IEEE Access
The journal for rapid open access publishing

IEEE Xplore Digital Library
http://ieeexplore.ieee.org/Xplore

PPETD: Privacy-Preserving Electricity Theft Detection Scheme with Load Monitoring and Billing for AMI Networks

MAHMOUD NABIL[1], MUHAMMAD ISMAIL[2], Senior Member, IEEE, MOHAMED MAHMOUD[1], Member, IEEE, WALEED ALASMARY[3], Member, IEEE, and ERCHIN SERPEDIN[4], Fellow, IEEE,

Index Terms—Privacy preservation, machine learning, electricity theft detection, dynamic billing, and secure multi-party computation.

Abstract—In advanced metering infrastructure (AMI) networks, smart meters installed at the consumer side should report fine-grained power consumption readings (every few minutes) to the system operator for billing, real-time load monitoring, and energy management. On the other hand, AMI networks are vulnerable to cyber-attacks where malicious consumers report false (low) electricity consumption to reduce their bills in an illegal way. Therefore, it is imperative to develop schemes to accurately identify the consumers that steal electricity by reporting false electricity usage. Most of the existing schemes rely on machine learning for electricity theft detection using the consumers' fine-grained power consumption meter readings. However, this fine-grained data that is used for electricity theft detection, load monitoring, and billing can also be misused to infer sensitive information regarding the consumers such as whether they are on travel, the appliances they use, etc. In this paper, we propose an efficient and privacy-preserving electricity theft detection scheme for AMI network and we refer to it as PPETD. Our scheme allows system operators to identify the electricity thefts, monitor the loads, and compute electricity bills efficiently using masked fine-grained meter readings without violating the consumers' privacy. PPETD uses secret sharing to allow the consumers to send masked readings to the system operator such that these readings can be aggregated for the purpose of monitoring and billing. In addition, secure two-party protocols using arithmetic and binary circuits are executed by the system operator and each consumer to evaluate a generalized convolutional-neural network model on the reported masked fine-grained power consumption readings for the purpose of electricity theft detection. An extensive analysis on real datasets is performed to evaluate the security and the performance of PPETD. Our results confirm that our scheme is accurate in detecting fraudulent consumers with privacy preservation and acceptable communication and computation overhead.

I. Introduction

Electricity theft is a serious problem in the existing power grid, which causes great economic loss. Many countries experience a considerable amount of electricity theft. In the United States, the electricity theft costs $6 billion/year [1] while in the United Kingdom, electricity theft costs $173 million [2]. In Canada, there is a loss of around $100 million per year [3]. For developing economics, the losses have much worse consequences. India loses $17 billion every year due to electricity theft [1]. Other developing countries lose almost 50% of their electricity revenue [5].

The Smart Grid (SG) is a revolutionary upgrade to the current power grid that aims to improve the grid efficiency, sustainability, and security [11]. One of the main components of the SG is the Advanced Metering Infrastructure (AMI) networks, which use two-way communications to connect the smart meters (SMs), installed at consumers' houses, to the system operator (SO). Figure 1 gives a conceptual architecture for the SG. As shown in the figure, the SG has AMI networks and a system operator. Each AMI network covers an area, e.g., a neighborhood, and sends to the SO a massive amount of fine-grained electricity consumption data, e.g., every few minutes. Using these data, the SO can run demand-response programs to reduce the load and balance supply and demand, compute bills using dynamic pricing, and monitor/forecast loads [?], [11].

This new architecture of the power grid can limit the traditional (physical) electricity theft attacks, e.g., meter tampering and line hooking [1]. However, the use of SMs opens the door wide for new serious cybersecurity threats. For example, in the context of electricity theft, a malicious consumer may hack its SM to manipulate (reduce) the amount of energy consumption reported to the SO to reduce their bills. These new attacks not only cause financial losses but also jeopardize the integrity of the power grid and hinder its reliability because the power consumption data reported by the SMs are used by the SO for energy management and load monitoring. The use of the SMs' false data may cause instability to the power grid and, in severe cases, may cause blackouts.

Different hardware and software-based solutions have been proposed in the literature to detect electricity theft cyber-attacks [1], [12], [13], [15]. Among the existing solutions, data-driven approaches are the most promising because they have small cost and provide good detection rate [1], [12], [13]. Specifically, recent studies [1], [12] have indicated that utilizing the fine-grained energy consumption data with machine-learning-based models can accurately detect electricity thefts. These models can be trained offline using honest and malicious energy consumption data samples. After that, the trained models can be used online to detect electricity theft. However, the existing models suffer from the following limitations.

Digital Object Identifier: 10.1109/ACCESS.2019.2925322

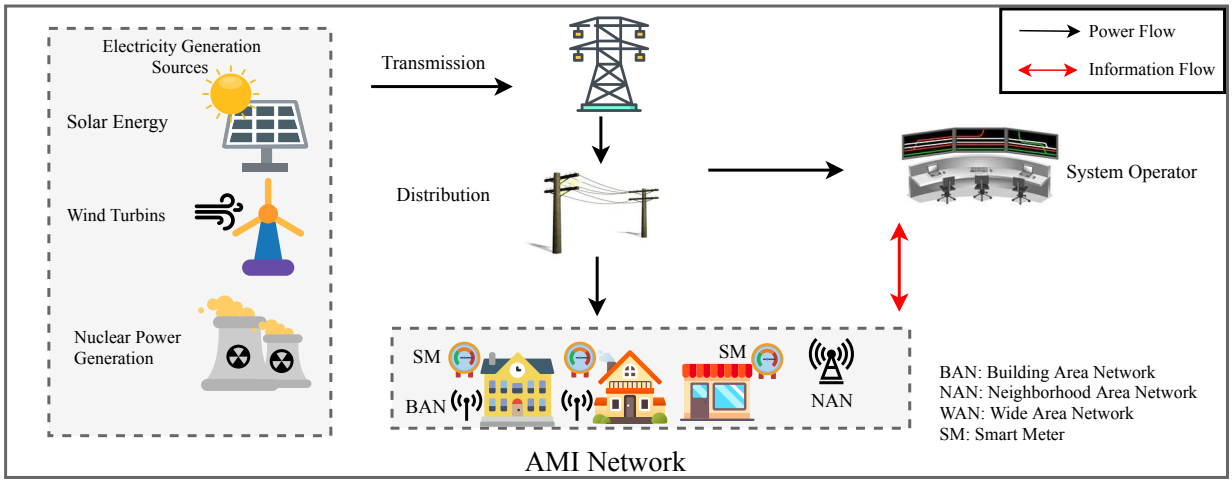


Fig. 1: Smart grid conceptual architecture.

First, most of the existing models need to access the consumers' fine-grained power consumption data to detect electricity theft, which seriously invades the privacy of the consumers. In particular, fine-grained consumption data can reveal sensitive information of the households' activities, such as appliances in use, the times the consumers leave and return home, whether consumers are on travel, etc. [17]. These data can be collected and misused. For instance, insurance companies could purchase these data to adapt their insurance plans according to the activities of each consumer. Marketing companies may also be interested in these data to identify potential customers. Most importantly, criminals (such as thieves) may use these data to know the best times to commit crimes (e.g., break into houses if consumers are on travel). Finally, according to the Electronic Privacy Information Center, determining consumers' personal activities is a serious privacy concern in smart grid [?].

To the best of our knowledge, very few works have tried to address privacy preservation in electricity theft detection [21]–[24], [29]. State estimation is used in [21]–[23] where a set of distributed algorithms are employed between all SMs in the network. Nevertheless, distributed state estimation approaches may fail if any SM manipulate the messages it sends to other peers. Also, state estimation approaches usually give lower performance than machine learning models [1]. Besides, the schemes in [21], [22] require knowing the power line losses, which may be infeasible in practice. In [24], [29] machine learning and statistical models are used for privacy-preserving electricity theft detection. However, an online trusted entity is assumed to participate in the electricity theft detection phase. In practice, it is impossible to guarantee that a party (that is assumed trusted) does not misuse the data of the consumers.

The second limitation in the literature is that some of the existing electricity theft detectors are either based on shallow machine learning techniques [26], [27] or static machine learning techniques [28]. On one hand, these schemes do not exploit the temporal correlation nature of the energy consumption data, which results in low detection rates. On the other hand, convolutional neural networks (CNNs) can deal with time-series data, and hence, can capture the temporal correlation in the time-dependent energy consumption data leading to an improved detection performance. The final limitation is that, most of the existing research works, e.g., [?], [1], focus on consumer-specific detectors, where a detector is designed for each consumer based on the historical meter readings of that consumer. These schemes cannot be used for the new consumers who do not have any historical readings. Besides, consumer-specific detectors are vulnerable to data contamination attacks where malicious consumers report false readings during the detector's design/training stage. On the other hand, general detectors are designed using energy consumption readings from all consumers, and hence, they can be used to detect electricity theft cyber-attacks launched by new consumers and are more robust against data contamination attacks.

In this paper, we propose an efficient and privacy preserving electricity theft detection scheme for AMI network, and we refer to it as PPETD. The scheme aims at efficient detection of electricity thefts using convolutional machine learning model while preserving consumers' privacy and enabling the SO to monitor the loads and compute electricity bills following a dynamic pricing mechanism. In PPETD, each SM shares a set of secrets with other SMs in the AMI network to efficiently compute shared pairwise secret masks with each SM. During an electricity consumption reporting, the SM should blind its fine-grained reading using all the masks shared with other meters, such that all the masks get canceled after aggregating all meters' masked readings. As a result, the SO can obtain the aggregated reading for load monitoring without accessing the fine-grained readings of each consumer to preserve his/her privacy. In addition, secure two-party computation protocols for arithmetic and binary circuits are executed by each consumer's SM and

the SO using a number of consecutive reports, referred to as electricity theft detection interval. These protocols use a secure convolutional neural network model that operates on the consumers' blinded fine-grained reports and can efficiently detect electricity thefts. Furthermore, the masks are designed in a way that allows the SO to obtain the total consumption of an SM over a billing interval to compute the bills following dynamic pricing.

