EPIC: Efficient Privacy-Preserving Scheme with E2E Data Integrity and Authenticity for AMI Networks

Ahmad Alsharif, *Member, IEEE,* Mahmoud Nabil, Samet Tonyali, Hawzhin Mohammed, Mohamed Mahmoud, *Member, IEEE,* and Kemal Akkaya, *Senior Member, IEEE,*

***Abstract*—In this paper, we propose EPIC, an efficient and privacy-preserving data collection scheme with E2E data integrity verification for AMI networks. Using efficient cryptographic operations, each meter should send a masked reading to the utility such that all the masks are canceled after aggregating all meters' masked readings, and thus the utility can only obtain an aggregated reading to preserve consumers' privacy. The utility can verify the aggregated reading integrity without accessing the individual readings to preserve privacy. It can also identify the attackers and compute electricity bills efficiently by using the fine-grained readings without violating privacy. Furthermore, EPIC can resist collusion attacks in which the utility colludes with a relay node to extract the meters' readings. A formal proof, probabilistic analysis are used to evaluate the security of EPIC, and ns-3 is used to implement EPIC and evaluate the network performance. In addition, we compare EPIC to existing data collection schemes in terms of overhead and security/privacy features.**

***Index Terms*—Smart Grid, AMI Networks, Privacy Preservation, Data Integrity, Collusion Resistance, and Dynamic Pricing.**

I. Introduction

The smart grid initiative aims to develop a clean, reliable, and efficient system. It extensively integrates information technology into the power grid [1]. One main component of the smart grid is the Advanced Metering Infrastructure (AMI) networks that connect smart meters (SMs) installed at consumers' side to the electric service provider (the utility). SMs should send fine-grained power consumption readings to the utility to perform real-time monitoring and energy management [2]. Moreover, the utility can reduce the power consumption during peak hours using dynamic pricing approach in which the electricity prices may change during the day to encourage consumers to reduce their power consumption.

However, the fine-grained power consumption readings can reveal sensitive information about the consumers' activities, such as the times consumers leave/return homes, as well as, the appliances they use since each appliance has a unique power consumption signature [3]–[5]. Privacy-preserving data aggregation is a promising technique to enable the utility to obtain an aggregated fine-grained reading from an AMI network without learning the individual readings to preserve the consumers' privacy. However, the existing schemes, such as [6]–[10], extensively use asymmetric-key cryptography in data aggregation, which typically involves large computation and communication overhead. They also do not address End-to-End (E2E) data integrity in which the utility can ensure that all the individual fine-grained readings are not altered during transmission and aggregation without accessing the individual readings to preserve privacy. Moreover, they do not address E2E authenticity in which the utility can ensure that the aggregated reading is computed using the fine-grained readings coming from intended consumers. Furthermore, generating electricity bills using the reported fine-grained readings based on dynamic prices is challenging since the utility should not have access to the fine-grained readings to preserve privacy, but these readings are needed to generate consumers' bills.

In this paper, we propose an **E**fficient **P**rivacy-preserving scheme with E2E data **I**ntegrity, authenticity and **C**ollusion-resistance for AMI networks, named "**EPIC**". The idea is that each SM selects a number of SMs in the network called "*proxies*" and efficiently computes shared pairwise secret masks with each proxy. Then, it should mask its fine-grained reading with all the masks shared with the proxies, such that all the masks are canceled after aggregating all meters' masked readings, and thus the utility can only obtain an aggregated reading to preserve consumers' privacy. EPIC can also resist collusion attacks in which the utility can collude with a relay meter to extract a meter's fine-grained readings because readings are masked by several secret masks shared with a number of different proxies. The number of the selected proxies controls the protection level against collusion attack. In addition, to ensure E2E data integrity and authenticity, a homomorphic hash and a hash MAC are computed on each masked reading. Then, hash MACs are aggregated while all the individual homomorphic hashes are forwarded to the utility. Using the individual homomorphic hashes and the aggregated MAC, the utility can ensure the data integrity of each individual fine-grained reading and the authenticity of each consumer in the network. Furthermore, the homomorphic hashes can be also used to enable the utility to generate dynamic electricity bills without accessing the individual readings to preserve privacy.

Our contributions can be summarized as follows.

A. Alsharif is with the Department of Computer Science, University of Central Arkansas, Conway, AR, 72035 USA and also with the department of Electrical & Computer Engineering, Tennessee Tech University, Cookeville, TN 38505 USA. E-mails: aalsharif@uca.edu

M. Nabil, H. Mohammed and M. Mahmoud are with the Department of Electrical & Computer Engineering, Tennessee Tech University, Cookeville, TN 38505 USA. E-mails: mnmahmoud42@students.tntech.edu, hmohammed42@students.tntech.edu, and mmahmoud@tntech.edu

S. Tonyali and K. Akkaya are with the Department of Electrical & Computer Engineering, Florida International University, Miami, FL 31174 USA. Emails: stony002@fiu.edu, kakkaya@fiu.edu

- *Efficient and collusion-resistant privacy-preserving power consumption collection.* EPIC uses secure and lightweight operations to efficiently mask the fine-grained readings and aggregate the masked readings to enable the utility to collect a fine-grained aggregated reading without leaking the consumers' sensitive information. It can also resist collusion attacks to reveal a meter's readings and allows the SMs to set their protection level.
- *E2E data integrity.* Since the meters' reading can be modified during the transmission to the utility, EPIC enables the utility to verify the integrity of the aggregated reading without accessing the individual readings to preserve consumers' privacy. It also enables the utility to identify the attackers who modify the readings.
- *E2E authenticity.* In EPIC, the utility can ensure that the aggregated reading was computed by the intended users in the network.
- *Dynamic pricing.* Using homomorphic hash properties, EPIC enables the utility to efficiently compute electricity bills based on dynamic pricing without violating consumers' privacy.

The results of a formal proof, probabilistic modeling, and analysis demonstrate that EPIC is secure. In addition, ns-3 is used to implement EPIC and evaluate the network performance. The results demonstrate that EPIC is efficient. We also compare EPIC to the existing data collection schemes in terms of overhead and security/privacy features.

A preliminary version of this paper appeared in [11]. The main difference between [11] and this paper are as follows. First, [11] did not address E2E data integrity, E2E authenticity, attacker identification, dynamic pricing, and details of key management and sharing secret masks offline and efficiently. This paper addresses all these challenges. Second, extensive analysis and simulation have been added to this paper. This includes a formal security proof, a comprehensive security analysis, probabilistic analysis of the collusion attacks and the proposed defense method, and updated ns-3 simulation results against similar existing schemes.

The remainder of the paper is organized as follows. Section II discusses the related works. The system models and preliminaries are presented in section III. The proposed masking and aggregation method is presented in section IV. The details of EPIC are given in section V. The security and privacy analysis is given in section VI, whereas the performance evaluation and experimental results are given in section VII. Finally, conclusions are drawn in section VIII.

II. Related Works

Several schemes have been proposed to collect power consumption readings in AMI networks and wireless sensor networks (WSNs) [6]–[10], [12]–[14]. In [6], Fan et al. use bilinear pairing with an aggregation method based on blind factors and solving the discrete log problem using Pollard's lambda method to obtain the aggregated reading and achieve collusion resistance. In [7], Lu et al. used homomorphic encryption to aggregate multi-dimensional data represented using a superincreasing sequence. Shen et al. in [8] proposed cube-data aggregation by using the Paillier cryptosystem and Horner's Rule. In [9], Li et al. proposed the use of two superincreasing sequences with the Paillier cryptosystem to achieve multi-subset data aggregation. In [10], Li et. al. used homomorphic-encryption-based aggregation scheme to send an aggregated reading to the utility. The utility should run anomaly detection system in every data collection round to detect data modification attack, but unlike EPIC, the system suffers from false positive and negatives.

TABLE I
Comparison between the proposed scheme and related schemes.

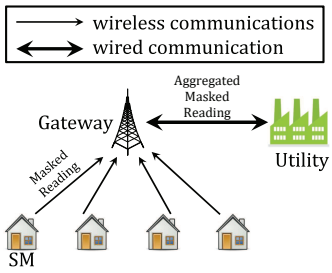
	EPIC	[6]	[7]	[8]	[9]	[10]
Privacy Preservation	√	√	√	√	√	√
Efficient Aggregation	√	NA	NA	NA	NA	NA
E2E Data Integrity & Attacker Identification	√	NA	NA	NA	NA	NA*
Collusion Resistance	√	√	NA	NA	√	NA
Support Dynamic Pricing	√	NA	NA	NA	NA	NA

NA: Not Addressed
*: Unlike EIPC that uses cryptography to detect data modification, [10] depends on anomaly detection to verify the data, which is vulnerable to false positives and negatives.

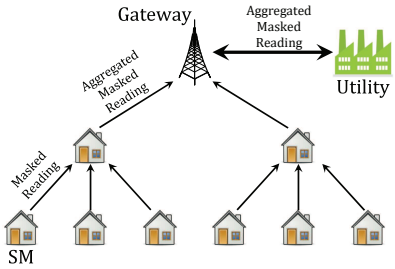
In [13] Garcia et al. combined Paillier's homomorphic encryption with additive secret sharing scheme to protect the scheme against collusion attacks. However, the encryption and aggregation complexities of [13] are $\mathcal{O}(n)$ and $\mathcal{O}(n^2)$ respectively while in EPIC they are $\mathcal{O}(1)$ and $\mathcal{O}(n)$. In [14], Li et. al. used homomorphic MAC and homomorphic hash functions to provide data integrity in WSNs against external attackers only with the assumption that all internal nodes are trusted. Compared to EPIC, homomorphic encryption based aggregation schemes such as [7]–[10], [12], [13] are inefficient as they require much larger size ciphertext and much more time for encryption, decryption, and aggregation than EPIC.

