of the protocol, a^m is known to every participant, but a_i is only known to A_i.

- *Secure_Compare*$(a, t) \rightarrow 1$ *if* $a > t$ *and 0 otherwise*
  This protocol has two participants. At the end of the protocol, both participants know if $a > t$.

- *Secure_Divide*$((x_1, y_1), (x_2, y_2)) \rightarrow \frac{x_1+x_2}{y_1+y_2}$
  This protocol has two participants. Participants 1 and 2 submit the private inputs (x_1, y_1) and (x_2, y_2), respectively. At the end of the protocol, both participants know $\frac{x_1+x_2}{y_1+y_2}$.

All these primitive protocols have privacy-preserving properties because the private input values are never disclosed to other participants.

4.1 Implementation

The Secure_Sum protocol can be implemented in several ways. In this paper, we adopt a randomization approach, which yields the protocol specified in Figure 2. Note that N is a very large integer. Because r is randomly chosen, s_1 is also a random value from the perspective of A_2. Therefore, A_2 is not able to discover a_1 from s_1. Following the same reasoning, $a_1, \ldots, a_n$ are never disclosed to the other consumers during the computation process. Because A_1 is the only participant who knows r, only A_1 can derive a correctly.

The remaining three primitive protocols are straightforward to implement. The Secure_Max protocol is implemented using the steps given in [13]. The Secure_Compare protocol is implemented using the generic solution given in [2]. The Secure_Divide protocol is implemented using the methods outlined in [1, 3].

4.2 f^1_{TPUC} Protocol

The f^1_{TPUC} protocol is readily implemented using the primitive protocols. Figure 3 presents the main steps in the protocol.

Since A_1 has the value a, the Secure_Compare protocol in Step 2 can only be executed between consumer A_1 and the utility. However, any consumer can become A_1; this is accomplished via a leader election process among the

1. A_1 obtains $a \leftarrow$ Secure_Sum$(a_1, \ldots, a_n)$
2. A_1 and the utility jointly perform the Secure_Compare protocol
   If Secure_Compare$(a, t) = 1$, then
   (a) Each A_i obtains $a^m \leftarrow$ Secure_Max$(a_1, \ldots, a_n)$
   (b) A^m (self-identified via a^m) reduces his or her energy consumption
3. The above steps are repeated until Secure_Compare$(a, t) = 0$

Figure 3. f^1_{TPUC} protocol.

consumers that determines who becomes A_1. Alternatively, A_1 can be chosen at random before each execution of the protocol.

4.3 f^2_{TPUC} Protocol

In the f^2_{TPUC} protocol, A_1 is also responsible for the Secure_Sum and Secure_Compare operations. An additive homomorphic probabilistic public key encryption (HEnc) system is used as a building block in the protocol. The private key is only known to the utility and the public key is known to all the participating consumers.

Let E_{pk} and D_{pr} be the encryption and decryption functions in an HEnc system with public key pk and private key pr. Without pr, it is not possible to discover x from $E_{pk}(x)$ in polynomial time. (Note that, when the context is clear, the subscripts pk and pr in E_{pk} and D_{pr} are omitted.) The HEnc system has the following properties:

- The encryption function is additive homomorphic, i.e., $E_{pk}(x_1) \times E_{pk}(x_2) = E_{pk}(x_1 + x_2)$.
- Given a constant c and $E_{pk}(x)$, $E_{pk}(x)^c = E_{pk}(c \cdot x)$.
- The encryption function has semantic security as defined in [7], i.e., a set of ciphertexts do not provide additional information about the plaintext to an unauthorized party or $E_{pk}(x) \neq E_{pk}(x)$ with very high probability.
- The domain and the range of the encryption system are suitable.

Any HEnc system is applicable, but in this paper, we adopt Paillier's public key homomorphic encryption system [9] due to its efficiency. Informally, the public key in the system is (g, N), where N is obtained by multiplying two large prime numbers and $g \in \mathbb{Z}^*_{N^2}$ is chosen randomly.