The novelty and contributions of this paper can be summarized as follows:

- We propose a privacy-preserving electricity theft detection scheme. To the best of our knowledge, this work is the first to investigate the energy theft detection problem using privacy-preserving machine learning model. Our model is evaluated using a set of privacy-preserving protocols and can detect electricity theft effectively.
- Our simulation results show that PPETD can preserve consumers' privacy with acceptable communication and computation overhead and a very slight loss in performance, compared with the existing machine-learning-based electricity theft detection schemes that do not consider privacy preservation [1].
- In addition to theft detection with privacy preservation, PPETD also allows the SO to perform load monitoring and bill computation following dynamic pricing. In addition, unlike existing schemes [29], our solution does not need an online trusted entity for electricity theft detection. In practice, there is not any guarantee that such an entity would not misbehave and misuse the consumers' data.

The remainder of this paper is organized as follows. The considered system models and design goals are presented in section II. Preliminaries are given in section III. The proposed electricity theft detection scheme is explained in section IV. The security analysis and communication/computation overhead are given in section V and section VI, respectively. The related works are discussed in section VII. Conclusions are drawn in section VIII.

II. System Models and Design Objectives

This section discusses the considered network and threat models and the design objectives of our scheme.

A. Network Model

As shown in Figure 1, the considered network model has the following entities: the system operator and a set of SMs forming the AMI network. The role of each entity is described below.

- The system operator (SO): The consumers of an AMI network should periodically report their fine-grained power consumption data to the SO that should use these data for energy management and load monitoring. In addition, the SO runs the neural network model for electricity theft detection and computes the electricity bills following dynamic pricing.
- Smart meters (SMs): We consider a set of SMs $\mathbb{SM} = \{SM_i, 1 \leq i \leq |\mathbb{SM}|\}$. SMs are installed at the consumers' premises and should periodically report to the SO masked readings for the fine-grained power consumption.

B. Threat Model

The SOs are honest-but-curious, which means they follow the proposed scheme honestly but they want to learn the fine-grained power consumption of the consumers. Some of the consumers are malicious and they may report false (low) power consumption to the SO to reduce their bills. Some consumers are also curious to know the fine-grained power consumption of other consumers. In addition, a set of external adversaries $\mathcal{A}$ eavesdrop all the communications between consumers and the SO to obtain the consumers' fine-grained readings to learn sensitive information about their activities. Moreover, adversaries can work individually or collude together to launch stronger attacks.

To conclude, PPETD mainly focuses on the detection of malicious consumers that report false (low) power consumption readings to steal electricity. In addition, we aim to preserve the privacy of the consumers' fine-grained power consumption data, i.e., no entity should be able to access these data. Other attacks are beyond the scope of this paper.

C. Functionality Requirements and Design Objectives

We aim to achieve the following functionality and security requirements in our scheme:

1) Functionality Requirements:

(F1) At the end of each reporting period, PPETD should allow each SO to obtain the total aggregated electricity consumption of each AMI network for the purpose of energy management and load monitoring.

(F2) PPETD should allow the SO to efficiently compute the electricity bill for each customer following dynamic pricing.

(F3) At the end of each electricity theft detection interval, PPETD should allow the SO to detect the malicious consumers effectively.

2) Security Requirements:

(S1) Consumers' privacy preservation: At any reporting period, no entity should be able to access the fine-grained power consumption data of any consumer.

(S2) Aggregated power confidentiality: At any reporting period, no entity except the SO should be able to access the aggregated power consumption data.

(S3) Electricity theft detection: At the end of each electricity theft interval, the SO and the SMs should be able to run a secure machine learning model that can detect the malicious consumers using the masked fine-grained power consumption

readings reported by the consumers. The scheme should be secure against any misbehavior from the malicious consumers that aim to steal energy without being detected.

III. Preliminaries

In this section, we present the protocols, techniques, and notations that we will use in the coming sections.

A. Additive Secret Sharing

Secret sharing is a famous cryptographic technique first proposed by A. Shamir in [32]. The idea is that, in order to protect a secret data (i.e., key), the data can be distributed among a group of parties. To illustrate, a secret x can be split into random-looking pieces $\{x_1 \dots x_n\}$ called shares. The shares are then distributed to the involved parties, where each party should get a single share. The secret x can be reconstructed by combining a sufficient subset of the shares, however, a smaller subset of the shares do not reveal any information about x. Shared values will be denoted as $[\![x]\!]$. Secret sharing is the basis of many secure multiparty computation protocols that rely on secure arithmetic circuit evaluation [34].

B. SPDZ Protocol

SPDZ [36] (or speedz) is a secure multi-party computation (SMC) protocol where a number of parties collaboratively perform certain computation(s) of a function on their private data without revealing this data to other parties. All parties are intended to know the function output. SPDZ protocol uses arithmetic circuits to evaluate the computations, thus it is highly efficient. To evaluate an arithmetic circuit privately on secret inputs, participants have to secretly share the inputs. In particular, a private input value x is split into N pieces x_i, where $\sum_{i=0}^{N} x_i = x$, for N-parties computation. When a computation is performed on secret shared values, each participant should perform a local computation on its shares of the secret values.

The notation of shared computation is as follows. When the desired computation is given by $x+y = z$, the values of x and y are shared to create secret-shared values $[\![x]\!]$ and $[\![y]\!]$. The required mutli-party secret computation should be represented as: $[\![x]\!] + [\![y]\!] = [\![z]\!]$. Each participant i should perform operations on x_i and y_i to obtain z_i. To obtain the value of z from $[\![z]\!]$, each participant should broadcast its shares z_i to other participants, where the summation of the shares gives the value of z.

In SPDZ, addition and multiplication operations are the two main operations considered by the protocol. Addition is performed by simply adding the shares locally, i.e., the shared computation $[\![x]\!] + [\![y]\!] = [\![z]\!]$ is achieved by local computation of $x_i + y_i = z_i$. Multiplication operation is more difficult to compute (i.e., cannot be computed locally). To get $[\![x]\!] \times [\![y]\!] = [\![z]\!]$, a triple (a, b, c) should be constructed so that $a \times b = c$. This triple is secret and can be shared offline before the circuit is evaluated such that every participant can compute a masked version of its shares $\alpha_i = (x_i - a_i)$ and $\beta_i = (y_i - b_i)$. Subsequently, all individuals broadcast their masked shares α_i and β_i. Both α and β can be reconstructed from the shares α_i and β_i, and can be made publicly available to every party. Then, a share z_i can be computed by:

$$\begin{aligned} z_i &= c_i + \alpha b_i + \beta a_i \\ &= c_i + (x - a)b_i + (y - b)a_i. \end{aligned}$$

Moreover, a random party adds on the public value $(x - a)(y - b) = \alpha\beta$ to its share of z_i so that by summing the local computations z_i, eventually determines the required multiplication z.

A Message Authentication Code (MAC) is used by SPDZ protocol [43] to protect the data from active adversaries during computations. Before starting the computations, the MAC key $\triangle$ is secret shared among the parties such that no party knows it. The MAC keys are used in generating a MAC value for all the secret shared values involved in the protocol. Note that, MAC values are also secret shared among the parties. Then, after the circuit evaluation, the parties open the output values of the protocol and the corresponding MACs, and check whether the MACs are correct or not.

C. Garbled Circuits

Garbled circuit [?] is a secure multi-party computation protocol that enables two parties to compute any function $f(.)$ on their private inputs. However, unlike SPDZ protocol, garbled circuits protocols represent $f(.)$ using binary gates. Therefore, garbled circuits are more general compared with the SPDZ protocol but they need more computations. In garbled circuits, one of the two parties is called generator and the other is called evaluator. The generator generates an encrypted (garbled) version of $f(.)$ and an encrypted version of the inputs of the circuit, and sends them to the evaluator. To evaluate the encrypted circuit, the evaluator needs to get also its encrypted inputs from the generator without revealing any information about these inputs. Therefore, an oblivious transfer protocol is used to transfer the evaluator's encrypted inputs from the generator to the evaluator. After getting all the required inputs to evaluate $f(.)$, the evaluator evaluates the garbled circuit securely to obtain the required result.

D. Convolutional neural networks (CNNs)

CNNs are widely used in solving many challenging machine learning problems such as computer vision applications, natural language processing, and speech processing [40]. This wide adoption of CNNs is due to their capability of capturing complex patterns in the inputs.

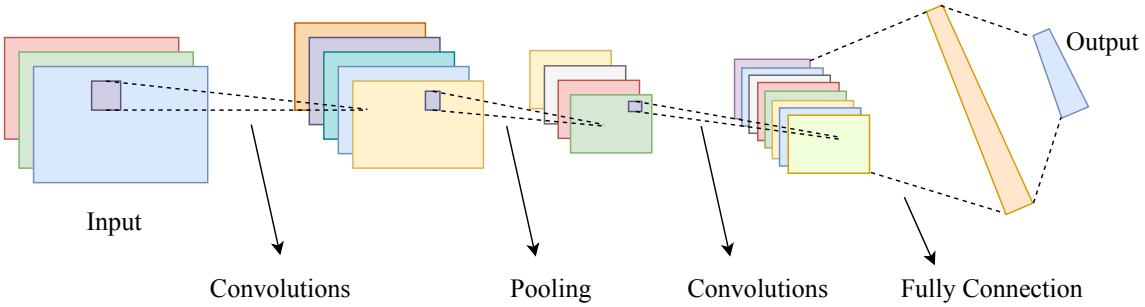


Fig. 2: Typical architecture of convolutional neural network (CNN).

1) Architecture of CNN: As shown in Figure 2, a typical architecture of convolutional neural network consists of the input layer, convolutional layer(s), pooling layer(s), fully connected layer, and Softmax output layer. The convolutional layers are composed of a number of learnable filters that are intended to extract features from the input. Learnable filters are usually small in size, and thus, they can interpret input data with strong local connectivity and dependency patterns. Convolving one filter with a channel from the input results in a two-dimensional feature map of that filter. After the convolutional step, a nonlinear activation function such as Rectified Linear Unit (ReLU), Sigmoid, or Tanh is used to enable the neural network model to make more complex decisions and solve difficult tasks. Sigmoid is commonly used with deep neural networks and if initialized properly, deep sigmoidal networks can outperform networks with other activation type [41].