In [15], Knirsch et al. proposed the use of one-time masking for privacy-preserving data aggregation. Specifically, in each data collection round, the SMs are arranged in a ring-based topology to sequentially update the smart meter masks before masking each fine-grained reading. However, the proposed scheme has several limitations. First, [15] can only support single-hop model with a ring-topology communication used for online masks agreement, while EPIC can support both single- and multi-hop models with efficient and offline mask agreement. Second, in each data collection round in [15], all the ring SMs must communicate sequentially to ensure the correctness of masks updates before the actual reading reporting to the utility begins. This, therefore, increases the time required by the utility to collect the fine-grained readings and limits the network scalability.

Homomorphic linear authenticators (HLAs) [16] have been widely used to achieve data integrity for cloud applications [17], [18]. In cloud-based applications, each user breaks its data into several blocks, uses its private key to generate an authentication tag for each block and stores these blocks and authentication tags on a cloud server. For data retrieval, a verifier sends a random challenge to the server and then uses



(a) Single-hop model.



(b) Multi-hop model.

Fig. 1. The considered network models.

the server response along with the users' public keys to ensure data integrity. Therefore, in cloud-based applications, data is not relayed by other users, instead, the data modification attacks can be launched by the cloud server. Different from HLA-based schemes [17], [18], EPIC considers a different network and threat models in which data is relayed by other SMs in the AMI network who may launch data modification attacks. Unlike [17], [18], EPIC ensures data integrity by using the lightweight homomorphic hash along with an aggregated hash MAC to ensure data integrity as will be explained in subsection V-D and subsubsection VI-B1.

To sum up, we compare in Table I EPIC against similar schemes. To the best of our knowledge, EPIC is the first solution that aims to achieve efficiency, privacy preservation, hop-by-hop and E2E data integrity, authenticity and attackers identification, high resistance to collusion attacks, and dynamic pricing based billing simultaneously for both single- and multi-hop network models.

III. System Models and Preliminaries

A. Network Model

As shown in Fig. 1, the considered network model consists of the utility and service subscribers in a residential area. Each subscriber's house is equipped with a SM to report fine-grained power consumption readings to the utility every short time interval. SMs can communicate with the utility through a local collector, called gateway. As shown in the figure, EPIC can be used in single-hop or multi-hop network models. The SMs are connected via a wireless mesh network using Wi-Fi where each meter can act as a router to relay meters' packets to connect them to the gateway. The gateway can communicate to the utility through a wired link with low delay and high bandwidth. For the single-hop model, SMs send reading packets to the gateway which aggregates all the readings, create a new packet, and send it to the utility. For the multi-hop model, a virtual minimum spanning tree network topology that allows bottom-up aggregation is built. Then, leaf SMs send their readings packets to their parent SM which uses its reading and the packets received from children SMs to create a new packet and forwards it to the next parent SM. This should continue until the utility receives a reading packet.

B. Adversary Model

Attackers could be external adversaries $\mathcal{A}$, or internal network nodes, such as SMs, the gateway, or the utility. Attackers may attempt to invade the consumers' privacy to learn their power consumption patterns. They may also try to breach the data integrity by modifying other meters' data. In addition, $\mathcal{A}$ can eavesdrop on all the communications between the different parties to infer any sensitive information about consumers. $\mathcal{A}$ can also launch some active attacks such as packet replay and impersonation. Moreover, the attackers can work individually or collude to launch stronger attacks.

C. Preliminaries

1) Bilinear Pairing: Let $\mathbb{G}_1$ be an additive cyclic group, $\mathbb{G}_2$ be a multiplicative cyclic group of the same prime order q, and P be a generator of $\mathbb{G}_1$. A pairing $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ has the following properties.

- *Bilinearity*: $\hat{e}(aP, bQ) = \hat{e}(P, abQ) = \hat{e}(abP, Q) = \hat{e}(P, Q)^{ab} \in \mathbb{G}_2 \ \forall \ P, Q \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}_q^*$.
- *Non-degeneracy*: $\hat{e}(P, P) \neq 1_{\mathbb{G}_2}$. $P, Q \in \mathbb{G}_1$.

2) Homomorphic Hash Function: Let $\mathbb{G}$ be an additive cyclic group of prime order p and has d random generators $\{P_1, P_2, \ldots, P_d\} \in \mathbb{G}$. A homomorphic hashing on message $m = \{m_1, m_2, \ldots, m_d\}$ can be constructed as

$$\mathcal{H}(m) \stackrel{\text{def}}{=} \sum_{i=1}^{d} m_i P_i$$

Homomorphic hash function is collision resistant, where it is infeasible to find m_1 and m_2 such that $\mathcal{H}(m_1) = \mathcal{H}(m_2)$. In addition, homomorphic hash function is one way, where given $\mathcal{H}(m_1)$, it is infeasible to compute m_1. Homomorphic hash function also has the following property: $\mathcal{H}(m_1 + m_2) = \mathcal{H}(m_1) + \mathcal{H}(m_2)$. We refer to reference [19] for more details on homomorphic hash functions.

IV. Efficient and Collusion-Resistant Aggregation

In this section, we present a collusion-resistant and efficient data aggregation technique that is used in EPIC. We refer to Table II for the main notations and parameters that will be used in this paper.

A. Data Masking

Masked Readings. Fig. 2 illustrates the data masking approach used to protect consumer privacy and resist collusion attacks. First, SM_i chooses α proxies $\{\mathcal{P}_{i,1}, \ldots, \mathcal{P}_{i,\alpha}\}$ and shares a secret mask value $s_{i,j}^{(t_x)}$ with each proxy $\mathcal{P}_{i,j}$ to be used for reporting the reading of time slot t_x. As shown in

TABLE II
MAIN NOTATIONS.

Symbol	Meaning
SM_i	Smart Meter i
SM_{pi}	Parent of Smart Meter i
ℓ_i	Total number of meters in SM_i subtree
n_i	Number of direct children of SM_i
c	Children of SM_i $(1 \leqslant c \leqslant n_i)$
σ_i	Signature generated by SM_i
r_i	Fine-grained reading of SM_i
$s_{i,j}$	Secret mask shared between SM_i and $\mathcal{P}_{i,j}$
m_i	Masked reading of SM_i
M_i	Aggregated masked readings for SM_i subtree
h_i	Homomorphic hash on masked reading of SM_i
x_i ,Y_i	private and public key of SM_i
$K_{i,j}$	Symmetric key between node i and node j
$\text{HMAC}_{K_{i,u}}(h)$	A keyed hash function on h
$mac_{i,u}$	HMAC $_{K_{i,u}}(h_i)$
MAC_i	Aggregated MAC computed by SM_i
$\mathcal{P}_{i,j}$	Proxy j of SM_i
α_i	Number of proxies selected by SM_i
β_i	Number of nodes that selected SM_i as a proxy
λ_i	Total number of SM_i proxies, $\lambda_i = \alpha_i + \beta_i$
$q, \mathbb{G}_1, \mathbb{G}_2, P, \hat{e}$	Public parameters of bilinear pairing
$\{P_1, ..., P_d\}$,$\mathbb{G}$,$\mathcal{H}$	Public parameters of homomorphic hash

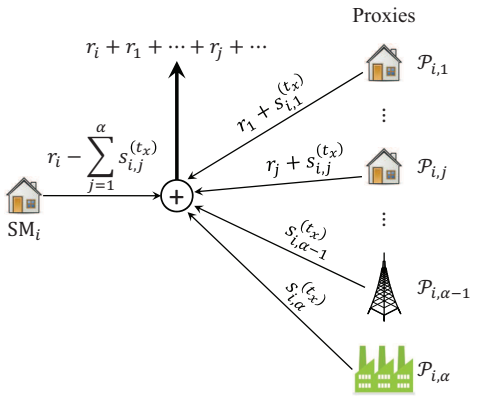


Fig. 2. SM_i masks its reading with secrets shared with proxies, and each proxy removes one secret from the mask by adding the mask to its reading.

the figure, each proxy $\mathcal{P}_{i,j}$ should add the mask $s_{i,j}^{(t_x)}$ to its fine-grained reading r_j, whereas SM_i masks its fine-grained reading r_i by subtracting the summation of all shared masks with its proxies. After aggregating all masked readings sent by SM_i and its proxies, the masks added by all proxies cancel the mask used by SM_i. *If all the SMs follow this masking technique, the resultant value after the aggregation should be the summation of all fine-grained readings.* Masks are used to achieve privacy preservation and collusion resistance as will be explained in section VI.

Mask Calculation. In EPIC, masks can be computed *offline* and *efficiently* as follows: $s_{i,j}^{(t_x)} = \text{HMAC}_{K_{i,j}^{(s)}}(Y_i, Y_j, day, t_x)$, where $\text{HMAC}_{K_{i,j}^{s}}()$ is a keyed hash function and $K_{i,j}^{(s)}$ is a short-time symmetric key that can be computed by the procedure that is explained in the next subsection, Y_i and Y_j are the public keys of the meter SM_i and the proxy $\mathcal{P}_{i,j}$ respectively, day is a unique day's date, and t_x is a sequence number of the readings of one day. Obviously, no other entity can derive the masks because it does not know the shared key.

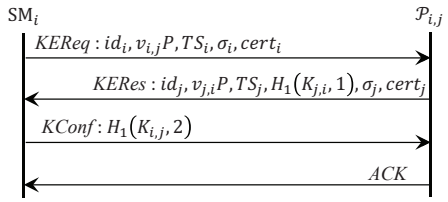


Fig. 3. Long-term key establishment procedure.

B. Efficient Key Agreement Procedure

Each meter needs to share a key with each proxy to efficiently calculate the secret mask. In this subsection, we describe two key agreement procedures to establish long-term and short-term keys. Initially, a long-term symmetric seed key $K_{i,j}$, shared between SM_i and its proxy $\mathcal{P}_{i,j}$ should be established and refreshed over a long period. Then, a short-term key $K_{i,j}^{(s)}$ is efficiently computed using the long-term key.

