Verifiable and Privacy-preserving Fine-Grained Data-Collection for Smart Metering

Moreno Ambrosin*, Hossein Hosseini†, Kalikinkar Mandal†, Mauro Conti*, Radha Poovendran†
*University of Padua, Department of Mathematics, email:{ambrosin,conti}@math.unipd.it
†University of Washington, Department of Electrical Engineering, email:{hosseinh,kmandal,rp3}@uw.edu

***Abstract*—The Advanced Metering Infrastructure (AMI) is a key component of future Smart Grids, which allows fine-grained and real-time monitoring of the energy consumption of private houses or entire buildings. In an AMI, intelligent devices, namely Smart Meters (SMs), communicate with a Utility Provider (UP) for both management and billing purposes. However, this technology comes at a price in terms of privacy for customers. Indeed, a UP or an external attacker can infer sensitive information on end users, by analyzing their metering data. Therefore, protecting customer's privacy becomes a primary concern in the design of future AMIs. In this paper, we present the initial design of a privacy-preserving AMI and propose a scheme for fine-grained anonymous metering data transmission. Our scheme allows SMs to periodically transmit anonymous metering data to the UP through a multi-hop path composed by other SMs. Unlike previously proposed approaches, our scheme does not rely on trusted third parties. We consider a scenario where SMs are either honest-but-curious or controlled by the UP, whose aim is to deanonymize the received metering data. Our scheme also provides a secure mechanism that allows a Verification Center to verify the functionality of every SM, by accessing its internal log.**

***Index Terms*—Smart Grid; Smart Metering; Advanced Metering Infrastructure; verifiability; security; privacy**

I. Introduction

The advent of Smart Grid technologies has the potential to bring an advanced control over both energy generation and distribution in energy networks. A fundamental component of modern Smart Grids is Advanced Metering Infrastructure (AMI), which intends to periodically communicate with intelligent metering devices, namely Smart Meters (SMs), to collect and analyze the energy usage of private houses or buildings. SMs transmit information of different types, so that a Utility Provider (UP) can collect fine-grained time series of metering data, for management purposes, or aggregate weekly energy consumption data, for billing [5], [3]. However, the benefits of AMI come at the price of losing privacy for utility customers, as privacy-sensitive information, such as the detailed user activities, can be inferred from raw metering data [10], [9]. Therefore, it is important to deploy mechanisms to preserve the privacy of customers, with respect to both the UP and external attackers, without compromising the utility of the data.

Several solutions have been proposed to offer privacy for customers. A possible approach is aggregating the metering data before delivering them to the UP [2], [6], [8], [11], [14]. However, compared to having the individual measurements, aggregation reduces the utility of the data for UP. Another possible approach is using trusted, or semi-trusted, third-party anonymizers [4], [5], which, while providing strong privacy guarantees, might be difficult to deploy and manage. The work in [13] also proposed a privacy-preserving scheme for billing. This scheme, however, requires the intervention of the customer in computations, which severely limits the practicality of the solution.

Contribution: In this paper, we present our ongoing research for designing a comprehensive privacy-preserving scheme for AMI, focusing on the anonymous reporting of metering data. Our scheme allows SMs to transmit fine-grained metering data to the UP, while preserving the unlinkability between the raw metering data and the source SM, and without the need for a trusted anonymizer. We also guarantee the authenticity and the integrity of the metering data received by the UP. We use a realistic system model and aim at a practical and easily deployable solution, with minimal modifications to the existing infrastructure.

We achieve SM anonymity by running a collaborative protocol among SMs. SMs also keep the log of their activities, so that a trusted Verification Center (VC) can periodically verify their functionality. We assume that SMs are honest-but-curious; however, the UP may deploy some malicious SMs in order to breach (or reduce) the anonymity of SMs. Also, an external attacker can eavesdrop the communication, to try to breach (or reduce) the anonymity of SMs and/or send fake data to the UP.

Organization: The rest of the paper is organized as follows. In Section II we introduce our system model, security model, and design goals. In Section III we present our scheme for anonymous reporting of metering data, and SMs behavior verification. In Section IV we analyze the proposed scheme based on our design goals. Finally, in Section V we conclude the paper and briefly state the directions of the future research.