The pooling layer is used to compress the output of the convolutional layer by sub-sampling the feature maps while retaining the most important information. After successive convolutional and pooling layers, the fully-connected layer is used where the features extracted can be used for inference. Softmax function is used as output layer for multi-class classification problems. Softmax outputs a probability vector for the input to be assigned a certain class label. Specifically, for an input vector $\mathbf{z} = (z_1, \ldots, z_K) \in \mathbb{R}^K$ of length K and M classes, the Softmax function is defined as follows:

$$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad for \quad i = \{1, \ldots, M\}.$$

2) CNN Training: The CNN training involves learning the weight and the bias parameters Θ by defining a cost function and selecting an optimizer. For the cost function, categorical cross-entropy C is defined to measure the loss due to the difference of two distribution y and $\hat{y}$ as follows:

$$C(y, \hat{y}) = \min_{\Theta}(-\sum_{c=1}^{M} y(c)\ log\ (\hat{y}(c))).$$

During training, an optimization method such as Stochastic Gradient Descent (SGD) [42] is used iteratively for optimizing the cost function. The optimization process involves a back-propagation step in which the CNN weights are updated using the gradients of the cost function with respect to the neural network's weights. Supervised labeled data is used to train the neural network. In addition, hyper-parameters of the neural network such as the number of neurons in each layer, the number of layers, type of optimizer, etc., can be determined using k-fold cross validation or any other validation method [?].

IV. Proposed Scheme

In this section, we give the details of PPETD starting with system setup. Then, we discuss how SMs report their power consumption readings, and how the SO verifies the reports and computes the aggregated reading for load monitoring. Finally, we explain how the electricity bills are computed using dynamic pricing, and how the SO can use the SMs' reports to run a privacy-preserving machine learning protocol with each SM to detect electricity thefts. For better readability, we define the main notations used in this section in Table I. We use the standard lowercase bold notation for vectors and uppercase bold notation for matrices. Note that, unless otherwise stated, all arithmetic operations are in $\mathbb{Z}_q^*$.

A. System Initialization

System initialization, carried out by a key distribution center (KDC), consists of the following phases: (1) Public system parameters; (2) Identity-based public/private key pairs; and (3) SMs' seed secret keys and SPDZ initialization.

1) Public system parameters: To generate the public parameters, the KDC should:

1) generate the bilinear pairing parameters $\{\mathbb{G}_1, \mathbb{G}_2, \hat{e}, P, q\}$, where $\mathbb{G}_1$ is an additive cyclic group, $\mathbb{G}_2$ is a multiplicative cyclic group of the same prime order q, and P is a generator of $\mathbb{G}_1$.
2) choose a random number $s \in \mathbb{Z}_q^*$ and compute $sP \in \mathbb{G}_1$,
3) choose two secure hash functions H_1 and H_2 defined as $H_1 : \{0,1\}^* \rightarrow \mathbb{G}_1$ and $H_2 : \{0,1\}^* \times \mathbb{G}_1 \rightarrow \mathbb{Z}_q^*$, and
4) choose a keyed hash function $\mathcal{H}(K, m) \rightarrow \mathbb{Z}_q^*$ where K is the key used to compute a keyed-hash on a message m.

TABLE I: Main notations.

Notation	Description
$[\![.]\!]$	A secret shared value
$r_i(t)$	Consumption report of SM_i at period t
r_i	Consumption report vector of SM_i over, $\mathbb{T}$ where $\mathrm{r}_i = [r_i(1) \dots r_i(\lvert\mathbb{T}\rvert)]^T$
$h_i(t)$	Hidden/masked reading of SM_i at period t
h_i	Hidden/masked consumption report vector of SM_i over $\mathbb{T}$, where $\mathrm{h}_i = [h_i(1) \dots h_i(\lvert\mathbb{T}\rvert)]^T$
SM_i	i-th smart meter
ID_i	Identity of SM_i
$\mathbb{T}$	Set of reporting periods used for theft detection $\mathbb{T} = \{t : 1 \leq t \leq \lvert\mathbb{T}\rvert\}$
$\mathcal{H}(K, m) \rightarrow \mathbb{Z}_q^*$	Keyed hash function
$m_i(t)$	Mask of SM_i at period t
m_i	Mask vector of SM_i over $\mathbb{T}$ $\mathrm{m}_i = [m_i(1) \dots m_i(\lvert\mathbb{T}\rvert)]^T$
n	Number of electricity theft detection intervals
l	Number of off/on/mid-peak reports
$q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, P, Q, H_1, H_2$	Public parameters for the ID-based scheme
$\sigma_i(t)$	Signature on the reading of SM_i at period t
$\mathbb{SM}$	Set of smart meters $\mathbb{SM} = \{SM_i, 1 \leq i \leq \lvert\mathbb{SM}\rvert\}$

Then, s is kept as a master secret and the public system parameters $param = \{\mathbb{G}_1, \mathbb{G}_2, \hat{e}, P, sP, H_1, H_2, \mathcal{H}\}$ are published.

2) Identity-based public/private key pairs: Each smart meter SM_i with an identity ID_i receives from the KDC a private key $X_i = sQ_i$ where $Q_i = H_1(\mathrm{ID}_i)$ is the corresponding public key. Similarly, the SO receives from the KDC its private/public key pairs as $X_{so} = sQ_{so}$ and $Q_{so} = H_1(\mathrm{ID}_{so})$. Since Identity-based (ID-based) cryptography is used, the SMs and the SO do not need to obtain digital certificates from the KDC to certify the public keys.

3) Smart meters' seed secret keys and SPDZ initialization: A seed secret key $K_{i,j}$ is computed for each pair of SMs in the AMI network in an efficient non-interactive way using ID-based cryptography [31]. In particular, a smart meter SM_i with private/public key pairs $X_i = sQ_i$ and $Q_i = H_1(\mathrm{ID}_i)$ can use its private key along with the public key Q_j of another smart meter SM_j to compute a key $K_{i,j} = \hat{e}(X_i, Q_j) = \hat{e}(sQ_i, Q_j) = \hat{e}(Q_i, Q_j)^s$. Similarly, SM_j uses its private key X_j along with SM_i's public key Q_i to compute the same key as $K_{i,j} = \hat{e}(Q_i, X_j) = \hat{e}(Q_i, sQ_j) = \hat{e}(Q_i, Q_j)^s$. This seed key is used by SM_i to compute secret masks for blinding its fine-grained readings (to preserve privacy). The masks are computed in such a way that after the SO aggregates all masked readings received from all SMs in the AMI network, all masks cancel each other and the aggregated power consumption is obtained [11], [35]. By performing the above steps, the SO can learn the total consumption for load monitoring without accessing the fine-grained power consumption readings of the consumers to preserve privacy.

By construction, any masked reading is secret shared between the SO and the SM, and hence, can be used in the computations of the SPDZ protocol. Thus, masks are used to preserve privacy and run multi-party computations on the fine-grained readings, as will be explained in subsection IV-E. Note that, the multiplication triplets required by the SPDZ protocol are independent of the computations of the online phase and can be made available offline using existing secure cryptographic protocols such as [43]. Thus, throughout this paper, we assume that the multiplication triplets are available for the SMs and the SO, and our main focus is on the online theft detection procedure. In addition, a MAC secret key $\triangle_{i,so} = \triangle_i + \triangle_{so}$ is secret shared between every SM_i and the SO for every circuit evaluation such that $\triangle_i$ and $\triangle_{so}$ are the SM and SO shares, respectively. Note that, every SM and the SO can randomly sample their MAC keys from $\mathbb{Z}_q^*$.

B. Reporting fine-grained power consumption readings

For each reporting period $t \in \mathbb{T}$, each $SM_i \in \mathbb{SM}$ should generate a power consumption report by executing the following steps:

- Step 1: SM_i generates a mask $m_i(t)$ using the seed secret key $K_{i,j}$ shared with every other SM as follows:
$$m_i(t) = \sum_{\substack{j<i \\ 1\leq j\leq \lvert\mathbb{SM}\rvert}} \mathcal{H}(K_{i,j}\ ,\ t) - \sum_{\substack{j>i \\ 1\leq j\leq \lvert\mathbb{SM}\rvert}} \mathcal{H}(K_{i,j}\ ,\ t).$$
It should be noted that the mask can be computed offline, i.e, before the reporting period starts.
- Step 2: SM_i masks its reading $r_i(t)$ using the mask $m_i(t)$ to get a masked reading $h_i(t)$,
$$h_i(t) = r_i(t) - m_i(t).$$
- Step 3: SM_i uses its private key X_i to compute an ID-based signature on $h_i(t)$ [30]. It first selects a random element $r \in \mathbb{Z}_q^*$ and computes $T = rQ_i$. Then, SM_i computes $e_i = H_2(h_i(t), T)$ and $S = (r + e_i)X_i$. The signature components are (T, S),
$$\sigma_i(t) = (T, S).$$
- Step 4: SM_i sends to the SO the following encrypted report:
$$Enc(h_i(t)||ID_i||t)||\sigma_i(t).$$
It also runs an input authentication protocol [43] with the MAC keys $\triangle_i$ and $\triangle_{so}$ to share a MAC on $r_i(t)$. Hence, the SO and the SM share a MAC on the reading $r_i(t)$. It should be noted that, $[\![r_i(t)]\!]$ is shared between the SO and the SM and the shares are $h_i(t)$ and $m_i(t)$. This input authentication step is necessary so that when $[\![r_i(t)]\!]$ is used as input in any computation, the integrity of the output is ensured.