1) Long-term seed key agreement: To share a long-term seed key $K_{i,j}$ with each proxy $\mathcal{P}_{i,j}$, SM_i chooses a random element $v_{i,j} \in \mathbb{Z}_q^*$ and composes a key establishment request (*KEReq*) packet as shown in Fig. 3. The packet contains id_i, $v_{i,j}P$, TS_i, σ_i and $cert_i$, where, $v_{i,j}P$ is the random element $v_{i,j}$ multiplied by the generator P of the additive group $\mathbb{G}_1$, σ_i is a signature and $\sigma_i = x_i H_2(id_i, v_{i,j}P, TS_i)$, H_2 is a hash function defined as $H_2{:}\{0,1\}^* \rightarrow \mathbb{G}_1$, and $cert_i$ is the certificate of SM_i. Finally, SM_i sends the *KEReq* packet to its proxies. Each proxy $\mathcal{P}_{i,j}$ verifies that the packet is not stale by checking the timestamp (TS_i) to thwart replay attacks. Then, $\mathcal{P}_{i,j}$ uses SM_i's public key, $Y_i = x_i P$, to verify the signature σ_i by checking $\hat{e}(\sigma_i, P) \stackrel{?}{=} \hat{e}(H_2(id_i, v_{i,j}P, TS_i), Y_i)$. The signature verification proof is as follows:

$$\begin{aligned}\hat{e}(\sigma_i, P) &= \hat{e}(x_i H_2(id_i, v_{i,j}P, TS_i), P) \\ &= \hat{e}(H_2(id_i, v_{i,j}P, TS_i), x_i P) \\ &= \hat{e}(H_2(id_i, v_{i,j}P, TS_i), Y_i))\end{aligned}$$

If the signature is successfully verified, each proxy $\mathcal{P}_{i,j}$ chooses a random element $v_{j,i} \in \mathbb{Z}_q^*$ and computes $v_{j,i}P$. Moreover, $\mathcal{P}_{i,j}$ calculates the long-term seed key $K_{j,i} = v_{j,i}v_{i,j}P$. Finally, it sends the key establishment response (*KERes*) packet to SM_i. As shown in the figure, the packet has id_j, $v_{j,i}P$, TS_j, $H_1(K_{j,i}, 1)$, σ_j, and $cert_j$, where $\sigma_j = x_j H_2(id_j, v_{j,i}P, TS_j, H_1(K_{j,i}, 1))$, $H_1()$ is a hash function (such as SHA-1), and $H_1(K_{j,i}, 1)$ is used for key confirmation. When SM_i receives the packet, it checks the timestamp and verifies the signature similar to the verification process done by the proxy. Then, it computes the long-term seed key $K_{i,j} = v_{i,j}v_{j,i}P = K_{j,i}$. Finally, it sends a key confirmation packet (*KConf*) to $\mathcal{P}_{i,j}$ so that $\mathcal{P}_{i,j}$ knows that SM_i has successfully computed the long-term key.

2) Short-term key computation: The long-term key $K_{i,j}$ is used as a seed for the computation of the short-term keys. First, SM_i and $\mathcal{P}_{i,j}$ use the seed key to compute bi-directional (forward and backward) hash chains as shown in Fig. 4. For the forward chain, both SM_i and $\mathcal{P}_{i,j}$ compute $F_1 = H_1(K_{i,j}, TS, 1)$ where TS is the timestamp. Then, all the elements of the forward hash chain are computed using $F_a = H_1(F_{a-1})$ for $2 \leq a \leq T$, where T is the number of short-term keys that need to be stored. Similarly,

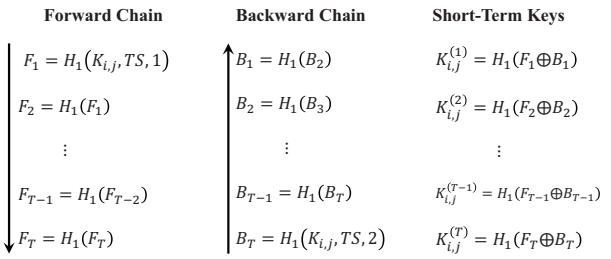


Fig. 4. Short-term keys computation.

the backward hash chain is computed by first computing $B_T = H_1(K_{i,j}, TS, 2)$, then the elements of the backward hash chain are computed as $B_b = H_1(B_{b+1})$ for $1 \leq b \leq T-1$. Finally, the short-term key is computed by XORing an element from the forward chain with the corresponding element from the backward chain and hashing the result as shown in the figure. Each short-term key should be used for a short time and after using all the T keys, the meters can compute a new set of keys using an updated TS. In this way, the smart meters do not need to compute and store a large number of keys in each round. Short-term keys are computed daily and each key should be used to compute a set of masks. After using the long-term key for a certain time, SM_i and $\mathcal{P}_{i,j}$ should establish a new long-term seed key, and derive a new set of short-term keys.

V. The Proposed Scheme

In this section we give the details of EPIC starting by system setup. Then, we show how SMs report and aggregate power consumption readings. Finally, we illustrate how the utility can recover the aggregated reading, verify its integrity, users' authenticity, and generate electricity bills for each user.

A. System Setup

An offline trusted authority (TA) should bootstrap the system as follows. First, the TA generates the bilinear mapping parameters $(q, \mathbb{G}_1, \mathbb{G}_2, P, \hat{e})$. It also chooses three different hash functions. H_1 is a regular hash function such as SHA-1, $H_2 : \{0,1\}^* \rightarrow \mathbb{G}_1$, and $\mathcal{H}$ is a homomorphic hash function with the generators $\{P_1, P_2, ..., P_d\} \in \mathbb{G}$. Furthermore, a keyed hash function $mac_{i,u} \leftarrow HMAC_{K_{i,u}}(h)$ is selected, where $mac_{i,u}$ is the HMAC on h using the symmetric key $K_{i,u}$. Then, the TA publishes the system public parameters as $pubs = \{q, P, P_1, P_2, ..., P_d, \mathcal{H}, H_1, H_2, HMAC_K, \hat{e}\}$.

In addition, every SM_i chooses a secret key $x_i \in \mathbb{Z}_q^*$ and computes the corresponding public key $Y_i = x_i P$. It should also obtain a certificate for the public key from a certificate authority. Finally, each SM_i should select several proxies, assuming that SM_i selects α_i proxies and be selected by β_i meters to act as a proxy for them, i.e. the total number of proxies for SM_i is $\lambda_i = \alpha_i + \beta_i$. Each SM and its proxies should establish the long-term seed key, derive the short-term keys, and compute the shared masks as explained in Section IV.

B. Leaf Meters: Report Generation

Each leaf meter SM_c generates a power consumption report by executing the following steps.

1) Masks its reading r_c to obtain a masked reading m_c.

$$m_c = r_c - \sum_{j=1}^{\alpha_c} s_{c,j} + \sum_{j=1}^{\beta_c} s_{j,c} \quad (1)$$

2) Hashes its masked reading m_c using homomorphic hash function $\mathcal{H}(\ )$ to get h_c.

$$h_c = \mathcal{H}(m_c) \equiv \mathcal{H}(r_c) - \sum_{j=1}^{\alpha_c} \mathcal{H}(s_{c,j}) + \sum_{j=1}^{\beta_c} \mathcal{H}(s_{j,c}) \quad (2)$$

3) Computes HMAC on h_c using the shared key with the utility as $mac_{c,u} = \text{HMAC}_{K_{c,u}}(h_c)$.
4) Generates a signature $\sigma_c = x_c H_2(m_c, mac_{c,u}, TS)$.

Finally, SM_c transmits to its parent SM_i the following tuple

$$m_c, TS, h_c, mac_{c,u}, \sigma_c \quad (3)$$

C. Non-leaf Nodes: Data Verification and Report Generation

The operations done by non-leaf meters SM_i and the gateway can be divided into two phases. In the first phase, SM_i receives n_i messages from its children and verifies the authenticity and integrity of the received messages. In the second phase, SM_i create a new message to be transmitted to the next parent. These two phases should be executed at each non-leaf node until the aggregated masked reading reaches the utility. The details of the two phases are as follows.

Phase 1. SM_i receives n_i messages from each child meter SM_c $(1 \leqslant c \leqslant n_i)$. If the child is a leaf-node, its message has this format $(m_c, TS, h_c, mac_{c,u}, \sigma_c)$ while if the child is a non-leaf node, the message has the following format $(M_c, TS, h_1, h_2, \ldots, h_{\ell_c}, MAC_c, \sigma_c)$ where M_c and MAC_c are aggregated masked reading and aggregated MAC computed by the non-leaf chiled SM_c as defined in Table II. Also, the message contains the hashes of the masked readings of the sub-tree nodes of child SM_c. SM_i should perform the following verifications.

1) Perform a batch verification for the received signatures

$$\hat{e}\left(\sum_{c=1}^{n_i} \sigma_c, P\right) \stackrel{?}{=} \prod_{c=1}^{n_i} \hat{e}\big(H_2(M_c, MAC_c, TS), Y_c\big) \quad (4)$$

2) Perform a batch verification for all the received hashes by checking

$$\mathcal{H}\left(\sum_{c=1}^{n_i} M_c\right) \stackrel{?}{=} \sum_{c=1}^{n_i} \sum_{j=1}^{\ell_c} h_j \quad (5)$$

If this verification passes, SM_i moves to the next step, otherwise, data modification attack is detected and SM_i can identify the attacker by applying divide-and-conquer verification recursively until the attacker is identified.

3) Store the latest received tuple (M_c, MAC_c, σ_c) from every child to help the utility to identify the attacker in case that the utility detects data modification attack, as will be explained in the next subsection.