II. Reference Model and Assumptions

In this section, we present our reference system model, security model, and design goals for our AMI.

A. System Model

Figure 1 shows our reference system model. We use realistic model and assumptions, which are similar to the previous works [1], [4], [7]. In our model, each SM periodically (e.g., every 10 minutes) transmits a *report*, containing timestamped metering data, to the UP. We assume the AMI to be a mesh network, where SMs are capable of both wireless connectivity, used to communicate with peer SMs, and wired connectivity, used to deliver data to the UP. SMs are organized in separate

978-1-4673-7876-5/15/$31.00 ©2015 IEEE

groups, within which each SM is in the wireless communication coverage of its neighborhood.

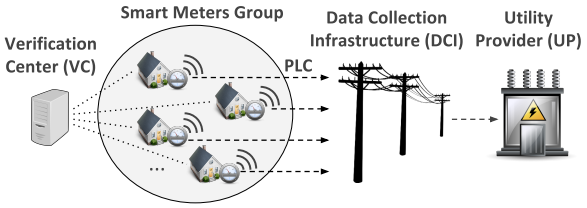

Fig. 1. System Model

We also suggest that SMs use Power Line Communications (PLC) to deliver data to the UP, as leveraging the existing electrical infrastructure gives several benefits, such as reduced costs of deployment and maintenance. Therefore, we assume that SMs are connected to a Data Collection Infrastructure (DCI), which relays the data to the UP.

In our model, SMs are capable of performing public key and symmetric key encryption/decryption and hashing [3]. SMs also keep a *log* of their activities in a file, so that a trusted Verification Center (VC) can periodically (e.g., once a month) access the internal log of SMs, and verify their behavior. Note that, VC *does not participate* in the protocol between SMs and the UP. SMs are *tamper resistant* devices, i.e., they cannot be accessed or compromised by an adversary [4], [13]. In addition, each SM has a trusted tamper resistant component, which controls SM's internal log and observes the outgoing traffic.

B. Attacker Model

We assume the presence of malicious SMs, which are deployed by the UP with the goal of breaching the anonymity of SMs. Note that, since UP is interested in receiving the metering data, malicious SMs (i.e., SMs controlled by the UP) have no incentive to drop the packets or perform Denial of Service attacks in the network. Non-malicious SMs are *honest-but-curious*, i.e., they follow the protocol, but are curious to find a link between a received message and its initiator. Finally, we assume the presence of an external attacker, who tries to breach the anonymity of the SMs by eavesdropping the wireless communications. She also intends to interfere with the protocol by sending fake metering data to the UP.

C. Design Goals

We have the following design goals for our system:

Confidentiality. The protocol should provide end-to-end (i.e., SM to UP) confidentiality for the raw metering data.

Authenticity and Integrity. UP should be able to authenticate the data (i.e., the data has been originated by a legitimate SM) and verify its integrity.

Anonymity. The metering data sent by each SM should be anonymous, i.e., no entity, including the UP, other SMs, or an external attacker, should be able to link the metering data with the identity of the source SM.

Verifiability. The VC should be able to verify the behavior of each SM.

Low Complexity. The protocol should have minimal requirements and introduce minimal modifications to the current infrastructure, to have low overhead and easy deployment.

III. Our Data Reporting Scheme

In this section, we present our scheme for anonymous metering data reporting and a posteriori verification of SMs' behavior. In the rest of the paper, we will use Src to indicate a source SM and use Int to indicate the intermediate SMs in the multi-hop path. Each SM assigns a permanent ID, $SM_0, \ldots, SM_{N_{(\cdot)}}$, to each of the $N_{(\cdot)}$ reachable SMs, i.e., the SMs in its wireless coverage. We also assume all the SMs in a group to be loosely synchronized with the UP. Table I summarizes the notation we use in this paper. In what follows, we provide the details of our scheme.