Beside the aforementioned steps, the SM should store the masks used over an electricity theft detection interval $\mathbb{T} = \{t : 1 \leq t \leq \lvert\mathbb{T}\rvert\}$ in a vector m_i defined as follows:

$$\mathrm{m}_i = [m_i(1) \dots m_i(\lvert\mathbb{T}\rvert)]^T.$$

The m_i vector is needed to evaluate the theft detection model, as will be explained in subsection IV-E.

C. Aggregating fine-grained power consumption data

After collecting all the SMs' reports, the SO should verify the received signatures and aggregate all the masked messages. These are done by performing the following steps:

- Step 1: For a report received from SM_i, the SO computes $e_i = H_2(h_i(t), T)$ for $1 \leq i \leq |\mathbb{SM}|$.
- Step 2: The SO first decrypts the received reports and then it verifies the received signatures to ensure the reports' integrity and authenticity by checking $\hat{e}(P, S) \stackrel{?}{=} \hat{e}(sP, T + e_i Q_i)$. The proof of this equation is as follows:

$$\begin{aligned}\hat{e}(P,S) &= \hat{e}(P,(r+e_i)X_i)\\ &= \hat{e}(P,(r+e_i)sQ_i)\\ &= \hat{e}(sP,(r+e_i)Q_i)\\ &= \hat{e}(sP,T+e_iQ_i).\end{aligned}$$

- Step 3: The SO computes the aggregated reading $(r_{so}(t))$ as

$$\begin{aligned}r_{so}(t) = \sum_{i=1}^{|\mathbb{SM}|} h_i(t) &= \sum_{i=1}^{|\mathbb{SM}|} r_i(t) + m_i(t)\\ &= \sum_{i=1}^{|\mathbb{SM}|} r_i(t).\end{aligned}$$

By performing these steps, PPETD can achieve the functional requirement (F1) of reporting aggregated power consumption reading for load monitoring by the SO.

Beside the aforementioned steps, the SO should store the reports of SM_i over an electricity theft detection interval $\mathbb{T} = \{t : 1 \leq t \leq |\mathbb{T}|\}$ in vector h_i defined as follows:

$$\mathrm{h}_i = [h_i(1) \ldots h_i(|\mathbb{T}|)]^T.$$

The h_i vector is needed to evaluate the theft detection model and it is considered as the SO share of r_i while the SM share is m_i. Moreover, at the end of each electricity theft detection interval, the SO should also secretly share the weights/biases of the neural network model by generating a set of random matrices corresponding to each weight/bias matrix, and then it should mask each matrix with one random matrix and share the result with the SM. Subsequently, the SM and the SO could proceed with the secure evaluation of the theft detection model as will be explained in subsection IV-E.

D. Dynamic Billing

As shown in Figure 3, the billing cycle is divided into n electricity theft detection intervals (e.g., days), denoted as $\{\mathbb{T}_i : 1 \leq i \leq n\}$. Each electricity theft detection interval is divided into three periods each has l fine-grained power consumption reporting slots. The periods are named on-peak (e.g., high demand day hours), off-peak (e.g., night hours), and mid-peak (e.g., other hours). The electricity has different prices in these periods. The SO can compute bills at the end of each billing interval as follows.

1) Smart meter: From Figure 3, each SM_i should report $3ln$ power consumption reports during each billing interval. At the electricity theft detection interval $\mathbb{T}_n$, SM_i should follow the same steps of the fine-grained power consumption reporting discussed in subsection IV-B, but SM_i should use masks such that the summation of the masks used in the off-peak period $\mathbb{T}_n$ should equal to the negative summation of all the masks used in the $n-1$ off-peak periods. The same requirement is applied to the masks of the on-peak and the mid-peak periods. Thus, the three requirements for the masks of interval $\mathbb{T}_n$ can be formulated as follows:

$$\begin{aligned}\text{Off-peak-masks at } (\mathbb{T}_n) &= -\sum_{1\leq k\leq l} m_i(t_k^n)\\ &= -\sum_{\substack{1\leq k\leq l\\ 1\leq m\leq n-1}} \sum_{\substack{j<i\\ 1\leq j\leq|\mathbb{SM}|}} \mathcal{H}(K_{i,j}\ ,\ t_k^m)\\ &\quad + \sum_{\substack{j>i\\ 1\leq j\leq|\mathbb{SM}|}} \mathcal{H}(K_{i,j}\ ,\ t_k^m)),\end{aligned}$$

$$\begin{aligned}\text{On-peak-masks at } (\mathbb{T}_n) &= -\sum_{l+1\leq k\leq 2l} m_i(t_k^n)\\ &= -\sum_{\substack{l+1\leq k\leq 2l\\ 1\leq m\leq n-1}} \sum_{\substack{j<i\\ 1\leq j\leq|\mathbb{SM}|}} \mathcal{H}(K_{i,j}\ ,\ t_k^m)\\ &\quad + \sum_{\substack{j>i\\ 1\leq j\leq|\mathbb{SM}|}} \mathcal{H}(K_{i,j}\ ,\ t_k^m)),\end{aligned}$$

$$\begin{aligned}\text{Mid-peak-masks } (\mathbb{T}_n) &= -\sum_{2l+1\leq k\leq 3l} m_i(t_k^n)\\ &= -\sum_{\substack{2l+1\leq k\leq 3l\\ 1\leq m\leq n-1}} \sum_{\substack{j<i\\ 1\leq j\leq|\mathbb{SM}|}} \mathcal{H}(K_{i,j}\ ,\ t_k^m)\\ &\quad + \sum_{\substack{j>i\\ 1\leq j\leq|\mathbb{SM}|}} \mathcal{H}(K_{i,j}\ ,\ t_k^m)).\end{aligned}$$

2) System Operator: During the billing interval, the SO stores all the masked readings received from SM_i, i.e., $h_i(t_k^m)$ where $1 \leq k \leq 3l$ and $1 \leq m \leq n-1$. Then, at the last interval $\mathbb{T}_n$, the SO receives masked readings for off-peak, on-peak, and mid-peak periods. By aggregating the masked readings of SM_i for each period at the end of the billing interval, the masks cancel each other and the total consumption can be obtained for each period, and then it can be multiplied by the time-use price to compute the bill.

By doing the above two steps, PPETD can achieve the functionality requirement (F2) of computing the consumers' bills following dynamic prices.

E. Privacy-Preserving Electricity Theft Detection

In this sub-section, we discuss how SO can detect electricity theft detection without violating the consumers' privacy, i.e., without learning the fine-grained power

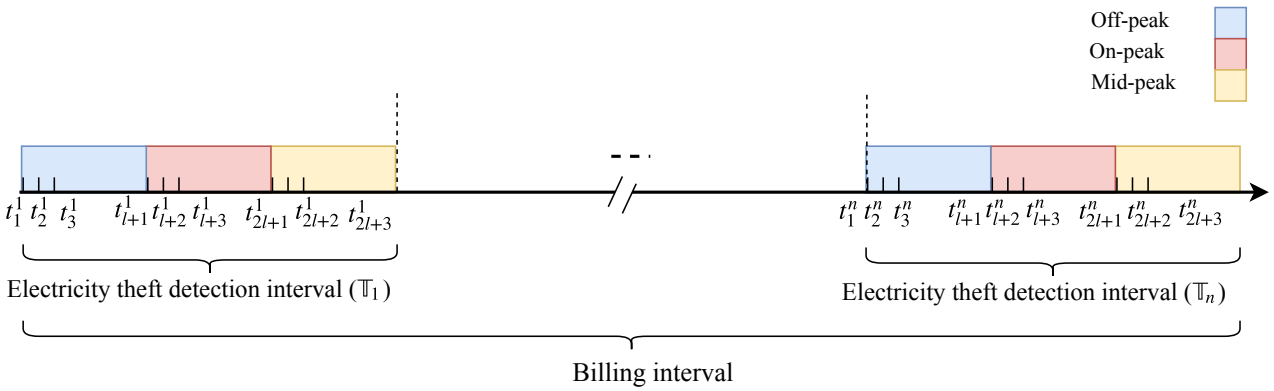


Fig. 3: Billing and electricity theft detection intervals.

consumption readings. Two building blocks are used by our scheme: secure addition/multiplication and evaluation of the non-linear activation function sigmoid(.). In this subsection, we first discuss the two building blocks and then discuss the construction used to detect electricity theft. Note that, unless otherwise stated, all arithmetic operations on the secret shares are in $\mathbb{Z}_q^*$.

1) Secure Addition and Multiplication: The main operation needed by a neural network's feed-forward layers can be expressed by $z = Wx + b$ where W is the weight matrix, x is the previous layer input, and b is the bias. Therefore, only additions and multiplications are needed to evaluate feed-forward layers. Inherently, SPDZ protocol offers secure addition and multiplication operations on the shared secret values. However, original SPDZ protocol was designed to work on single values, i.e., it was not designed for matrices. Inspired by recent works [44], [45], we use matrices' triplets to generalize the SPDZ protocol to operate on matrices rather than single values. The computation of the feed-forward layer secret shares of $Wx + b$ is given in Protocol 1. The triplets' matrices are defined as $c = Ay$, where A has the same dimension as W and y has the same dimension as x. In the protocol, each participant first computes $[\![U]\!] = [\![W]\!] - [\![A]\!]$ and $[\![v]\!] = [\![x]\!] - [\![y]\!]$. Then, in step 2, participants jointly reveal U and v. After that, the participants compute locally their shares z in steps 3 to 5. Note that, the new MAC values can be calculated using similar steps to Protocol 1 [43]. Subsequently, participants need to evaluate non-linear activation sigmoid(z) and share the result again securely between the participating parties.

2) Secure Sigmoid Evaluation: In neural networks, nonlinear activation functions are needed by feed-forward layers to support complex decisions [41]. We consider the support of a secure evaluation of non-linear activation function such as *sigmoid*(.) in our solution. The secure evaluation of *sigmoid*(.) can either be achieved using SPDZ protocol with polynomial approximation [46] or garbled circuits [44]. However, since SPDZ protocol can only support secure addition and multiplication, polynomial approximation usually needs a high degree polynomial to efficiently generate low error *sigmoid*(.) approximation, which may not be practical and may incur high computation and communication cost [44].