Phase 2. In this phase, SM_i should execute the following steps before sending a reading packet to its parent.

1) Masks its fine-grained reading r_i to obtain its own masked reading m_i

$$m_i = r_i - \sum_{j=1}^{\alpha_i} s_{i,j} + \sum_{j=1}^{\beta_i} s_{j,i} \quad (6)$$

2) Aggregates its masked reading m_i with the masked readings received from its children meters to generate an aggregated masked readings M_i as

$$M_i = m_i + \sum_{c=1}^{n_i} M_c \quad (7)$$

3) Hashes its masked reading m_i using homomorphic hash function to get h_i as

$$h_i = \mathcal{H}(m_i) \equiv \mathcal{H}(r_i) - \sum_{j=1}^{\alpha_i} \mathcal{H}(s_{i,j}) + \sum_{j=1}^{\beta_i} \mathcal{H}(s_{j,i}) \quad (8)$$

4) Computes HMAC on h_i with the shared key with the utility as $mac_{i,u} = \text{HMAC}_{K_{i,u}}(h_i)$.
5) Aggregates its MAC with the received aggregated MACs using XOR operations to obtain $MAC_i = mac_{i,u} \oplus (\bigoplus_{c=1}^{n_i} MAC_c)$.
6) Generates a signature $\sigma_i = x_i H_2(M_i, MAC_i, TS)$.

Finally SM_i sends to its parent SM_{pi} the following tuple

$$M_i, TS, h_1, h_2, \ldots, h_{\ell_i}, MAC_i, \sigma_i \quad (9)$$

This process of verification and aggregation proceeds in a bottom-up manner to the utility.

D. Utility: Aggregated Reading Recovery, Data Integrity Verification and Billing

Data recovery and verification. The utility receives $\left(M_{gw}, TS, h_1, h_2, \ldots, h_{\ell_{gw}}, MAC_{gw}, \sigma_{gw}\right)$ from the gateway. The utility first verifies σ_{gw}, homomorphic hashes, and TS, as described in *phase 1* of subsection V-C. Then, it verifies the aggregated MAC_{gw} as follows.

1) Calculates all the individual MACs from the received hashes $\left\{mac'_{u,j} = HMAC_{K_{u,j}}(h_j),\ \forall j \in \{1..\ell_{gw}\}\right\}$,
2) Calculates the aggregated MAC $MAC'_u = \bigoplus_{j=1}^{\ell_g} mac'_{u,j}$.
3) Compares the calculated MAC with received MAC.

$$MAC'_u \stackrel{?}{=} MAC_{gw} \quad (10)$$

If the verification passes, the utility can recover the aggregated reading of all SMs by removing its masks from M_{gw} by

$$M_{gw} + \sum_{j=1}^{\beta_u} s_{j,i} = \sum_{i=1}^{n} r_i \quad (11)$$

where n is the total number of meters in the AMI network.

Attacker identification. If a smart meter SM_i modifies both the aggregated masked reading and a homomorphic hash of any child in its subtree, i.e., transmits M'_c and h'_c instead of M_c and h_c to bypass its parent verification done in Equation 5, then the utility verification done in Equation 10 fails because MAC_c was computed by SM_c on h_c not h'_c. In this case, data modification attack is detected and the utility suspects all non-leaf SMs since any non-leaf SM can launch this attack. Therefore, the utility runs the following verifications in a bottom-up manner, i.e., starting from the first non-leaf nodes up to the last non-leaf node which is the gateway, until the attacker is identified.

In order to identify the attacker, the utility should retrieve SM_i children reports $(M_c, MAC_c, \sigma_c)\ 1 \le c \le n_i$ from SM_i and (M_i, MAC_i, σ_i) from SM_{pi}, which is the parent of SM_i. Then, the utility check if

$$\hat{e}\left(\sum_{c=1}^{n_i} \sigma_c, P\right) \stackrel{?}{=} \prod_{c=1}^{n_i} \hat{e}\left(H_2\left(M_c, MAC_c, TS\right), Y_c\right) \quad (12)$$

If SM_i can provide valid signatures from its children, it can pass the verification done in equation 12, otherwise, SM_i is identified as an attacker. The attacking SM can pass this verification iff he sends the correct M_c not M'_c, however, it will be identified by the next verification process.

If the verification in equation 12 passes for all non-leaf nodes, then the utility should to check the correctness of the messages sent by each meter SM_i. First, the utility extracts its individual masked reading from the aggregated masked reading M_i using

$$m'_i = M_i - \sum_{i=c}^{n_i} M_c$$

Then, it re-calculates $mac_{i,u}$ from the verified masked readings as

$$mac'_{i,u} = \text{HMAC}_{K_{i,u}}\left(\mathcal{H}\left(m'_i\right)\right)$$

After that, the utility re-calculates $mac_{i,u}$ from the verified aggregated MACs as

$$mac''_{i,u} = MAC_i \oplus \left(\bigoplus_{c=1}^{n_i} MAC_c\right)$$

Finally the utility checks if

$$mac'_{i,u} \stackrel{?}{=} mac''_{i,u} \quad (13)$$

If the verification fails, SM_i is identified as an attacker. This process continues in a bottom-up manner until the attacker is identified. The attacker cannot pass this check because he cannot compute a valid mac value for the modified packet. This is because the computation of a valid mac value requires the knowledge of the shared key between the victim meter, SM_c, and the utility. Therefore, EPIC can ensure E2E data integrity and authenticity without accessing the fine-grained readings to preserve consumers' privacy.

Dynamic-pricing-based billing. For dynamic pricing, the utility can divide the day into periods with different electricity prices. Assuming that the meters should report w power consumption readings during each billing period, Table III gives the w masked readings generated by n meters, where each column represents the masked readings sent in one time slot, while each row represents all the masked readings sent by each meter during the billing period (i.e., w readings). As explained earlier, the reading r_i of SM_i at a time slot t_x is masked using the mask $\sum_{j=1}^{\alpha_i} s_{i,j}^{(t_x)} - \sum_{j=1}^{\beta_i} s_{j,i}^{(t_x)}$ to produce the masked reading $m_i = r_i + \sum_{j=1}^{\alpha_i} s_{i,j}^{(t_x)} - \sum_{j=1}^{\beta_i} s_{j,i}^{(t_x)}$. The

TABLE III
ADDITION OF w MASKED REPORTS OF EACH METER TO OBTAIN THE TOTAL POWER CONSUMPTION DURING BILLING PERIOD.

	t_1	...	t_{w-1}	t_w	***Billing Period Consumption***
$\mathbf{SM}_1$	$r_1^{(1)} + \sum_{j=1}^{\alpha_1} s_{1,j}^{(1)} - \sum_{j=1}^{\beta_1} s_{j,1}^{(1)}$	...	$r_1^{(w-1)} + \sum_{j=1}^{\alpha_1} s_{1,j}^{(w-1)} - \sum_{j=1}^{\beta_1} s_{j,1}^{(w-1)}$	$r_1^{(w)} + s_{1,u}^{(b)} - \sum_{k=1}^{w-1}(\sum_{j=1}^{\alpha_1} s_{1,j}^{(k)} - \sum_{j=1}^{\beta_1} s_{j,1}^{(k)})$	$s_{1,u}^{(b)} + \sum_{k=1}^{w} r_1^{(k)}$
$\vdots$	$\vdots$		$\vdots$	$\vdots$	$\vdots$
$\mathbf{SM}_n$	$r_n^{(1)} + \sum_{j=1}^{\alpha_n} s_{n,j}^{(1)} - \sum_{j=1}^{\beta_n} s_{j,n}^{(1)}$	...	$r_n^{(w-1)} + \sum_{j=1}^{\alpha_n} s_{n,j}^{(w-1)} - \sum_{j=1}^{\beta_n} s_{j,n}^{(w-1)}$	$r_n^{(w)} + s_{n,u}^{(b)} - \sum_{k=1}^{w-1}(\sum_{j=1}^{\alpha_n} s_{n,j}^{(k)} - \sum_{j=1}^{\beta_n} s_{j,n}^{(k)})$	$s_{n,u}^{(b)} + \sum_{k=1}^{w} r_n^{(k)}$
gw	$\sum_{j=1}^{\alpha_{gw}} s_{gw,j}^{(1)} - \sum_{j=1}^{\beta_{gw}} s_{j,gw}^{(1)}$	...	$\sum_{j=1}^{\alpha_{gw}} s_{gw,j}^{(w-1)} - \sum_{j=1}^{\beta_{gw}} s_{j,gw}^{(w-1)}$	$-\sum_{k=1}^{w-1}(\sum_{j=1}^{\alpha_{gw}} s_{gw,j}^{(k)} - \sum_{j=1}^{\beta_{gw}} s_{j,gw}^{(k)})$	
u	$\sum_{j=1}^{\alpha_u} s_{u,j}^{(1)} - \sum_{j=1}^{\beta_u} s_{j,u}^{(1)}$	...	$\sum_{j=1}^{\alpha_u} s_{u,j}^{(w-1)} - \sum_{j=1}^{\beta_u} s_{j,u}^{(w-1)}$	$-\sum_{k=1}^{w-1}(\sum_{j=1}^{\alpha_u} s_{u,j}^{(k)} - \sum_{j=1}^{\beta_u} s_{j,u}^{(k)})$	
Total	$\sum_{i=1}^{n} r_i^{(1)}$	...	$\sum_{i=1}^{n} r_i^{(w-1)}$	$\sum_{i=1}^{n} s_{i,u}^{(b)} + \sum_{i=1}^{n} r_i^{(w)}$	

masks should be computed in such a way that the summation of all the masks used during billing period is zero. This can be done as follows. At the end of each billing period (report at t_w), the mask SM_i should use is equal to the negative summation of all the previous $w-1$ masks plus a billing mask, $s_{i,u}^{(b)}$, shared between the meter and the utility, i.e., the mask used in the last reading of a billing period is