TABLE I
Notation

Notation	Description
N_S, N_I	Number of reachable meters for Src and Int, respectively.
SM_j	The j-th SM in the neighborhood of a SM.
$\tilde{t} = \lfloor \frac{t}{T} \rfloor$	Quantized version of current time slot based on T.
$d_{S,\tilde{t}}$	Metering data of Src at interval $\tilde{t}$.
$\nu_{S,\tilde{t}}$, $\nu_{I,\tilde{t}}$	Random nonce of Src and Int, respectively, at interval $\tilde{t}$.
k_G	Group key shared between a group G of SMs and UP.
pk_S, sk_S	Public and secret keys pair of Src.
pk_I, sk_I	Public and secret keys pair of Int.
pk_j, sk_j	Public and secret keys pair of SM_j.
pk_{UP}, sk_{UP}	Public and secret keys pair of UP.
pk_{VC}, sk_{VC}	Public and secret keys pair of VC.
$\mathrm{E}_k(\cdot)$, $\mathrm{D}_k(\cdot)$	Public key Encryption and Decryption with key k.
$\mathrm{H}(\cdot)$	Hash function.
$\mathrm{HMAC}_k(\cdot)$	Hash-based Message Authentication Code, with key k.
$X^{(1)}, X_{(2)}$	First byte/last two bytes of a bit string X (decimal value).
q	Probability that Int forwards a received packet to another SM.
$h = \lceil \frac{1}{1-q} \rceil$	Smallest integer not less than $\frac{1}{1-q}$.

A. Anonymous Metering Data Reporting

we divide the time into intervals of T seconds, and represent the current time-interval as $\tilde{t} = \lfloor \frac{t}{T} \rfloor$. Each SM generates a random nonce $\nu_{S,\tilde{t}}$, at the beginning of each interval $\tilde{t}$. At every interval $\tilde{t}$, each SM transmits its metering data to the UP through a random multi-hop path composed by other SMs in its group. The source SM, Src, starts the process of metering data transmission by randomly selecting one of the reachable SMs as the first hop. Every intermediate SM, Int, forwards a received packet to another peer SM with probability q. Int chooses the next hop, based on the random string Y, obtained as the hash of the random string X, contained in each packet it receives, concatenated with its current nonce. Let $c = Y^{(1)} (\bmod\ h)$ and $j = Y_{(2)} (\bmod\ N_I)$; if $c = 0$, which occurs with probability $1-q$, Int forwards the packet to the UP, otherwise it forwards the packet to the j-th SM in its neighborhood.

Our protocol for anonymous reporting of metering data consists of three steps: *Initialization*, *Forwarding* and *Data Verification*.

1) Initialization: The source SM, Src, concatenates the metering data with the corresponding time-interval, the group ID, G, and $\text{HMAC}_{k_G}(d_{S,\tilde{t}})$, and produces the message M by encrypting it with the public key of the UP. Src then generates a random bit string X, such that $X^{(1)} \neq 0$ (since each message should traverse at least one hop before being forwarded to the UP) and computes $j = X_{(2)} (\text{mod } N_S)$. Finally, Src creates PKT by encrypting $M \| \tilde{t} \| X \| \text{HMAC}_{k_G}(\tilde{t} \| X)$ with the public key of SM_j, the j-th SM in its neighborhood, and transmits PKT to SM_j.

The SM's log component encrypts the string $X \| \mathbf{0} \| \tilde{t}$ with the public key of the VC and stores it into its log file, for VC to later verify the SM's functionality. Note that $\mathbf{0}$ is a zero string with the same length of the nonce. The details of the initialization phase are provided in Algorithm 1.

Algorithm 1 Initialization: Src transmits its metering data to the first hop.

Input: Metering data $d_{S,\tilde{t}}$ at time $\tilde{t}$; UP's public key pk_{UP}; VC's public key pk_{VC}; Group key k_G; ID and public key of the N_S reachable SMs; Integer h.