Protocol 1 Secure evaluation of feed-forward layers output.

Input: $[\![W]\!]$, $[\![x]\!]$, $[\![b]\!]$, $[\![A]\!]$, $[\![y]\!]$, $[\![c]\!]$
Output: $[\![Wx + b]\!]$

1: SM_i and SO computes $[\![U]\!] = [\![W]\!] - [\![A]\!]$ and $[\![v]\!] = [\![x]\!] - [\![y]\!]$.
2: SM_i and SO reveal U and v.
3: SM_i computes $[\![Wx]\!]_{SM_i} = U[\![y]\!]_{SM_i} + [\![A]\!]_{SM_i} v + [\![c]\!]_{SM_i}$
4: SO computes $[\![Wx]\!]_{SO} = Uv + U[\![y]\!]_{SO} + [\![A]\!]_{SO} v + [\![c]\!]_{SO}$
5: SM_i and SO computes $[\![z]\!] = [\![Wx + b]\!] = [\![Wx]\!] + [\![b]\!]$

Protocol 2 Secure evaluation of sigmoid activation.

Input: $[\![z]\!]$ and a random r from SO
Output: $[\![sigmoid^*(z)]\!]$

1: Reconstruct z form its shares.
2: Compute SM_i share as $[\![sigmoid^*(z)]\!]_{SM_i} = compare(z_1, z)\cdot(0-(a_1z+b1))+compare(z_2, z)\cdot((a_1z+b_1)(a_2z+b_2)) + \cdots + compare(z_{m-1}, z)\cdot((a_{m-1}z+b_{m-1})1) + 1 - r$
3: Compute SO share as $[\![sigmoid^*(z)]\!]_{SO} = r$

Instead of using polynomial approximation, we resort to piece-wise continuous linear approximation, i.e., splines [47] and garbled circuits. We followed the same method presented in [44] for sigmoid approximation and secure two-party evaluation. In particular, a smooth function $f(.)$ can be splitted into several intervals, in each, a straight line is used to approximate the interval. The approximated version of *sigmoid*(.) can be defined as follows:

$$sigmoid^*(z) = \begin{cases} 0 & \text{if } z < z_1 \\ a_1 z + b_1 & \text{if } z_1 \le z < z_2 \\ \dots & \\ a_{m-1} z + b_{m-1} & \text{if } z_{m-1} \le z < z_m \\ 1 & \text{otherwise.} \end{cases}$$

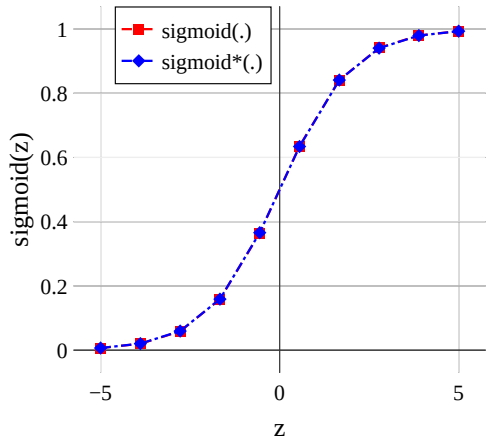


Fig. 4: *sigmoid*(.) function and its approximation using 25 spline points.

The slopes $a_{(.)}$, the intercepts $b_{(.)}$, and the values $z_{(.)}$ are chosen in such a way that the overall fitness is maximized. Figure 4 illustrates the approximation of the *sigmoid*(.) function using only 25 splines[1]. Protocol 2 gives the secure evaluation of $sigmoid^*(z)$ using garbled circuits. In step 1, the input z is reconstructed from its shares. Then, in step 2, the $sigmoid^*(z) - r$ is computed. Hence, the share of SM_i is $sigmoid^*(z) - r$ and the share of the SO is r as in step 3. The $compare(.,.)$ function, in step 2, is needed to determine which interval the input lies in and it outputs either zero or one. This function cannot be evaluated directly using SPDZ. Thus, garbled circuits are used for the evaluation of the $sigmoid^*(z)$.

From Figure 4 and Protocol 2, we note that the piecewise linear approximation method has small approximation error and it can be efficiently computed using garbled circuits two-party protocol. Note that, the new MAC value after the sigmoid evaluation can also be computed using garbled circuit by securely reconstructing the shared MAC key and calculating the new MAC value, and then sharing the result between the participants.

3) Our CNN Model Construction: In this subsection, we present the detailed construction of privacy-preserving CNN-based electricity theft detection. As shown in Figure 5, we use 1D CNN network that includes a convolutional layer, an average pooling layer, a fully connected layer, and a softmax output layer. The layers are described as follows.

- For input layer, 1D convolution is used where the shape of the input data is equal to the length of the electricity theft detection interval $\mathbb{T}$. The purpose of this layer is to capture the temporal correlation in the input sequence using a number of convolutional filters. The number of filters were determined from a pre-defined set using k-fold cross validation. Since convolution is merely a set of multiplications and additions, thus convolution layer can be evaluated securely using SPDZ protocol (i.e., Protocol 1).
- An average pooling layer is used to combine the features extracted from the previous layer. The reason for selecting average pooling is that it is friendly with the SPDZ protocol (only additions and multiplications, unlike max or min pooling) and can be computed using SPDZ protocol (i.e., Protocol 1).
- One fully connected hidden layer is used to extract more features and enable our model to make complex decisions while the loss in the performance due to the sigmoid approximation is minimized. Protocol 2 is used for the secure evaluation of this layer.
- Softmax output layer is used to classify the target (i.e., electricity theft). Since we only have two categories, electricity theft or not, the final output is of shape of two. Our scheme does not compute the softmax using secure two-party computation. However, we assume the input to the softmax layer once computed can be revealed and verified, and then the SO can compute each class probability.

[1]Python libraries scipy.interpolate.UnivariateSpline and numpy.polyfit are used to generate this approximation

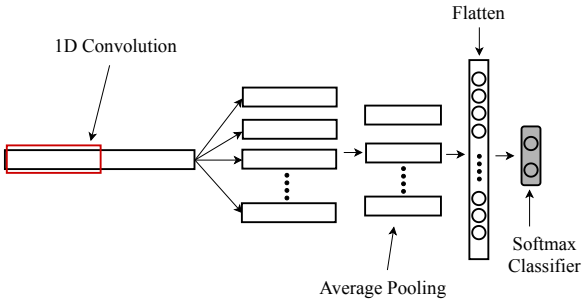


Fig. 5: 1D convolutional network model used.

The CNN model is evaluated securely by the SO and SM_i at the end of each electricity theft detection interval and the final MACs are used to verify the result. Therefore, PPETD can achieve the functionality requirement (F3) of privacy-preserving electricity theft detection.

Note that, despite the fact that recurrent neural networks (RNNs) can also be used to detect temporal correlations in the input sequence, secure RNNs evaluation can give low performance compared to secure CNNs due to the accumulated approximation error of the composite non-linear activation functions used by RNNs.

V. Security/Privacy Analysis

In this section, we discuss how our schemes can achieve the privacy-preservation requirements mentioned in Section II-C2.

A. Electricity theft Detection

In order to evaluate PPETD, we conduct the training on the high-performance cluster (HPC) of the Tennessee Tech University using two NVIDIA Tesla K80 GPUs. The privacy-preserving inference is conducted on two servers having Intel Core i7-4765T 2.00 GHz and 8 GB RAM

TABLE II: Electricity Theft Cyber Attacks

Cyber attacks in Jokar et al. 2016 [1]	Attack Type
$f_1(r_i(t_j^d)) = \alpha r_i(t_j^d)$	Partial Reduction
$f_2(r_i(t_j^d)) = \beta(d,t_j) r_i(t_j^d)$	Partial Reduction
$f_3(r_i(t_j^d)) = \begin{cases} 0 & \forall t \in [t_{\rm s}(d), t_{\rm f}(d)] \\ r_i(t_j^d) & \forall t \notin [t_{\rm s}(d), t_{\rm f}(d)] \end{cases}$	By-pass
$f_4(r_i(t_j^d)) = \mathbb{E}[r_i(t^d)]$	Partial Reduction
$f_5(r_i(t_j^d)) = \beta(d,t_j)\mathbb{E}[r_i(t^d)]$	By-pass/Partial Reduction
$f_6(r_i(t_j^d)) = r_i(t^d_{\lvert\mathbb{T}\rvert - j + 1})$	Price-based Load Control

connected on local LAN. To train our model, we used Python 3 libraries such as Pandas, Numpy, Tensorflow [48] and Keras [49]. For SPDZ protocol, we used Tensorflow-encrypted [50] which is a Python framework for privacy-preserving machine learning models. For garbled circuits, we used OblivC [?] which is a framework for Yao's garbled circuits. It is a compiler for secure two-party computation protocols, which incorporates most recent optimization techniques.

1) Experimental Data: Dataset: Real smart meter data from the Irish Smart Energy Trials [51] is used in our experiments. The dataset was published by the Electric Ireland and Sustainable Energy Authority of Ireland in January 2012. We used the energy consumption readings for $|\mathbb{SM}| = 200$ smart meters over 536 days between 2009−2010. We define a set of days $\mathbb{D} = \{1, \ldots, |\mathbb{D}|\}$, where a set $\mathbb{T}_d$ of electricity consumption readings are reported in each day d. In the dataset, the consumers report their readings every 30 minutes. Since we consider that each electricity theft interval is one day, the CNN input size $|\mathbb{T}_d|$ is 48. The total number of readings per consumer in the dataset is $25,728$. The total number of a SM's honest days of the 200 consumers is $107,200$ days.

Electricity Theft Attacks: It might not be easy to collect false readings sent by malicious SMs (electricity thieves) because the consumer reports may all be honest or it has not been detected previously as a malicious consumer.