$$s_{i,u}^{(b)} - \sum_{k=1}^{w-1} \left(\sum_{j=1}^{\alpha_i} s_{i,j}^{(t_k)} - \sum_{j=1}^{\beta_i} s_{j,i}^{(t_k)} \right)$$

so that the summation of all masked readings of SM_i gives the total power consumed by SM_i plus the billing mask, i.e.,

$$\sum_{k=1}^{w} m_i^{(k)} = \sum_{k=1}^{w} r_i^{(k)} + s_{i,u}^{(b)}$$

The utility should compute $\sum_{k=1}^{w} r_i^{(k)}$ to bill SM_i. It can use the homomorphic hash property $\mathcal{H}(m_1+m_2) = \mathcal{H}(m_1) + \mathcal{H}(m_2)$ to compute $\sum_{k=1}^{w} r_i^{(k)}$ as follows. First, the utility should add all the w homomorphic hashes sent by SM_i in the billing period to obtain It is clear that only the utility can remove $\mathcal{H}\left(s_{i,u}^{(b)}\right)$ and hence only the utility can obtain $\mathcal{H}\left(\sum_{k=1}^{w} r_i^{(k)}\right)$. Since the range of the readings is small, the utility can build a look-up table and obtain the total power consumption of SM_i, $\sum_{k=1}^{w} r_i^{(k)}$, from $\mathcal{H}\left(\sum_{k=1}^{w} r_i^{(k)}\right)$. It should be noted that, it is easy to obtain $\sum_{k=1}^{w} r_i^{(k)}$ from $\mathcal{H}\left(\sum_{k=1}^{w} r_i^{(k)}\right)$ since all the masks are canceled and the total consumption of a billing period is not a large number, but it is extremely hard to obtain m_i from h_i since the masks can make m_i a large number. Knowing the power consumption of SM_i during the billing period does not degrade consumers' privacy because the time period is long enough to prevent sensitive data leakage [20].

VI. SECURITY AND PRIVACY ANALYSIS

A. Privacy Analysis

1) Singular Attacks: An adversary, $\mathcal{A}$, can eavesdrop on all the communications of all the network nodes and can obtain the individual masked readings of a leaf SM. However, based on Equation 1, $\mathcal{A}$ must know all the λ_c masks, $\lambda_c = \alpha_c + \beta_c$, shared between the leaf meter, SM_c, and its proxies to be able to extract the meter's fine-grained reading. Since no entity can compute the correct masks except SM_c and its proxies, as explained in subsection IV-B, $\mathcal{A}$ cannot obtain the meters' fine-grained readings.

In the following, we present a formal security proof to show that the masking technique used in EPIC is semantic secure against chosen-plaintext attacks even if only one mask value is used to mask the fine-grained reading.

Theorem 1. *The masking scheme is semantically secure against chosen-plaintext attacks under the pseudorandom function (PRF) assumption for HMAC.*

Proof: The theorem proof as a game is constructed as follows.

- *Initialization*: The challenger $\mathcal{C}$ is initiated with a set of one-time secret masks generated as explained in subsection IV-A using the HMAC function which is used as a PRF [21].
- *Challenge*: The adversary $\mathcal{A}$ outputs two fine-grained readings r_0 and r_1 to $\mathcal{C}$. $\mathcal{C}$ chooses a random bit $b \in \{0,1\}$ and responds with a ciphertext $m_b = Enc(r_b) = r_b + s$, where s is the one-time secret mask and $s \gg r_b$.
- *Guess*: The adversary $\mathcal{A}$ responds with $b' \in \{0,1\}$ as a guess for b. The advantage of the adversary against the masking in the above game can be defined as

$$\text{Adv}_{\mathcal{A}} = \left| \Pr[b' = b] - \frac{1}{2} \right|$$

Because $s \gg r_b$, s is generated by a PRF and used only one time, the advantage $\text{Adv}_{\mathcal{A}}$ in this case becomes

$$\text{Adv}_{\mathcal{A}} = \left| \frac{1}{2} - \frac{1}{2} \right| = 0$$

Therefore, the masking scheme is semantically secure and no adversary can extract the fine-grained reading.

■

In addition, each mask value is used for only one reporting period to ensure that the masked readings look different even if the leaf meter reports the same fine-grained reading at different time slots. Therefore, given two consecutive reports of a meter, $\mathcal{A}$ can not learn if the power consumption has changed or not.

2) Collusion attacks: Unlike the singular attacks launched by a single adversary, we consider in the following a stronger attack in which the adversary can collude with other nodes in the AMI networks.

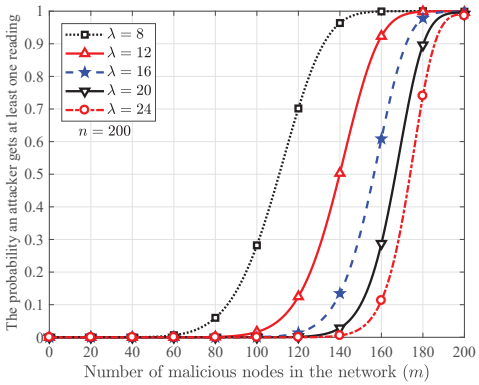


Fig. 5. The probability an attacker gets at least one reading for $n = 200$.

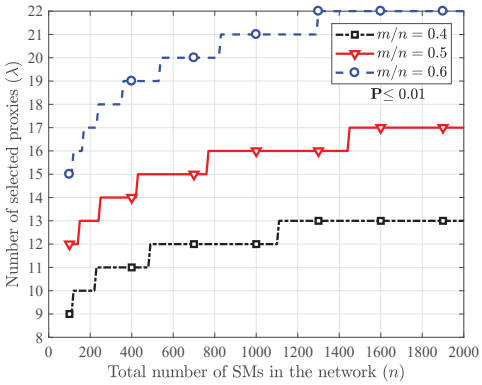


Fig. 6. Number of proxies vs. network size for $\mathbb{P} \leq 0.01$.

In EPIC, the fine-grained reading of each meter is protected by λ secret masks shared with λ proxies. Therefore, for an attacker to compute the fine-grained reading of a victim meter, *the attacker must collude at least with all the victim's proxies*. In particular, the attacker can try to recover the fine-grained reading of a victim meter from either its masked reading or its homomorphic hash.

To recover the fine-grained reading of a victim leaf meter from its masked reading, *the attacker must collude with all the victim's proxies* to obtain all secret masks and use them to get the fine-grained reading from the masked reading given in Equation 1. If the victim is a non-leaf meter, based on Equation 7, *the attacker must collude with both the victim's direct children and proxies*.

On the other hand, to recover the fine-grained reading of a victim meter from its homomorphic hash, based on Equation 2 or 8, *the attacker must collude with all the victim's proxies* to remove the hashes of the secret masks and obtain the $\mathcal{H}(r_i)$ from $\mathcal{H}(m_i)$. Since the value of r_i is a small number, the attacker can build a look-up table and recover r_i from $\mathcal{H}(r_i)$.

In all the previous attack scenarios, *the protection level against collusion attack is determined by the number of selected proxies* λ. Therefore, in the following, we model an attack and investigate how a proper value for λ can ensure a satisfactory protection level against collusion attack.

Consider that each SM selects λ proxies, the network has n nodes, including SMs, the gateway, and the utility, and the network has m malicious nodes that collude with the attacker. The probability that a SM selects all the λ proxies from the m malicious nodes follows the hypergeometric probability distribution and is given by

$$\frac{{}^{m}C_{\lambda}}{{}^{(n+1)}C_{\lambda}}$$

Then, the probability that a meter is secure against collusion attack is

$$1 - \frac{{}^{m}C_{\lambda}}{{}^{(n+1)}C_{\lambda}}$$

Let $\mathbb{P}$ be the probability that the attacker can recover at least the readings of any SM in the $(n-m)$ benign meters. $\mathbb{P}$ can be expressed as

$$\mathbb{P} = 1 - \prod_{i=1}^{n-m} \left(1 - \frac{{}^{m}C_{\lambda}}{{}^{(n+1)}C_{\lambda}}\right)$$

To assess how hard for the attackers to launch successful collusion attack in EPIC, Fig. 5 gives $\mathbb{P}$ versus m at different cases of λ for $n = 200$ SMs. As shown in the figure, if each SM selects $\lambda = 8$ proxies and 60 SMs colludes with the attacker, the probability that the attacker can obtain at least one meter's readings is almost zero. The attacker needs to collude with 80 SMs, 40% of the SMs in the network, so that the probability he can get at least one meter's readings becomes 0.06. If each SM increases the level of protection against collusion attacks by adding 4 more proxies, i.e. increasing λ from 8 to 12, the SMs are almost secure when the attacker colludes with 80 SMs. In this case, the attacker needs to collude with 120 SMs, 60% of the network, so that $\mathbb{P}$ becomes 0.12. We can conclude that, the increase of the number of proxies (λ) can make collusion attack harder to succeed.

To illustrate how many proxies should be selected by each SM to be secure against collusion attacks, Fig. 6 shows λ vs. n such that $\mathbb{P} \leq 0.01$. We define m/n as the ratio of malicious nodes in the network. A SM can select a proper number of proxies to be secure against collusion attack based on the network size and a risk assessment for the number of potential malicious meters in the network. For example, for a network with 100 SMs and 40 SMs of them are malicious, a SM can be secured by selecting 9 proxies, whereas, if the network size increases to 2,000 SMs and 800 of them are malicious, i.e., same m/n ratio, the SM should increase the number of proxies from 9 to 12 to ensure that the probability of successful collusion attack is less than 0.01. This indicates that although the number of SMs significantly increases from 100 to 2,000, a slight increase in the number of proxies is needed to secure the meters against collusion attacks. Moreover, in an extreme case in which $m/n = 0.6$ and $n = 2,000$, 22 proxies are needed to ensure that $P \leq 0.01$. We can conclude from this analysis that SMs can control the protection level against collusion attacks by selecting a proper number of proxies, and the ratio of proxies to the network size (λ/n) is small to achieve a satisfactory protection against collusion attacks.