$M \leftarrow E_{pk_{UP}}(d_{S,\tilde{t}} \parallel \tilde{t} \parallel G \parallel \text{HMAC}_{k_G}(d_{S,\tilde{t}}))$;
$X \leftarrow$ Random bit string, where $X^{(1)} (\text{mod } h) \neq 0$;
$j \leftarrow X_{(2)} (\text{mod } N_S)$;
$PKT \leftarrow E_{pk_j}(M \parallel \tilde{t} \parallel X \parallel \text{HMAC}_{k_G}(\tilde{t} \parallel X))$;
Transmit PKT to SM_j;
Store $E_{pk_{VC}}(X \| \mathbf{0} \| \tilde{t})$ into the log;

2) Forwarding: When an intermediate SM, Int, receives a packet, it first decrypts the packet with its secret key, sk_I, to obtain $M \| \tilde{t} \| X \| \phi$. Then, Int checks the following conditions: $\tilde{t}$ is not the current or the previous time-interval, $\text{HMAC}_{k_G}(\tilde{t} \| X) \neq \phi$, or X has been already received by Int at the current or the previous time-interval. If any of these conditions hold, Int drops the packet and stops the procedure; otherwise Int continues the procedure by storing X in a local memory, which will be erased after two time-intervals. Int then computes Y as $\text{H}(X \| \nu_{I,\tilde{t}})$ and obtains $c = Y^{(1)} (\text{mod } h)$ and $j = Y_{(2)} (\text{mod } N_I)$. If $c = 0$, Int forwards M to UP; otherwise, it creates PKT by encrypting $M \| \tilde{t} \| Y \| \text{HMAC}_{k_G}(\tilde{t} \| Y)$ with the public key of SM_j, the j-th SM in its neighborhood, and forwards PKT to SM_j. Similar to the source SM, Int's log component encrypts the string $Y \| \nu_{I,\tilde{t}} \| \tilde{t}$ with the public key of the VC and stores it into its log, for VC to later verify the SM's functionality. The details of the initialization phase are provided in Algorithm 2.

3) Data Verification: Upon receiving the message M, UP first extracts $d_{S,\tilde{t}} \| \tilde{t} \| G \| \phi$ by decrypting M with its secret key sk_{UP}. If $\phi = \text{HMAC}_{k_G}(d_{S,\tilde{t}})$, UP verifies both the authenticity and the integrity of the metering data; otherwise it discards the packet.

Algorithm 2 Forwarding: Int forwards the received packet to either UP or another SM, or discards the packet if it is not verified.

Input: Received packet PKT; Secret key sk_I; VC's public key pk_{VC}; Group key k_G; Random nonce $\nu_{I,\tilde{t}}$ at time $\tilde{t}$; ID and public key of the N_I reachable meters; Integer h; Local memory.

$M \parallel \tilde{t} \parallel X \parallel \phi \leftarrow D_{sk_I}(PKT)$;
if ($\tilde{t}$ not current or the previous time-interval) $\vee$ ($\text{HMAC}_{k_G}(\tilde{t} \| X) \neq \phi$) $\vee$ (X already received in the current or the previous time-interval) **then**
    Discard the packet and **stop**;
end if
Store X in local memory;
$Y \leftarrow \text{H}(X \parallel \nu_{I,\tilde{t}})$;
$c \leftarrow Y^{(1)} (\text{mod } h)$; $j \leftarrow Y_{(2)} (\text{mod } N_I)$;
if $c = 0$ **then**
    Forward M to UP;
else
    $PKT \leftarrow E_{pk_j}(M \parallel \tilde{t} \parallel Y \parallel \text{HMAC}_{k_G}(\tilde{t} \parallel Y))$;
    Forward PKT to SM_j;
end if
Store $E_{pk_{VC}}(Y \parallel \nu_{I,\tilde{t}} \parallel \tilde{t})$ into the log;

B. Verifying SMs' Functionality

Our scheme allows the VC to verify the correct functionality of SMs by accessing their internal log. In particular, the VC selects from the log of each SM, the entries corresponding to the packets originated by the SM, i.e., the ones of the form $\tilde{t} \parallel \mathbf{0} \parallel X$, and, for each of them, VC verifies the behavior of the forwarding SMs. Let Src be the source SM for the entry $\tilde{t} \parallel \mathbf{0} \parallel X$. The VC performs the following operations:

1. VC computes the ID j of the next SM as $X_{(2)} (\text{mod } N_S)$;
2. VC searches the SM_j's log for the entries that contain $\tilde{t}$;
3. By using $\nu_{j,\tilde{t}}$ (which is the same for each entry containing $\tilde{t}$), VC computes Y as $\text{H}(X \parallel \nu_{j,\tilde{t}})$ and $c = Y^{(1)} (\text{mod } h)$;
4. If $c \neq 0$, the VC repeats from operation (1), with $X = Y$ and $j = i$;
5. Otherwise, the VC concludes that SM_j (correctly) forwarded M to the UP, and terminates the verification.