To tackle this problem, a set of electricity theft attacks are defined in [1]. These attacks are used to create malicious dataset. Table II summarizes the attacks. Three types of attacks are considered: partial reduction, by-pass filters, and price-based load control. We denote $r_i(t_j^d)$ as the j^{th} electricity report of smart meter SM_i at day d. Each function $f(\cdot)$ creates a different attack scenario that aims to reduce the consumers' energy consumption report $r_i(t_j^d)$. The first attack $f_1(\cdot)$ aims to reduce $r_i(t_j^d)$ by some fraction, where α denotes a flat reduction ratio. On the other hand, attack $f_2(\cdot)$ dynamically reduces $r_i(t_j^d)$ by a value controlled by the time function $\beta(d, t_j)$. The third attack $f_3(\cdot)$ represents a selective time filtering function, where the malicious consumer reports zero readings during the interval $[t_{\rm i}(d), t_{\rm f}(d)]$, otherwise, the consumer reports the actual consumption $r_i(t_j^d)$. Here, $t_{\rm i}(d)$ and $t_{\rm f}(d)$ denote the initial and the final periods of the electricity theft interval, respectively. The next two attacks $f_4(\cdot)$ and $f_5(\cdot)$ are based on the expected value of the energy consumption of the malicious consumer for a given day, denoted by $\mathbb{E}[r_i(t^d)]$. In $f_4(\cdot)$, the attacker reports a flat value during the day, while in $f_5(\cdot)$, the attacker reduces $\mathbb{E}[r_i(t^d)]$ dynamically from time to time using function $\beta(d, t_j)$. The last attack $f_6(\cdot)$ is a reverse function that reorders the energy consumption readings during the day so that higher readings are reported during low tariff periods to reduce the electricity bill.

Data Pre-processing: To process the dataset, we first choose the parameters of the electricity theft attacks' functions. For functions $f_1(\cdot)$, $f_2(\cdot)$, and $f_5(\cdot)$, α and $\beta(d, t_j)$ are random variables that are uniformly distributed over the interval $[0.1, 0.6]$ [1]. For attack $f_3(\cdot)$, $t_{\rm s}(d)$ is a uniform random variable in $[0, 42]$, and the duration of the attack, i.e., $t_{\rm f}(d) - t_{\rm s}(d)$, is a uniform random variable in $[8, 48]$, and hence, the maximum value of $t_f(d) = 48$. Applying the electricity theft attack's functions on each SM's readings results in $\hat{X}_{SM_i}$ dataset which contains 536 honest samples (days) and $3,216$ malicious samples. Each sample in $\hat{X}_{SM_i}$ has 48 energy consumption values. For each $\hat{X}_{SM_i}$ dataset, adaptive synthetic sampling approach (ADASYN) [52] is performed to balance the size of honest and malicious classes. Consequently, the total size of each $\hat{X}_{SM_i}$ dataset is $6,432$ honest and malicious samples. The total number of samples for all $SM_i \in \mathbb{SM}$ is 1.2 million. Each $\hat{X}_{SM_i}$ dataset is partitioned into training set $\hat{X}_{SM_i,\rm tr}$ and testing set $\hat{X}_{SM_i,\rm tst}$ with ratio 5:4. The training sets for all SMs are merged together to form $\hat{X}_{\rm tr}$. Similarly, the test sets for all SMs are merged together to form $\hat{X}_{\rm tst}$. Feature scaling on training data produces $X_{\rm tr}$ and applying the same scores on the test data produces $X_{\rm tst}$.

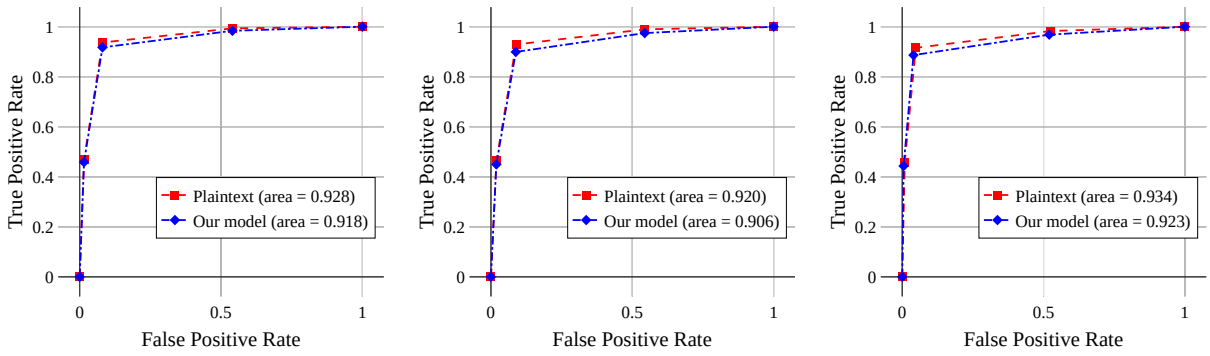
Performance Metrics: Three metrics are used to evaluate the performance of the detector defined as as follows. The detection rate (DR) measures the percentage of the correctly detected malicious consumers. The false acceptance (FA) rate measures the percentage of the honest samples that are falsely identified as malicious. The highest difference (HD) measures the difference between DR and FA [1].

$$DR = \frac{TP}{TP + FP}, \quad FA = \frac{FP}{TN + FP}, \quad HD = DR - FA$$

where, TP, FP and TN stands for true positive, false positive and true negative, respectively.

2) Results and Discussion: To select the hyper-parmaters of our CNN model, we used 3-fold cross validation on $X_{\rm tr}$ to tune the number of CNN filters and the number of neurons in the hidden layer. Then, the three top performance models are chosen to be evaluated on $X_{\rm tst}$. Note that, since we focus on the privacy of the online theft detection phase, we assume the model is either trained on an anonymized dataset or the training is done using privacy-preservation method such as differential privacy [?].

Finally, our baseline is the plaintext CNN model (without privacy preservation) and we compare it to our privacy-preserving model. We denote the top selected models as MD1 with "128 CNN filters, 1 stride size, 6 units filter size, and 2,048 hidden units"; MD2 with "256



(a) ROC for MD1 (b) ROC for MD2 (c) ROC for MD3

Fig. 6: ROC curves of MD1, MD2 and MD3, for the plaintext model and our privcay-presreving model.

TABLE III: The results of CNN models.

Model	Method	DR(%)	FA(%)	HD(%)	Accuracy(%)
MD1	Plaintext	93.6	8.00	85.6	93.2
	Our model	91.5	7.40	84.1	91.8
MD2	Plaintext	92.9	8.80	84.0	92.4
	Our model	90.0	8.79	81.2	90.2
MD3	Plaintext	91.5	4.80	86.7	92.4
	Our model	88.6	3.90	84.6	90.3
Jokar et al. 2016 [1]	Plaintext	94.0	11.0	83.0	–

CNN filters, 1 stride size, 5 units filter size, 1,536 hidden units"; and MD3 with "64 CNN filters, 1 stride size, 5 units filter size, 1,536 hidden units". Adam optimizer is used to train all models.

Table III gives the evaluation results obtained by each of the three models with and without privacy preservation. In addition, Figure 6 shows the Receiver Operating Characteristics (ROC) curves for the three models with and without privacy preservation. The given results indicate that the overall performance of our privacy-preserving model is comparable to the that of the plaintext model. In addition, our proposed detector is a general model that does not rely on a specific consumer's data. Hence, it is more robust against contamination attacks comparing to the proposed detector in [1].

Given that the cryptographic techniques used for the secure evaluation of the electricity theft detector, such as additive secret sharing [32], SPDZ protocol [43], and garbled circuits [?], are secure, the security of our scheme can be proved as follows. Firstly, before the secure evaluation of the electricity theft detector, the SM and the SO share the fine-grained readings and the weights of the CNN model. The security of additive secret sharing ensures that no entity can know the input of the other party. Subsequently, during the evaluation of CNN model, the consumer and SO use their secret shares locally for the secure addition and need communications for the secure multiplication and the evaluation of the non-linear activation functions. The MACs used during model evaluation ensure the integrity of the computations against the modifications of the active adversaries [43], and thus ensure the correctness of the final result. Therefore, PPETD can satisfy the security requirement of privacy-preserving theft detection (S3).

B. Resistance to Attacks

Our schemes can achieve the following desirable security/privacy requirements.

- Consumers' privacy preservation: The fine-grained power consumption readings of consumers are masked and no entity, including the SO, can access the readings to preserve consumers privacy. Note that we assume each SM shares a secret mask with each of all the SMs in the AMI network and the number of colluding SMs is always less than the total number of added masks. Only masked readings are reported by the SMs and they cannot reveal any information in the readings. Therefore, to get $r_i(t)$, an adversary has to collude with all other SMs in network to reveal $r_i(t)$ which is infeasible if the number of SMs is large enough [11]. Thus, PPETD satisfies the security requirement of privacy presrevation (S1).
- Aggregated power confidentiality: After receiving the masked fine-grained power consumption reports of the SMs, the SO can aggregate the reports to obtain the total power consumption for load monitoring. Since the reports sent from the SMs to the SO are encrypted, an external adversary cannot get any information about the total consumption. Thus, PPETD satisfies the security requirement of the aggregated power confidentiality (S2).
- Resistance to replay attacks: The time stamp used with each report from the SM to the SO prevents any adversary from intercepting and re-transmitting

TABLE IV: Computation cost of the used secure operations.

Secure Operation	Addition	Multiplication	Sigmoid
Time	0.081 μs	0.643 μs	1.34 s

SMs' reports claiming that they are fresh to disrupt the operation of our scheme.

- Resistance to impersonation attacks: The ID-based signature sent with each report from the SMs to the SO prevents any adversary from impersonating an SM in the AMI network.

VI. Computation and Communication Overhead

To measure the cryptography operations used by our scheme, we used Python charm cryptographic library [53] running on Raspberry Pi 3 device with 1.2 GHz Processor and 1 GB RAM. Supersingular elliptic curve with the asymmetric Type 3 pairing of size 160 bits (MNT159 curve) is used for pairing operations [?].