B. Security Analysis

1) Data integrity: If an external adversary $\mathcal{A}$ manipulates the transmitted messages between a child meter and its parent, the attack can be easily detected by the parent because it can verify the integrity of the received messages by verifying

the received signature. Forging a signature or modifying a valid signature is infeasible without knowing the private key of the child meter. In addition, $\mathcal{A}$ may record valid packets exchanged between a meter and its parent (such as the packets given in (3) and (9) and replay them at later time to disrupt the reading collection scheme. Since packets have timestamps, the stale packets can be easily identified and dropped. If $\mathcal{A}$ tries to change the timestamp so that the packet looks fresh, $\mathcal{A}$ needs to know the private key of the victim meter to compute a valid signature on the packet of the modified timestamp.

Comparing to external attackers, internal attacks can launch stronger attacks. In particular, they may breach the data integrity by launching three different attacks: (1) modification of a child's homomorphic hash only; (2) modification of a child's masked reading only; and (3) modification of both child's homomorphic hash and masked reading. Consider SM_c be the victim child, SM_i be the malicious parent, and SM_{pi} be the parent of SM_i. SM_{pi} can either detect the attack of SM_i or help the utility to detect the attack. The first two attacks can be detected by SM_{pi} because the batch verification process of the individual homomorphic hash values done by SM_{pi} (given in Equation 5) fails. For the third attack, modification of both M_c and h_c, the utility can detect the attack from the aggregated MAC verification done in Equation 10. To identify the malicious SM_i, the utility should use the procedure explained in subsection V-D. Therefore, EPIC can ensure E2E data integrity without accessing the fine-grained readings to preserve consumers' privacy.

2) E2E users' Authenticity: EPIC achieves hop-by-hop authentication in which each parent meter can authenticate the child meters because each packet is signed by the child meter. Therefore, it is infeasible for $\mathcal{A}$ to impersonate meters by sending packets under their names, and thus parent meters accept only messages from authenticated children. In addition, EPIC can also ensure E2E authenticity since the verification process done by the utility in subsection V-D requires the use of symmetric keys shared between the utility and each legitimate user in the AMI network. Therefore, successful verification process means that the received aggregated reading was computed from the intended system users.

3) Key agreement: Long-term key agreement. The security of the key computation, shown in Fig. 3, relies on the hardness of the Discrete-Logarithmic Problem (DLP). If $\mathcal{A}$ eavesdrops on the communication between SM_i and $\mathcal{P}_{i,j}$ given in Fig. 3, he can obtain $v_{i,j}P$ and $v_{j,i}P$. However, given $v_{i,j}P$ and P, it is computationally infeasible to obtain $v_{i,j}$. Therefore, only the involved parties can compute the keys.

Short-term key agreement For backward and forward secrecy, as shown in Fig. 4, given the current short-term key, $\mathcal{A}$ can compute neither the past keys nor the future keys. Assuming an attacker could obtain a short-term key $K_{i,j}^{(s)} = H_1(F_s \oplus B_s)$, it is computationally infeasible to extract $F_s \oplus B_s$ from $K_{i,j}^{(s)}$ because the hash function is irreversible.

VII. Performance Evaluation

In this section, we first evaluate EPIC in terms of the communication and computation overheads for the single-hop model, then, we present our ns-3 experiment results to assess the network performance for the single- and multi-hop models.

TABLE IV
Computational times and sizes for cryptographic operations.

	Cryptographic Operation	Time
T_1	Pairing $e(P_1, P_2)$	1.025 ms
T_2	$g_1 \times g_2 \in \mathbb{G}_2$ $(512bit)$	1.22 μs
T_3	$\sigma_1 + \sigma_2 \in \mathbb{G}_1$ $(512bit)$	4.4 μs
T_4	$H_2 : \{0,1\}^* \rightarrow \mathbb{G}_1$	0.05 ms
T_5	$xP_1 \in \mathbb{G}_1$	1.44 ms
T_6	$\mathcal{H}(m) \in \mathbb{G}$	1.3 ms
T_7	$h_1 + h_2 \in \mathbb{G}$	1.31 μs
T_8	HMAC [RFC 2104] (128 bit)	1.58 μs
T_9	Modular Exponentiation (1024bit)	5.88 ms
T_{10}	Point Multiplication (1024bit)	1.36 μs
T_{11}	Paillier encryption $(1024bits)$	19.47 ms
T_{12}	Paillier decryption $(1024bits)$	18.88 ms
T_{13}	Paillier aggregation $(1024bits)$	19.9 μs
T_{14}	Paillier Ciphertext Exponentiation	24.25 ms

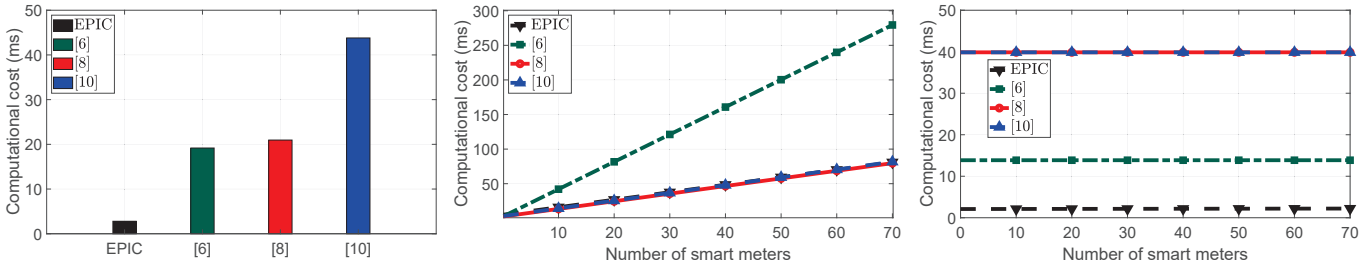
A. Computation and Communication Overhead

To evaluate the communication and computation overheads of EPIC, we implemented the required cryptographic operations using Python charm cryptographic library [22] running on an Intel Core i7-4765T 2.00 GHz and 8 GB RAM. We used supersingular elliptic curve with the symmetric Type 1 pairing of size 512 bits (SS512 curve) for bilinear pairing and a standard elliptic curve secp160r1 for the homomorphic hash function [23]. All cryptographic operations were run 1,000 times and average measurements are reported in the upper part of Table IV. Since we compare the overhead of EPIC to the proposed schemes in [6], [8], [10], we include in the lower part of Table IV the computation measurements of the cryptographic operations needed in these schemes.

1) Computation Overhead: The computation overhead is defined as the processing time required by each node in the network. These nodes are SMs, the gateway and the utility.

For the single hop model, the time-consuming operations required by SMs are one homomorphic hash generation which requires T_6; one HMAC generation which requires T_8; and one signature generation which requires $T_4 + T_5$. Using the measurements in Table IV, the total time required by each meter is 2.79 ms. For n SMs, the computations required by the gateway are batch signatures verification (as in Eq. 4) which requires $(n+1)T_1 + (n-1)T_2 + (n-1)T_3 + nT_4$; batch homomorphic hashes verification (as in Eq. 5) which requires $T_6 + (n-1)T_7$; one homomorphic hash generation which requires T_6; one HMAC generation which requires T_8; and one signature generation which requires $T_4 + T_5$. The total time required by the gateway for these operations is $1.0836n + 5.1128$ ms. For the utility, one signature verification operation plus n HMAC computations are required to verify the received packet and one arithmetic addition operation to obtain the aggregated reading. These operations require $0.001n + 2.1037$ ms. For the schemes in [6], [8], [10], we followed the same procedure to compute the computation overhead of each entity and the results are shown in Fig. 7.

As shown in Fig. 7(a), EPIC imposes the least computation overhead on the SMs comparing to the existing schemes.



(a) SM computations. (b) Gateway computations. (c) Utility Computations.

Fig. 7. Computation overhead comparison

This is because EPIC uses efficient masking technique, while the other schemes use computationally-extensive operations to encrypt the fine-grained readings. For the gateway, Fig. 7(b) shows that the computational overhead of the gateway in EPIC is found to be close to those of [8] and [10]. For the computation overhead of the utility, EPIC is more efficient than the existing schemes because simple arithmetic addition is needed to remove the utility mask and recover the aggregated reading as shown in equation 11 , while in the schemes of [6], [8], [10], time-consuming decryption operation is needed, as given in the lower part of Table IV. In addition, as shown in Fig. 7(c), the proposed schemes in [6], [8], [10] have constant computation time because the utility decrypts only one aggregated ciphertext regardless of the number of SMs, whereas in EPIC, the utility's computation overhead increases linearly at a rate of 1.31 μs/SM because the utility receives one homomorphic hash for every SM, and thus more operations are needed. However, the utility's computation overhead of EPIC is much less than those of the other schemes.

2) Communication Overhead: The communication overhead is measured by the size of transmitted messages between the network entities in bytes. In specific, we evaluate SM-to-Gateway and Gateway-to-Utility communication overhead.

The SM-to-Gateway communication overhead in EPIC can be computed using the packet format in (3) as follows. Each SM transmits a 16-byte masked reading, a 4-byte timestamp, a 20-byte homomorphic hashe, a 16-byte MAC and a 64-byte signature. Therefore, the total size of a SM's message is 120 bytes. For a network of size n, the communication overhead between all SMs and the gateway is $120n$. On the other hand, the Gateway-to-Utility communication overhead depends mainly on n which is the total number of SMs in the network. The total size of the gateway message to the utility is $20n + 100$ bytes. We used the same procedure to compute the communication overhead of the proposed schemes in [6], [8] and [10].