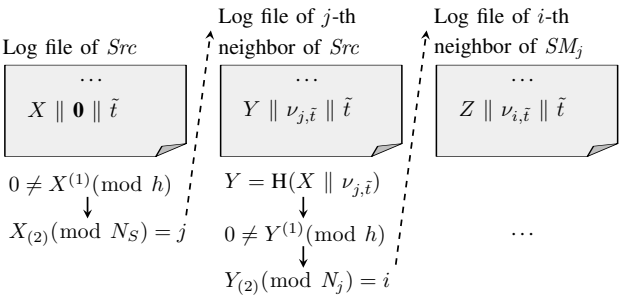

Fig. 2. Message chain reconstruction performed by the VC.

IV. Discussion

In this section, we provide a discussion on how the proposed scheme meets our design goals we introduced in Section II-C.

Confidentiality. Each SM encrypts its metering data with the UP's public key before transmitting the data. Therefore, our scheme guarantees end-to-end (i.e., SM to UP) confidentiality for metering data.

Authenticity and Integrity. We intend to protect UP from both *Impersonating Attack*, where an external attacker tries

to send fake metering data to the UP, and *Replay Attack*, where the attacker tries to re-transmit a previously-transmitted valid packet. We prevent these attacks at the SM level. Each intermediate SM, Int, memorizes all the packet IDs it receives in the current or the previous time-interval. Every time it receives a packet, Int checks if the time-interval $\tilde{t}$ specified in the packet does not correspond with the current or the previous time-interval, or whether it had already received another packet with the same message ID, or if it cannot verify the HMAC output. If any of first two conditions hold, Int realizes that a replay attack has been performed, and, if the third condition holds, Int declares an impersonating attack. In any case, Int discards the packet, and terminates the forwarding procedure. Also, by verifying HMAC output, UP ensures that the data has been originated by a legitimate SM and it has not been altered. Therefore, our scheme provides both authenticity and integrity for the metering data transmitted to the UP.

Anonymity. Our scheme guarantees anonymous metering data reporting for SMs, with respect to both the UP and an external attacker. Packets from SMs traverse a multi-hop random path before being sent to the UP, where each SM randomly chooses the next-hop SM for forwarding the data. This gives to SMs the level of anonymity provided by the protocol Crowds [12]. Moreover, in our protocol there is no evident correlation between packets that a SM receives and forwards. This prevents an external attacker from mounting *Correlation Attacks* by eavesdropping the wireless communications. SMs also put a quantized version of the time-interval, instead of the real time reference, into the packets, so that UP cannot perform the *Timing Attacks*.

Verifiability. As stated in Section III-B, VC can identify the irregularities in the execution of the protocol by SMs, based on the information contained in SM's log files. As an example, consider SM_j in Figure 2. Given $i = Y_{(2)} (\bmod\ N_j)$, if the log file of the i-th neighbor of SM_j (SM_i in Figure 2) does not contain any entry starting with $Z = \mathrm{H}(Y \parallel \nu_{I,\tilde{t}})$, VC concludes that SM_j did not correctly forward the packet to SM_i, meaning that SM_j is either malicious or faulty.

Low Complexity. As stated in Section II-A, our scheme introduces small modifications to the existing electrical infrastructure, which are mainly related to SMs.

Our scheme also introduces limited overhead on SMs, in terms of computation and storage. The computation overhead mainly is due to the execution of some cryptographic operations during the *Initialization* and *Forwarding* algorithms. In particular, the *Initialization* algorithm requires each SM to compute 3 public key encryptions, 2 HMAC, and 1 hashing; the *Forwarding* algorithm requires each SM to compute 1 public key decryption, 2 public key encryptions, 1 HMAC, and 1 hashing. However, as shown in [3], SMs can efficiently compute the basic cryptographic operations.