1) Computation Overhead: For fine-grained power consumption reporting, each SM needs to compute a mask to blind its reading $r_i(t)$ and compute a signature. Note that, the signature generation and verification operations take 0.34ms and 6.27ms, respectively. Therefore, the computation overhead on an SM to report a power consumption reading is 0.35ms. On the other hand, the computation overhead needed from the SO to verify each SM's reading is 6.27ms, while the overhead of aggregating 200 readings is 0.071 μs. These results are highly efficient and comparable to the literature [11], [35].

For privacy-preserving CNN model evaluation, Table IV measures the computation cost of each secure operation needed by our scheme. The operations needed include secure addition, secure multiplication, and secure $sigmoid^*(.)$ evaluation. It is shown that these operations are efficient. The total time needed to evaluate 2048 hidden units CNN model is around 48 minutes. This time is dominated by the evaluation of the garbled circuit of the hidden layer of the model. Moreover, this time is only needed once every day (at the end of the day, i.e., at the end of the electricity theft interval) for the purpose of the privacy preserving theft detection, which is acceptable cost for privacy.

2) Communication Overhead: For fine-grained power consumption reporting, each SM sends a masked report, signature, identity, and timestamp of total size of 56 bytes. For privacy-preserving CNN model evaluation, the overhead of the communication between the SM and the SO is only 256 bits for a single secure multiplication. The addition operation is done locally and no communication is needed. For the secure evaluation of $sigmoid^*(.)$, the overhead of the communication between the SM and the SO is about 850 KB. It can be concluded that the the overhead of $sigmoid^*(.)$ is dominated by the garbled circuit's online execution. Overall, the communication overhead in each electricity theft detection interval is around 1900 MB, which is practically acceptable cost for preserving the consumers' privacy.

VII. Related Work

A. Privacy-preserving electricity theft detection

Several schemes have addressed security and privacy issues in the literature [?], [?], [?], [?], [?], [?], [?], [?], [?]. However, very few works in the literature have tried to address privacy-preserving electricity theft detection [21]–[23]. Nevertheless, these approaches consider weak threat models that makes the privacy problem simple. Salinas et. al. [21], [22] is the first work that tried to study the privacy problem in electricity theft detection. In the proposed schemes, three peer-to-peer distributed algorithms have been proposed to solve a linear system of equations (LSE) for SMs' "honesty coefficients". Since the scheme is distributed, it fails if the SMs manipulate the messages submitted to other peers. In addition, the power line losses are needed before the execution of the scheme, which is not feasible in practice.

The proposed electricity theft detection algorithm in [23] uses a peer-to-peer state estimation approach using Kalman filter to detect irregularities in the readings reported by the customers. In the proposed algorithm, the SMs cooperate to privately estimate line segment currents and voltages. Then, these estimations are used by the SO to determine the malicious consumers. However, this work substantially differs from ours in two aspects. First, we use privacy-preserving machine learning to detect electricity theft, which usually outperforms state estimation approaches [1]. Second, the proposed state estimation uses a distributed algorithm between all SMs and assumes a semi-honest threat model, and thus, the scheme fails if the SMs manipulate the messages submitted to other peers.

In [29], a CNN model is used to detect electricity theft in smart grid. The proposed approach assumes that SMs should report their encrypted electricity consumption to two system entities, namely server gateway and gateway. The server gateway is assumed to be a fully trusted entity that can decrypt the consumer's fine-grained power consumption to run the CNN model and report the electricity theft result to the gateway. On the other hand, the gateway is responsible for aggregating the consumption of the consumers in a certain residential area and reporting the aggregated result to the utility for load management. Likewise, a trusted entity is assumed in [24]. In practice, it is impossible to guarantee that a party (that is assumed trusted) does not misuse the data of the consumers. A recent work that uses CNN model is presented in [12]. In Table V, we compare PPETD to the relevant schemes in the literature.

B. Privacy-preserving neural network inference

This paper also falls under secure neural network inference. In this category, the weights of a trained model is used for privacy-preserving inference [44], [55], [56]. However, the application setting of these works is different

TABLE V: Comparison between our scheme and the relevant schemes in the literature.

	Our scheme	[29]	[12]	[23]
Technique	CNN	CNN	CNN	state estimation
User privacy	√	√*	×	√**
Dynamic Billing	√	×	×	×
Grid Monitoring	√	√	√	√

∗ : Online trusted entity is assumed to run the machine learning model.
∗∗ : Semi-honest threat model is assumed.

from ours in the following aspects. Specifically, unlike our work that aims to compute securely a machine learning model to detect the electricity thefts in the AMI network, some of these works [55], [56] aim to optimize the secure neural network inference phase in general for hardware implementation. In addition, some of these works have different problem focuses and not application specific where they are customized for general secure multi-party evaluation of deep neural network. In contrast, in our application, we customize the secret sharing and the secure evaluation phases to serve other purposes of smart grid such as aggregation and dynamic billing.

VIII. Conclusion

In this paper, we have presented PPETD, a novel scheme that uses the fine-grained power consumption data reported by the consumers for electricity theft detection, load monitoring, and the computation of electricity bills based on dynamic pricing. To preserve consumers' privacy, no entity is able to access the fine-grained power consumption data of individual consumers. Secure protocols are executed by each consumer and the SO to evaluate a machine learning model using a set of masked power consumption data to detect electricity theft. SPDZ protocol is used to evaluate the neural network multiplications and additions while garbled circuits are used to evaluate the non-linear activation functions. Moreover, extensive simulations have been conducted on real dataset to evaluate our scheme. Simulation results indicate that our scheme can detect fraudulent users efficiently with acceptable communication and computation overhead. In specific, we compared the privacy-preserving electricity theft model against the plaintext and we found that a very slight loss in the detection rate performance which is considered an acceptable cost for privacy.

References

[1] P. Jokar, N. Arianpoo, and V. C. Leung, "Electricity theft detection in AMI using customers' consumption patterns," IEEE Transactions on Smart Grid, vol. 7, no. 1, pp. 216–226, 2016.

[2] IBM, "Energy theft: Incentives to change," Tech. rep, 2012.

[3] CBC, "Electricity theft by B.C. Grow-ops costs 100M a year," Last accesed: Nov 2017. [Online]. Available: http://www.cbc.ca/news/ canada/british-columbia/story/2010/10/08/bc-hydro-grow-optheftw.html

[4] S. K. Singh, R. Bose, and A. Joshi, "Entropy-based electricity theft detection in AMI network," IET Cyber-Physical Systems: Theory & Applications, 2017.

[5] P. Antmann, "Reducing technical and non-technical losses in the power sector," World Bank, Washington, DC, 2009.

[6] U.S. Department of Energy. (2008) The smart grid: An introduction. https://www.energy.gov/oe/downloads/smart-grid-introduction-0. [Online; accessed 27-OCT-2018].

[7] P. Palensky and D. Dietrich, "Demand side management: Demand response, intelligent energy systems, and smart loads," IEEE transactions on industrial informatics, vol. 7, no. 3, pp. 381–388, 2011.

[8] M. Sun, Y. Wang, G. Strbac, and C. Kang, "Probabilistic peak load estimation in smart cities using smart meter data," IEEE Transactions on Industrial Electronics, vol. 66, no. 2, pp. 1608–1618, 2019.

[9] Y. Wang, Q. Chen, T. Hong, and C. Kang, "Review of smart meter data analytics: Applications, methodologies, and challenges," IEEE Transactions on Smart Grid, 2018.

[10] Y. Wang, Q. Chen, C. Kang, and Q. Xia, "Clustering of electricity consumption behavior dynamics toward big data applications," IEEE transactions on smart grid, vol. 7, no. 5, pp. 2437–2447, 2016.

[11] A. Alsharif, M. Nabil, S. Tonyali, H. Mohammed, M. Mahmoud, and K. Akkaya, "EPIC: efficient privacy-preserving scheme with E2E data integrity and authenticity for ami networks," IEEE Internet of Things Journal, 2018.

[12] Z. Zheng, Y. Yang, X. Niu, H.-N. Dai, and Y. Zhou, "Wide and deep convolutional neural networks for electricity-theft detection to secure smart grids," IEEE Transactions on Industrial Informatics, vol. 14, no. 4, pp. 1606–1615, 2018.

[13] K. Zheng, Q. Chen, Y. Wang, C. Kang, and Q. Xia, "A novel combined data-driven approach for electricity theft detection," IEEE Transactions on Industrial Informatics, 2018.

[14] R. R. Bhat, R. D. Trevizan, R. Sengupta, X. Li, and A. Bretas, "Identifying nontechnical power loss via spatial and temporal deep learning," in Proc. of 15th IEEE International Conference on Machine Learning and Applications (ICMLA), 2016, pp. 272–279.

[15] M. Tariq and H. V. Poor, "Electricity theft detection and localization in grid-tied microgrids," IEEE Transactions on Smart Grid, 2016.

[16] Electronic Privacy Information Center (EPIC). The Smart Grid and Privacy. https://epic.org/privacy/smartgrid/smartgrid.html. [Online; accessed Nov. 30, 2018].

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

[18] 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.

[19] X. Li, X. Liang, R. Lu, X. Shen, X. Lin, and H. Zhu, "Securing smart grid: cyber attacks, countermeasures, and challenges," IEEE Communications Magazine, vol. 50, no. 8, 2012.

[20] K. Gajowniczek, T. Ząbkowski, and M. Sodenkamp, "Revealing household characteristics from electricity meter data with grade analysis and machine learning algorithms," Applied Sciences, vol. 8, no. 9, p. 1654, 2018.

[21] S. Salinas, M. Li, and P. Li, "Privacy-preserving energy theft detection in smart grids," in Sensor, Mesh and Ad Hoc Communications and Networks (SECON), 2012 9th Annual IEEE Communications Society Conference on. IEEE, 2012, pp. 605–613.

[22] ——, "Privacy-preserving energy theft detection in smart grids: A p2p computing approach," IEEE Journal on Selected Areas in Communications, vol. 31, no. 9, pp. 257–267, 2013.