The values of computations and communication overhead presented in this section were used within the ns-3 simulation presented in the following subsection.

B. Experiment and Measurement Results

1) Experimental Setup: We used network simulator ns-3.27 [24] to assess the impact of the communication/computation overhead reduction on the network performance. We implemented a wireless mesh network that mimics the AMI network using IEEE 802.11s standard. The underlying MAC protocol is IEEE 802.11g. TCP was used at the transport layer for a reliable communication, and maximum segment size (MSS) is 536 bytes.

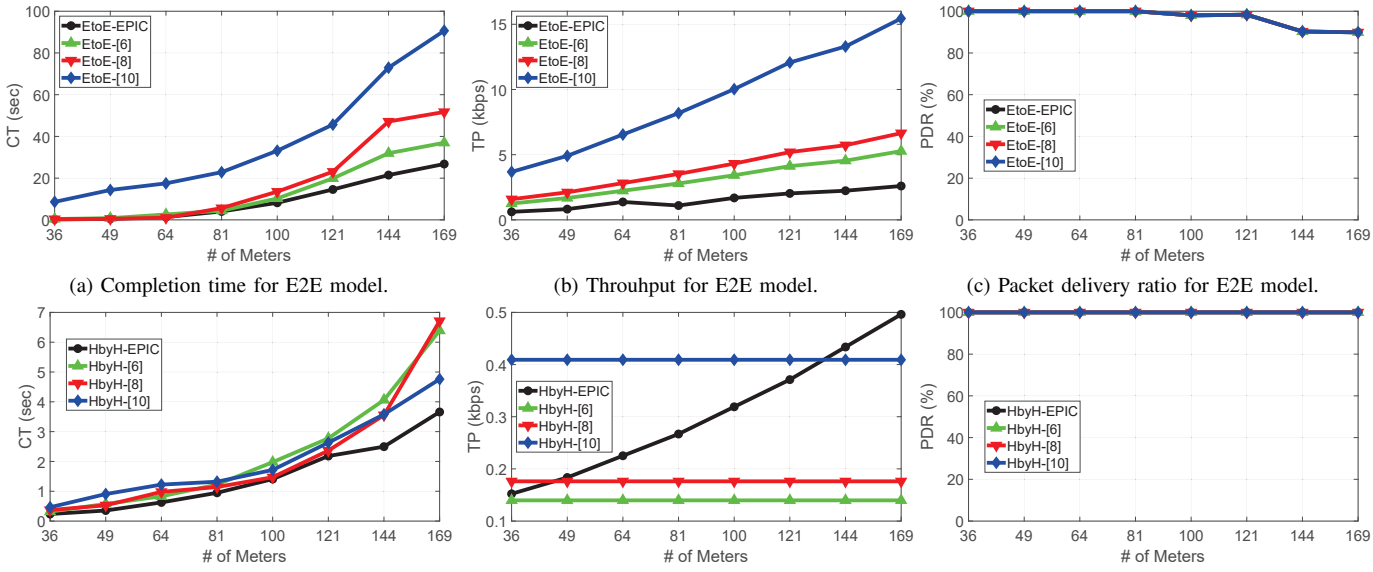
We created grid topologies of size N, where $N \in \{36, 49, 64, 81, 100, 121, 144, 169\}$. For each N, we ran the schemes for 30 rounds and average results are reported. One of the nodes in each topology acts as the gateway while the other $(N-1)$ nodes act as smart meters.

At the beginning of each data collection cycle, each meter reports its power consumption reading to the gateway. We assumed that the data collection is performed periodically every 60 seconds [25]. We simulated two different network models: End-to-End (EtoE) and Hop-by-Hop (HbyH). In EtoE data collection model, all smart meters simultaneously report their readings to the gateway directly and multi-hop packet relay may be needed. In HbyH model, a minimum spanning tree of the network is created, and parent-child relationships are assigned. Moreover, in HbyH model, leaf meters send their readings to their parent meter periodically, the parent meter aggregates its reading with the readings received from the child meters, and sends an aggregated reading to their parent meter. This process goes on up to the gateway. Finally, the gateway aggregates the readings received from its child meters and sends an aggregated reading to the utility.

2) Baselines and Performance Metrics: We use three existing works [6], [8], [10] as baselines to compare the performance of EPIC. In [6], the meters send their readings in blinded form, and the data aggregation is performed on ciphertext. The total power usage can be recovered by computing a discrete log problem. The scheme in [8] uses Paillier cryptosystem to perform data aggregation using partially homomorphic encryption. This baseline generates larger data packets when compared to [6]. The last baseline [10] also uses Paillier cryptosystem, but it differs from [8] in that it uses two signatures for each report. Hence, [10] introduces extra overhead.

For performance evaluation, we used the following metrics:

- *Average Completion Time (CT)*: It is the elapsed time for gathering all the measurement data from all of the nodes and aggregating them at the gateway in one cycle. We measure CT at the application layer so that the cryptographic operations are taken into account.
- *Throughput (TP)*: It is the average amount of data received by the gateway per second. This parameter is an indicator for the bandwidth usage of each scheme, i.e., as measurement of this metric increases, as it is worse.



(a) Completion time for E2E model. (b) Throuhput for E2E model. (c) Packet delivery ratio for E2E model.

(d) Completion time for HbyH model. (e) Throuhput for HbyH model. (f) Packet delivery ratio for HbyH model.

Fig. 8. Computation overhead comparison

- *Packet Delivery Ratio (PDR)*: It is the ratio of the number of packets received by the gateway to the number of packets that are expected to be received by the gateway.

3) Simulation Results and Discussions: In Fig. 8(a) and Fig. 8(d), we present the data collection completion time values. As the network grows, the time required to complete a data collection cycle increases. In both data collection methods, EPIC requires the least time for all topology sizes in a round because it both has a moderate processing delay for aggregation and generates comparable size of power readings. In addition, the approaches require similar amount of time for data collection until 81-node topologies. Thereafter, the values dramatically increase for EtoE data collection. This difference can be attributed to the propagation delay mostly. It includes backoff waitings due to external collisions while accessing the medium to transmit the data packets and the path discovery process performed by the HWMP which is default routing protocol of IEEE 802.11s standard [26]. The EtoE data collection typically needs more hops to deliver data packets to the destination because parent and child meters are one-hop neighbors of each other in the data reporting hierarchy trees. Thus, the data packets are exposed to more backoff waitings on the path towards the destination. This results in a dramatic increase in EtoE data collection. Also, we would like to point out the remarkable difference between [10] and the other approaches. Although [10] takes shorter than [6] does to perform cryptographic operations, it incurs an extra delay due to the segmentation by TCP at the transport layer. Since [10] generates larger data packets than MSS, a power reading is transmitted in multiple segments, which results in an extra delay due to the extra backoff waitings.

It can be seen that in HbyH data collection that the approaches require far less time to complete a data collection round. Also, they have similar values for all topology sizes. Since parent and child meters are one-hop neighbor of each other, the backoff waitings decrease thanks to less competition in medium access. EPIC slightly outperforms the other approaches. It always requires less time than the other approaches do to complete a data collection round. Although it incurs similar processing delay for aggregation and generates larger aggregated readings beyond a level of the hierarchy tree towards the gateway, it requires the least time. This is because EPIC takes advantage of far shorter propagation delays below that level thanks to smaller aggregated power readings [27].

Secondly, we analyze the throughput performance to discuss the bandwidth usage of the approaches. As shown in Fig. 8(b) and Fig. 8(e), the approaches produce more throughput at the gateway in the EtoE data collection when compared to the throughput values for the HbyH data collection. This is because power readings are aggregated at some intermediate meters in the HbyH model. Hence, average amount of data received by the gateway decreases. [10] produces the most throughput in both the EtoE and HbyH data collection methods since it generates the biggest data packets compared to the others. It is followed by [8] and [6], respectively. EPIC produces the least throughput because it generates the smallest data packets for power readings. The TP values linearly increase in EtoE data collection because the number of power readings delivered to the gateway increases as the network grows. However, in HbyH data collection, the throughput that EPIC produces at the gateway linearly increases while the others remain constant although the number of power readings reported by each approach is the same at each topology size. The additional overhead is required to achieve E2E data integrity, authenticity and dynamic pricing that are not achieved in the other schemes.

We investigate the PDR values in order to find out how reliable the schemes are. As shown in Fig. 8(c) and Fig. 8(f), all schemes achieve more than 89% PDR. Also, all the scheme achieves the same PDR which indicates that the PDR depends on the network topology, i.e., how the SMs are organized, not the data collection scheme. While the schemes

can achieve 100% at each topology size in the HbyH data collection, the values slightly decrease after 81-node topology in the EtoE data collection. This is due to the loss of one of three-way handshake messages between the gateway and a physically distant node (especially the nodes at the edge of the network) in the network. The TCP is a connection-oriented communication protocol, so it needs to establish a connection before sending data packets. As the number of hops between the hosts increases, it is more likely to lose any of the three-way handshake messages. Since there is a limit to retransmit these messages, it is likely to fail a connection. If the connection fails, the data packets cannot be transferred, and this results in lower PDR values.

VIII. Conclusion

In this paper, we proposed EPIC, an efficient privacy-preserving scheme with E2E data integrity verification and collusion-resistance for AMI networks. EPIC enables the utility to verify the integrity of the aggregated reading and identify the attackers without accessing the individual readings to preserve privacy. The utility can also generate electricity bills based on dynamic prices without violating consumers' privacy. A formal security proof, and a probabilistic model are provided to demonstrate that EPIC can preserve the consumers' privacy with E2E data integrity and high protection against collusion attacks. Moreover, we evaluated the performance of EPIC using ns-3 and the measurements demonstrated that EPIC is efficient when compared to similar existing schemes and can collect periodic power consumption data in the AMI network without consuming excessive bandwidth so that the other types of traffic can obtain more bandwidth.