The verification scheme of Section III-B also introduces an additional storage overhead on SMs, which we can estimate as follows. Assume that the number of reachable SMs for each SM is constant; Thus, the average number of packets forwarded by every SM in each time-interval will be $h + 1$, and, therefore, in each time-interval, every SM will memorize, on average, $h+1$ log entries. As an example, assume that each SM reports its metering data every 10 minutes; assume $h = 4$ and a log entry of size 256 Bytes after encryption with RSA with the key of size 2048 bits. On a monthly basis, a SM will require an average of $(4+1) \times 256\ Bytes \times 6 \times 24 \times 30 = 5,529,600$ Bytes (~ 5.3 MBytes) of memory to store log's information. Thus, by even keeping the log file in memory for a year, the total amount of required memory will be ~ 64 MBytes which is still negligible.

V. Conclusions

In this paper, we presented the initial design of our privacy-preserving Advanced Smart metering Infrastructure (AMI), focusing on anonymous metering data reporting. Our solution adopts a collaborative multi-hop protocol for Smart Meters (SM), which allows them to anonymously transmit periodic metering data to a Utility Provider (UP). Our solution also allows a Verification Center (VC) to verify the functionality of each SM, by accessing its internal log.

This paper aims at presenting our initial research effort. Future work will focus on introducing additional features to our metering infrastructure, such as protocols for anonymous two-way communication between SMs and UP for transmitting query/response, aggregating metering data at the SM level, and anonymously compute the energy bill. We also aim at providing a fully-functional implementation prototype on real SM devices, in order to perform an extensive evaluation of the proposed solutions.

References

[1] "Smart Meters and Smart Meter Systems: A Metering Industry Perspective," Tech. Rep., 2011.

[2] G. Danezis, C. Fournet, M. Kohlweiss, and S. Zanella-Béguelin, "Smart meter aggregation via secret-sharing," in *SEGS '13*, 2013, pp. 75–80.

[3] B. Defend and K. Kursawe, "Implementation of privacy-friendly aggregation for the smart grid," in *SEGS '13*, 2013, pp. 65–74.

[4] T. Dimitriou and G. Karame, "Privacy-friendly tasking and trading of energy in smart grids," in *SAC '13*, 2013, pp. 652–659.

[5] C. Efthymiou and G. Kalogridis, "Smart grid privacy via anonymization of smart metering data," in *SmartGridComm '10*, 2010, pp. 238–243.

[6] Z. Erkin and G. Tsudik, "Private computation of spatial and temporal power consumption with smart meters," in *ACNS'12*, 2012, pp. 561–577.

[7] I. Kitagawa and S. Sekiguchi, "Technologies Supporting Smart Meter Networks," *FUJITSU Scientific and Technical Journal*, vol. 49, no. 3, pp. 307–312, 2013.

[8] K. Kursawe, G. Danezis, and M. Kohlweiss, "Privacy-friendly aggregation for the smart-grid," in *PETS'11*, 2011, pp. 175–191.

[9] M. Lisovich, D. K. Mulligan, S. B. Wicker *et al.*, "Inferring personal information from demand-response systems," *IEEE Security & Privacy*, vol. 8, no. 1, pp. 11–20, 2010.

[10] A. Molina-Markham, P. Shenoy, K. Fu, E. Cecchet, and D. Irwin, "Private memoirs of a smart meter," in *BuildSys '10*, 2010, pp. 61–66.

[11] K. Ohara, Y. Sakai, F. Yoshida, M. Iwamoto, and K. Ohta, "Privacy-preserving smart metering with verifiability for both billing and energy management," in *ASIAPKC '14*, 2014, pp. 23–32.

[12] M. K. Reiter and A. D. Rubin, "Crowds: Anonymity for web transactions," *ACM Trans. on Information and System Security*, vol. 1, no. 1, pp. 66–92, Nov. 1998.

[13] A. Rial and G. Danezis, "Privacy-preserving smart metering," in *WPES '11*, 2011, pp. 49–60.

[14] M. Steiner, G. Tsudik, and M. Waidner, "Key agreement in dynamic peer groups," *IEEE Trans. on Parallel and Distributed Systems*, vol. 11, no. 8, pp. 769–780, Aug 2000.