[23] S. A. Salinas and P. Li, "Privacy-preserving energy theft detection in microgrids: A state estimation approach," IEEE Transactions on Power Systems, vol. 31, no. 2, pp. 883–894, 2016.

[24] C. Richardson, N. Race, and P. Smith, "A privacy preserving approach to energy theft detection in smart grids," in Smart Cities Conference (ISC2), 2016 IEEE International. IEEE, 2016, pp. 1–4.

[25] A. Nizar, Z. Dong, and Y. Wang, "Power utility nontechnical loss analysis with extreme learning machine method," IEEE Transactions on Power Systems, vol. 23, no. 3, pp. 946–955, 2008.

[26] J. Nagi, K. S. Yap, S. K. Tiong, S. K. Ahmed, and F. Nagi, "Improving SVM-based nontechnical loss detection in power

utility using the fuzzy inference system," IEEE Transactions on power delivery, vol. 26, no. 2, pp. 1284–1285, 2011.
[27] C. Ramos, A. de Sousa, J. Papa, and X. Falcao, "A new approach for nontechnical losses detection based on optimum-path forest," IEEE Transactions on Power Systems, vol. 26, no. 1, pp. 181–189, 2011.
[28] A. Jindal, A. Dua, K. Kaur, M. Singh, N. Kumar, and S. Mishra, "Decision tree and SVM-based data analytics for theft detection in smart grid," IEEE Transactions on Industrial Informatics, vol. 12, no. 3, pp. 1005–1016, 2016.
[29] D. Yao, M. Wen, X. Liang, Z. Fu, K. Zhang, and B. Yang, "Energy theft detection with energy privacy preservation in the smart grid," IEEE Internet of Things Journal, 2019.
[30] J. C. Choon and J. H. Cheon, "An identity-based signature from gap diffie-hellman groups," in International workshop on public key cryptography. Springer, 2003, pp. 18–30.
[31] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in Annual international cryptology conference. Springer, 2001, pp. 213–229.
[32] A. Shamir, "How to share a secret," Communications of the ACM, vol. 22, no. 11, pp. 612–613, 1979.
[33] M. Ben-Or, S. Goldwasser, and A. Wigderson, "Completeness theorems for non-cryptographic fault-tolerant distributed computation," in Proceedings of the twentieth annual ACM symposium on Theory of computing. ACM, 1988, pp. 1–10.
[34] D. Chaum, C. Crépeau, and I. Damgard, "Multiparty unconditionally secure protocols," in Proceedings of the twentieth annual ACM symposium on Theory of computing. ACM, 1988, pp. 11–19.
[35] H. Mohammed, S. Tonyali, K. Rabieh, M. Mahmoud, and K. Akkaya, "Efficient privacy-preserving data collection scheme for smart grid ami networks," in Global Communications Conference (GLOBECOM), 2016 IEEE. IEEE, 2016, pp. 1–6.
[36] I. Damgård, M. Keller, E. Larraia, V. Pastro, P. Scholl, and N. P. Smart, "Practical covertly secure mpc for dishonest majority–or: breaking the spdz limits," in European Symposium on Research in Computer Security. Springer, 2013, pp. 1–18.
[37] A. C.-C. Yao, "Protocols for secure computations," in FOCS, vol. 82, 1982, pp. 160–164.
[38] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in Advances in neural information processing systems, 2012, pp. 1097–1105.
[39] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," arXiv preprint arXiv:1404.2188, 2014.
[40] Y. LeCun, Y. Bengio et al., "Convolutional networks for images, speech, and time series," The handbook of brain theory and neural networks, vol. 3361, no. 10, p. 1995, 1995.
[41] J. Pennington, S. Schoenholz, and S. Ganguli, "Resurrecting the sigmoid in deep learning through dynamical isometry: theory and practice," in Advances in neural information processing systems, 2017, pp. 4788–4798.
[42] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in Proceedings of COMPSTAT'2010. Springer, 2010, pp. 177–186.
[43] M. Keller, V. Pastro, and D. Rotaru, "Overdrive: making spdz great again," in Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, 2018, pp. 158–189.
[44] J. Liu, M. Juuti, Y. Lu, and N. Asokan, "Oblivious neural network predictions via minionn transformations," in Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2017, pp. 619–631.
[45] V. Chen, V. Pastro, and M. Raykova, "Secure computation for machine learning with SPDZ," CoRR, vol. abs/1901.00329, 2019. [Online]. Available: http://arxiv.org/abs/1901.00329
[46] R. Livni, S. Shalev-Shwartz, and O. Shamir, "On the computational efficiency of training neural networks," in Advances in neural information processing systems, 2014, pp. 855–863.
[47] P. Dierckx, Curve and surface fitting with splines. Oxford University Press, 1995.
[48] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/
[49] F. Chollet et al., "Keras," https://github.com/fchollet/keras, 2015.
[50] D. Morten, B. DeCoste, Y. Dupis, M. Giraud, I. Livingstone, J. Mancuso, J. Patriquin, and A. Trask, "Tf encrypted," https://github.com/tf-encrypted/tf-encrypted, 2018.
[51] "Irish Social Science Data Archive," Last accesed: Nov 2017. [Online]. Available: http://www.ucd.ie/issda/data/commissionforenergyregulationcer/
[52] H. He, Y. Bai, E. A. Garcia, and S. Li, "Adasyn: Adaptive synthetic sampling approach for imbalanced learning," in Proc. of IEEE International Joint Conference on Computational Intelligence, 2008, pp. 1322–1328.
[53] 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.
[54] R. Gilad-Bachrach, N. Dowlin, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing, "Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy," in International Conference on Machine Learning, 2016, pp. 201–210.
[55] M. S. Riazi, C. Weinert, O. Tkachenko, E. M. Songhori, T. Schneider, and F. Koushanfar, "Chameleon: A hybrid secure computation framework for machine learning applications," in Proceedings of the 2018 on Asia Conference on Computer and Communications Security. ACM, 2018, pp. 707–721.
[56] B. D. Rouhani, M. S. Riazi, and F. Koushanfar, "Deepsecure: Scalable provably-secure deep learning," in Proceedings of the 55th Annual Design Automation Conference. ACM, 2018, p. 2.
[57] C. Juvekar, V. Vaikuntanathan, and A. Chandrakasan, "{GAZELLE}: A low latency framework for secure neural network inference," in 27th {USENIX} Security Symposium ({USENIX} Security 18), 2018, pp. 1651–1669.

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.

Muhammad Ismail (S'10-M'13-SM'17) received the B.Sc. (with Hons.) and M.Sc. degrees in Electrical Engineering (Electronics and Communications) from Ain Shams University, Cairo, Egypt, in 2007 and 2009, respectively, and the Ph.D. degree in Electrical and Computer Engineering from the University of Waterloo, Waterloo, ON, Canada, in 2013. From Dec. 2013 to Aug. 2019, Dr. Ismail worked as an Assistant Research Scientist at the Department of Electrical and Computer Engineering, Texas A&M University at Qatar. He is currently an Assistant Professor with the Department of Computer Science, Tennessee Tech. University, USA. His research interests include smart grids, wireless networks, and cyber-physical security. He was the co-recipient of the Best Paper Awards in the IEEE ICC 2014, the IEEE Globecom 2014, the SGRE 2015, the Green 2016, and the Best Conference Paper Award from the IEEE Communications Society Technical Committee on Green Communications and Networking at the IEEE ICC 2019. He is an Associate Editor for the IEEE TRANSACTIONS ON GREEN COMMUNICATIONS AND NETWORKING, IET Communications, and Elsevier PHYCOM. He was an Editorial Assistant for the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY from January 2011 to July 2013. He was a Workshop Co-Chair in IEEE Greencom 2018, TPC Co-Chair in IEEE VTC 2017 and 2016, Publicity and Publication Co-Chair in CROWNCOM 2015, and Web-Chair in IEEE INFOCOM 2014. He has been a Technical Reviewer for several IEEE conferences and journals.

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 of 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.

Waleed Alasmary received B.Sc. in Computer Engineering (first class honours) from Umm Al-Qura University, Saudi Arabia, in 2005, and the M.A.Sc in Electrical and Computer Engineering from University of Waterloo, Canada, in 2010, and his Ph.D. degree in Electrical and Computer Engineering from University of Toronto, Toronto, Canada, in 2015. Waleed subsequently joined the College of Computer and Information Systems in Umm Al-Qura University as an assistant professor of Computer Engineering. During his PhD, he was a visiting research scholar at the Network Research Laboratory at UCLA in the Spring of 2014. Dr. Alasmary is a Fulbright visiting scholar in the CSAIL laboratory at MIT during the academic year of 2016/2017. His "Mobility impact on IEEE 802.11p" article is among the Most Cited Ad Hoc Networks journal articles list. He published articles in prestigious conferences and journals such as IEEE Trans. on Vehicular Tech. and IEEE Comms. Surveys and Tutorials. His research interest includes mobile computing, ubiquitous sensing, and intelligent transportation systems.

Erchin Serpedin (F'13) is a professor with the Department of Electrical and Computer Engineering, Texas A&M University at Qatar. He received the specialization degree in transmission and processing of information from Ecole Superieure D'Electricite (SUPELEC), Paris, in 1992, the M.Sc. degree from the Georgia Institute of Technology in 1992, and the Ph.D. degree from the University of Virginia in January 1999. Dr. Serpedin is the author of two research monographs, one textbook, 15 book chapters, 150 journal papers, and 250 conference papers. His current research interests include signal processing, machine learning, cyber security, smart grids, bioinformatics, and wireless communications. Dr. Serpedin is currently serving as an associate editor of the IEEE Signal Processing Magazine. He served as an associate editor for more than 12 journals, including journals such as the IEEE Transactions on Information Theory, IEEE Transactions on Signal Processing, IEEE Transactions on Communications, IEEE Signal Processing Letters, IEEE Communications Letters, IEEE Transactions on Wireless Communications, Signal Processing (Elsevier), Physical Communications (Elsevier), EURASIP Journal on Advances in Signal Processing, and as a Technical Chair for six major conferences.