IX. Acknowledgement

This work is supported by US National Science Foundation under the grant number CNS-1619250.

References

[1] A. Sherif, A. Alsharif, M. Mahmoud, M. M. Abdallah, and M. Song, "Efficient privacy-preserving aggregation scheme for data sets," *Proceedings of the 25th International Conference on Telecommunications (ICT)*, June 2018.

[2] A. H. Mohsenian-Rad, V. W. Wong, J. Jatskevich, R. Schober, and A. Leon-Garcia, "Autonomous demand-side management based on game-theoretic energy consumption scheduling for the future smart grid," *IEEE trans. on Smart Grid*, vol. 1, no. 3, pp. 320–331, 2010.

[3] G. W. Hart, "Nonintrusive appliance load monitoring," *Proceedings of the IEEE*, vol. 80, no. 12, pp. 1870–1891, 1992.

[4] C. Laughman, K. Lee, R. Cox, S. Shaw, S. Leeb, L. Norford, and P. Armstrong, "Power signature analysis," *IEEE power and energy magazine*, vol. 99, no. 2, pp. 56–63, 2003.

[5] A. Alsharif, M. Nabil, M. Mahmoud, and M. M. Abdallah, "Privacy-preserving collection of power consumption data for enhanced AMI networks," *Proceedings of the 25th International Conference on Telecommunications (ICT)*, June 2018.

[6] C. Fan, S. Huang, and Y. Lai, "Privacy-enhanced data aggregation scheme against internal attackers in smart grid," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 1, pp. 666–675, 2014.

[7] R. Lu and X. Liang and X. Li and X. Lin and X. Shen, "EPPA: An efficient and privacy-preserving aggregation scheme for secure smart grid communications," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 9, pp. 1621–1631, Sept 2012.

[8] H. Shen, M. Zhang, and J. Shen, "Efficient privacy-preserving cube-data aggregation scheme for smart grids," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 6, pp. 1369–1381, June 2017.

[9] S. Li, K. Xue, Q. Yang, and P. Hong, "PPMA: Privacy-preserving multisubset data aggregation in smart grid," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 2, pp. 462–471, 2018.

[10] F. Li and B. Luo, "Preserving data integrity for smart grid data aggregation," *Proceedings of the IEEE Third International Conference on Smart Grid Communications (SmartGridComm)*, pp. 366–371, 2012.

[11] H. Mohammed, S. Tonyali, K. Rabieh, M. Mahmoud, and K. Akkaya, "Efficient privacy-preserving data collection scheme for smart grid ami networks," *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*, Dec 2016.

[12] Z. Erkin and G. Tsudik, "Private computation of spatial and temporal power consumption with smart meters," *Proceedings of the International Conference on Applied Cryptography and Network Security*, pp. 561–577, 2012.

[13] F. D. Garcia and B. J. B., "Privacy-friendly energy-metering via homomorphic encryption," *Springer Security and Trust Management*, pp. 226–238, 2010.

[14] Z. Li and G. Guang, "Data aggregation integrity based on homomorphic primitives in sensor networks," *Proceedings of the Springer International Conference on Ad-Hoc Networks and Wireless*, pp. 149–162, 2010.

[15] F. Knirsch, G. Eibl, and D. Engel, "Error-resilient masking approaches for privacy preserving data aggregation," *IEEE Transactions on Smart Grid*, vol. 9, no. 4, pp. 3351–3361, July 2018.

[16] H. Shacham and B. Waters, "Compact proofs of retrievability," *Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security*, pp. 90–107, 2008.

[17] A. F. Barsoum and M. A. Hasan, "Provable multicopy dynamic data possession in cloud computing systems." *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 3, pp. 485–497, 2015.

[18] J. Yu, K. Ren, C. Wang, and V. Varadharajan, "Enabling cloud storage auditing with key-exposure resistance," *IEEE Transactions on Information forensics and security*, vol. 10, no. 6, pp. 1167–1179, 2015.

[19] M. N. Krohn, M. J. Freedman, and D. Mazieres, "On-the-fly verification of rateless erasure codes for efficient content distribution," *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 226–240, 2004.

[20] F. D. Garcia and B. Jacobs, "Privacy-friendly energy-metering via homomorphic encryption," *Springer International Workshop on Security and Trust Management*, pp. 226–238, 2010.

[21] M. Bellare, "New proofs for NMAC and HMAC: security without collision resistance," *Journal of Cryptology*, vol. 28, no. 4, pp. 844–878, 2015.

[22] J. A. Akinyele, C. Garman, I. Miers, M. W. Pagano, M. Rushanan, M. Green, and A. D. Rubin, "Charm: a framework for rapidly prototyping cryptosystems," *Journal of Cryptographic Engineering*, vol. 3, no. 2, pp. 111–128, 2013.

[23] The Standards for Efficient Cryptography Group (SECG), "2: Recommended elliptic curve domain parameters," 2000.

[24] The NS-3 Consortium, "ns-3: Network Simulator 3," ns-3.27, 2017. [Online]. Available: https://www.nsnam.org//ns-3-27/

[25] A. Beussink, K. Akkaya, I. F. Senturk, and M. M. E. A. Mahmoud, "Preserving consumer privacy on IEEE 802.11s-based smart grid AMI networks using data obfuscation," *Proceedings of the IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2014.

[26] M. Bahr, "Update on the hybrid wireless mesh protocol of IEEE 802.11 s," *Proceedings of the IEEE International Conference on Mobile Adhoc and Sensor Systems*, 2007.

[27] J. Korhonen and Y. Wang, "Effect of packet size on loss rate and delay in wireless links," *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, 2005.

Ahmad Alsharif (M'18) is currently a cybersecurity instructor at the Computer Science Department, University of Central Arkansas, USA, and a Ph.D. candidate at the Department of Electrical & Computer Engineering, Tennessee Tech. University, USA. He received the BSc and MSc degrees in Electrical Engineering from Benha University, EGYPT in 2009 and 2015, respectively. In 2009, he was one of the recipients of the young innovator award from the Egyptian Industrial Modernisation Centre. His research interests include security and privacy in smart grid, cyberphysical systems, vehicular Ad Hoc networks, multi-hop cellular networks.

Mahmoud Nabil is currently a Graduate Research Assistant in the Department of Electrical & Computer Engineering, Tennessee Tech. University, USA and pursuing his Ph.D. degree in the same department. He received the B.S. degree and the M.S. degree in Computer Engineering from Cairo University, Cairo, Egypt in 2012 and 2016, respectively. His research interests include machine learning, cryptography and network security, smart-grid and AMI networks, and vehicular ad-hoc networks.

Dr. Samet Tonyali received his PhD in Electrical & Computer Engineering at Florida International University in 2018. He received his B.S. and M.S. degrees in Computer Engineering at Marmara University, Istanbul, TURKEY in 2011 and 2013, respectively. He worked as a teaching assistant for 2 and a half years and as a graduate research assistant for 3 and a half years during his M.S. and PhD education. His interests are smart grid communications, cyberphysical systems, Internet of Things, and security and privacy.

Hawzhin Mohammad received his BSc degree with distinction in electrical engineering from Salahaddin University-Erbil in 2000 and MSc degree from Tennessee Tech. University, USA in 2017. He is currently working toward PhD in the Department of Electrical & Computer Engineering, Tennessee Technological University, USA. His Research interests include wireless network security.

Dr. Mohamed M. E. A. Mahmoud received PhD degree from the University of Waterloo in April 2011. From May 2011 to May 2012, he worked as a postdoctoral fellow in the Broadband Communications Research group - University of Waterloo. From August 2012 to July 2013, he worked as a visiting scholar in University of Waterloo, and a postdoctoral fellow in Ryerson University. Currently, Dr Mahmoud is an associate professor in Department Electrical and Computer Engineering, Tennessee Tech University, USA. The research interests of Dr. Mahmoud include security and privacy preserving schemes for smart grid communication network, mobile ad hoc network, sensor network, and delay-tolerant network. Dr. Mahmoud has received NSERC-PDF award. He won the Best Paper Award from IEEE International Conference on Communications (ICC'09), Dresden, Germany, 2009. Dr. Mahmoud is the author for more than twenty three papers published in major IEEE conferences and journals, such as INFOCOM conference and IEEE Transactions on Vehicular Technology, Mobile Computing, and Parallel and Distributed Systems. He serves as an Associate Editor in Springer journal of peer-to-peer networking and applications. He served as a technical program committee member for several IEEE conferences and as a reviewer for several journals and conferences such as IEEE Transactions on Vehicular Technology, IEEE Transactions on Parallel and Distributed Systems, and the journal of Peer-to-Peer Networking.

Dr. Kemal Akkaya is a professor in the Department of Electrical and Computer Engineering at Florida International University. He received his PhD in Computer Science from University of Maryland Baltimore County in 2005 and joined the department of Computer Science at Southern Illinois University (SIU) as an assistant professor. Dr. Akkaya was an associate professor at SIU from 2011 to 2014. He was also a visiting professor at The George Washington University in Fall 2013. His current research interests include security and privacy, energy-aware routing, topology control, and quality of service issues in a variety of wireless networks such as sensor networks, multimedia sensor networks, smart-grid communication networks and vehicular networks. Dr. Akkaya is a member of IEEE. He is the area editor of Elsevier Ad Hoc Network Journal and serves on the editorial board of IEEE Communication Surveys and Tutorials. He has served as the guest editor for Journal of High-Speed Networks, Computer Communications Journal, Elsevier Ad Hoc Networks Journal and in the TPC of many leading wireless networking conferences including IEEE ICC, Globecom, LCN and WCNC. He has published over 150 papers in peer reviewed journal and conferences. He has received "Top Cited" article award from Elsevier in 2010.