To implement the f^2_{TPUC} protocol and according to Equation (2), each consumer A_i needs to calculate $\frac{a_i \cdot t}{a}$ between A_i and the utility C so that a_i is not disclosed to C and t is not disclosed to A_i. We adopt the Secure_Divide primitive and an HEnc system to solve the following problem:

$$\delta_i = \frac{a_i}{a}(a - t) = a_i - \frac{a_i \cdot t}{a} \tag{2}$$

1. A_1 obtains $a \leftarrow$ Secure_Sum$(a_1, \ldots, a_n)$
2. A_1 and utility C jointly perform the Secure_Compare protocol
   If Secure_Compare$(a, t) = 1$, then
   (a) A_1 randomly selects r from $\{0, N-1\}$
      - Set $y_1 = N - r$ and $y_2 = a + r \mod N$
      - Send y_1 to $A_2, \ldots, A_n$ and y_2 to C
   (b) Each A_i $(2 \leq i \leq n)$ randomly selects r_i from $\{0, N-1\}$
      - Compute $E(t)^{a_i}$ to get $E(a_i \cdot t)$
      - Set $x_{1i} = N - r_i$ and $s_i = E(a_i \cdot t) \times E(r_i) = E(a_i \cdot t + r_i)$
      - Send s_i to C
   (c) Utility C sets $x_{2i} = D(s_i)$ for $2 \leq i \leq n$
   (d) Each A_i $(2 \leq i \leq n)$ with input (x_{1i}, y_1) and C with input (x_{2i}, y_2) jointly perform the Secure_Divide protocol
      - A_i obtains $\kappa_i =$ Secure_Divide$((x_{1i}, y_1), (x_{2i}, y_2))$
      - A_i sets $\delta_i = a_i - \kappa_i$
      - A_i reduces his or her power consumption according to δ_i

Figure 4. f^2_{TPUC} protocol.

Also, we assume that $E(t)$ is initially broadcasted by the utility.

Figure 4 presents the main steps in the f^2_{TPUC} protocol. A_1 is the designated consumer in the participating neighborhood, who is responsible for computing a and distributing $N - r$ to the other consumers and $a + r \mod N$ to the utility. Note that the value of a computed in Step 1 should not include the value a_1 (this is easily achieved via a small modification to the Secure_Sum protocol) and A_1 does not adjust his or her energy consumption. This prevents the disclosure of t to A_1. For instance, if A_1 obtains a δ_1, then A_1 can derive t based on Equation (2). To ensure fairness, A_1 can be randomly selected from among the participating consumers before each execution of the protocol.

The purpose of Step 2(a) is to hide the a value from the utility and the other consumers. Since r is chosen randomly, y_1 and y_2 are randomly distributed in $\{0, N-1\}$. As a result, the other consumers $A_2, \ldots, A_n$ cannot discover a from y_1; similarly, the utility cannot discover a from y_2.

The goal of Step 2(b) is to hide a_i from the utility and t from A_i. Since the encryption scheme is semantically secure, from $E(t)$ and without the private key, the consumers cannot learn anything about t. In addition, because r_i is chosen randomly, the x_{2i} value computed in Step 2(c) does not reveal any information regarding a_i.

The operations performed in Steps 2(b) and 2(c) are based on the additive homomorphic property of the encryption function E. Since $x_{1i} + x_{2i} = a_i \cdot t$ and $y_1 + y_2 = a$, $\kappa_i = \frac{a_i \cdot t}{a}$. Therefore, the protocol correctly returns δ_i for each A_i, except for A_1.

5. Protocol Efficiency and Privacy

This section analyzes the complexity and privacy properties of the protocols.

5.1 Protocol Complexity

Since the Secure_Sum protocol only performs additions and each participant only turns in one input, the protocol is very efficient. The complexity of the Secure_Compare protocol depends on the number of bits needed to represent the maximum value between a and t. Once the number of bits required to represent these numbers is fixed, the complexity of the Secure_Compare protocol is constant. The main operation in the Secure_Max protocol is the comparison of two numbers, so the protocol itself is very efficient. In the case of a neighborhood with 1,000 consumers, if the communication delay is negligible, then the running time of the f^1_{TPUC} protocol is just a few seconds.

According to [1], the computational cost of the Secure_Divide protocol is bounded by $\mathrm{O}(\log l)$, where l is the number of bits used to represent the maximum value between $a_i \cdot t$ and a. Because $l = 20$ is generally sufficient in our problem domain, the computational cost of Secure_Divide is constant and very small. If the number of consumers in the neighborhood is small and the utility can execute the Secure_Divide protocol with each consumer concurrently, then the f^2_{TPUC} protocol can also be completed in a few seconds. Based on the above analysis, it is reasonable for the utility to set up a fifteen- or thirty-minute interval between executions of the protocols.

5.2 Protocol Privacy

With regard to the f^1_{TPUC} protocol, a is disclosed to A_1 and a^m is disclosed to all the participating consumers. Since a is aggregated information, the disclosure of a can hardly cause any privacy violations. Although a^m is disclosed, no one can link a^m to a particular consumer. Thus, the disclosure risk of the f^1_{TPUC} protocol is not significant.

The f^2_{TPUC} protocol only discloses a to A_1, so it is more privacy preserving than the f^1_{TPUC} protocol. However, because the Secure_Divide protocol has to be executed between every consumer and the utility, the protocol is less efficient than f^1_{TPUC}. Therefore, depending on whether or not efficiency is more important than privacy, one protocol is more or less applicable than the other protocol in a real-world situation.

6. Conclusions

Intelligent power usage control in the smart grid requires utilities to collect fine-granular energy usage data from individual households. Since this data can be used to infer information about the daily activities of energy consumers, it is important that utility companies and their consumers employ privacy-preserving protocols that facilitate intelligent power usage control while protecting sensitive data about individual consumers.

The two privacy-preserving protocols described in this paper are based on energy consumption adjustment strategies that are commonly employed by utilities. Although the protocols are not as privacy-preserving as the ideal model that engages a trusted third party, they are efficient and limit the amount of information disclosed during their execution. Our future research will focus on refining these protocols and will develop privacy-preserving protocols for other types of energy usage control.

Acknowledgements

The research efforts of the first two authors were supported by the Office of Naval Research under Award No. N000141110256 and by the NSF under Grant No. CNS 1011984. The effort of the third author was supported in part by the Future Renewable Electric Energy Distribution Management Center, an NSF Engineering Research Center, under Grant No. EEC 0812121; and by the Missouri S&T Intelligent Systems Center.

References

[1] M. Atallah, M. Bykova, J. Li, K. Frikken and M. Topkara, Private collaborative forecasting and benchmarking, *Proceedings of the ACM Workshop on Privacy in the Electronic Society*, pp. 103–114, 2004.

[2] A. Ben-David, N. Nisan and B. Pinkas, FairplayMP: A system for secure multi-party computation, *Proceedings of the Fifteenth ACM Conference on Computer and Communications Security*, pp. 257–266, 2008.

[3] M. Blanton, Empirical Evaluation of Secure Two-Party Computation Models, CERIAS Technical Report TR 2005-58, Center for Education and Research in Information Assurance and Security, Purdue University, West Lafayette, Indiana, 2005.

[4] S. Drenker and A. Kader, Nonintrusive monitoring of electric loads, *IEEE Computer Applications in Power*, vol. 12(4), pp. 47–50, 1999.

[5] O. Goldreich, *The Foundations of Cryptography, Volume II: Basic Applications*, Cambridge University Press, Cambridge, United Kingdom, 2004.

[6] O. Goldreich, S. Micali and A. Wigderson, How to play any mental game, *Proceedings of the Nineteenth Annual ACM Symposium on the Theory of Computing*, pp. 218–229, 1987.

[7] S. Goldwasser, S. Micali and C. Rackoff, The knowledge complexity of interactive proof systems, *Proceedings of the Seventeenth Annual ACM Symposium on the Theory of Computing*, pp. 291–304, 1985.

[8] M. Lisovich, D. Mulligan and S. Wicker, Inferring personal information from demand-response systems, *IEEE Security and Privacy*, vol. 8(1), pp. 11–20, 2010.

[9] P. Paillier, Public key cryptosystems based on composite degree residuosity classes, *Proceedings of the Seventeenth International Conference on the Theory and Application of Cryptographic Techniques*, pp. 223–238, 1999.

[10] E. Quinn, Privacy and the New Energy Infrastructure, CEES Working Paper No. 09-001, Center for Energy and Environmental Security, University of Colorado Law School, Boulder, Colorado, 2009.

[11] E. Quinn, Smart Metering and Privacy: Existing Law and Competing Policies, Report for the Colorado Public Utilities Commission, University of Colorado Law School, Boulder, Colorado, 2009.

[12] Smart Grid Interoperability Panel, Guidelines for Smart Grid Security (Introduction, Volumes 1, 2 and 3), NIST IR 7628, National Institute of Standards and Technology, Gaithersburg, Maryland, 2010.

[13] L. Xiong, S. Chitti and L. Liu, Topk queries across multiple private databases, *IEEE Security and Privacy*, pp. 145–154, 2005.

[14] A. Yao, Protocols for secure computations, *Proceedings of the Twenty-Third Annual Symposium on the Foundations of Computer Science*, pp. 160–164, 1982.

[15] A. Yao, How to generate and exchange secrets, *Proceedings of the Twenty-Seventh Annual Symposium on the Foundations of Computer Science*, pp. 162–167, 1986.