Towards Privacy-preserving Anomaly-based Attack Detection against Data Falsification in Smart Grid

Yu Ishimaki*, Shameek Bhattacharjee†, Hayato Yamana*, and Sajal K. Das‡
*Department of Computer Science and Communications Engineering, Waseda University, Tokyo, Japan
{yuishi, yamana}@yama.info.waseda.ac.jp
† Dept of Computer Science, Western Michigan University, Kalamazoo, USA (shameek.bhattacharjee@wmich.edu)
‡ Dept of Computer Science, Missouri University of Science and Technology, Rolla, USA (sdas@mst.edu)

***Abstract*—In this paper, we present a novel framework for privacy-preserving anomaly-based data falsification attack detection in a smart grid advanced metering infrastructure (AMI). Specifically, we propose an anomaly detection framework over homomorphically encrypted data. Unlike existing privacy-preserving anomaly detectors, our framework detects the presence of not only energy theft (i.e., *deductive attack*), but also more advanced data integrity attacks (i.e., *additive and camouflage attacks*) over encrypted data without diminishing detection sensitivity. We optimize the anomaly detection procedure such that potentially expensive operations over homomorphically encrypted space are avoided. Moreover, we optimize the encryption method designed for a resource constrained device such as smart meters, and the time to complete encryption gets 40x faster over the naïve adoption of the encryption method. We also validate the proposed framework using a real dataset from smart metering infrastructures, and demonstrate that the data integrity attacks can be detected with high sensitivity, without sacrificing user privacy. Experimental results with a real dataset of 200 houses from an AMI in Texas showed that the detection sensitivity of the plaintext algorithm is not degraded due to the use of homomorphic encryption.**

I. INTRODUCTION

In an advanced metering infrastructure (AMI), the fine-grained power consumption data collected from smart meters plays a central role in the energy optimization, efficiency, and operational reliability of the emerging smart grid technologies. Such meter data drives key operations such as automated billing, load forecast, and critical peak shifts. Additionally, data falsification attacks are launched by organized adversaries due to the immediate operational and economic impact of such attacks. Thus, attack detection mechanism in AMI is required by the utility to detect the presence of such attacks.

Since data driven attack detections require fine-grained power consumption data from customers', it violates privacy of each customer during such computations. The customer privacy has been known as a major concern [1]. Thus, it is indispensable to strike a balance between achieving privacy and security. Therefore, in this paper, we propose a framework for anomaly-based attack detection mechanism that is also privacy preserving at the same time.

978-1-7281-6127-3/20/$31.00

Privacy preservation can be accomplished in one of the following ways: (i) *differential privacy* (DP), (ii) *secure multiparty computation* (SMC), and (iii) *homomorphic encryption* (HE). First, Differential Privacy (DP) adds an adequate amount of noise to hide sensitive data. While it allows the quantification of privacy leakage, there exists a trade-off between the information leakage level and the accuracy of the statistics generated over noisy data. Thus, an exact computation cannot be achieved in order to guarantee the privacy level. Moreover, balancing the best trade-off is an open problem [1].

Second, in SMC based on secret sharing, each data reading is split into multiple pieces, called shares, which are sent to the other parties involved in the protocol, e.g., a set of computational servers operated by third-parties, so that adding all the shares yield the original value. While in this case, the smart meters can delegate a desirable computation to multiple servers in feasible time, it is assumed that *non-colluding servers* are operated by different third parties [2], [3]. The disadvantage of this approach is high management cost of two or more non-colluding servers, and the strong privacy assumption on non-collusion of third parties. Since each of the servers must communicate with each other to perform multiplications, this approach is more vulnerable to traffic analysis attacks.

Thirdly, compared with SMC, HE offers lower communication complexity, and requires a single computational server only rather than multiple non-colluding servers. Existing works such as data aggregation [4] and billing [5] are mostly based on Additive HE (AHE), which supports only addition and constant multiplication over encrypted data. However, highly sensitive data integrity attack detectors require more complex operations such as division and logarithms.

Motivation and Challenges: Our investigation of previous works reveals that none of the existing methods based on HE addresses the problem of privacy preserving data integrity attack detection which offers both security and privacy simultaneously. Unlike the traditional AHE, recently developed HE schemes enable both addition and multiplication of integers or real/complex numbers over encrypted data. This motivates us to consider the *feasibility of HE-compatible anomaly-based attack detection in a smart grid as a novel problem*. Among

previous anomaly detectors in AMI, the *harmonic to arithmetic mean* (HM-AM) ratio-based detector [6] has been shown to be an effective one against various attack types. However, it was designed under the assumption that the raw power consumption data from a set of different smart meters is visible throughout the computation, thus compromising customers' privacy completely. Thus, in this paper, we adopt HE to remodel a HM-AM ratio-based anomaly detection framework. Yet, the following three key challenges exist: (1) It is crucial to adopt an HE such that the attack detector preserves the detection sensitivity compared with the detector without HE. (2) The computation of the HM-AM ratio requires several HE-unfriendly operations, which makes entire system inefficient due to a large HE parameter setting. (3) A standard HE encryption process becomes expensive for resource-constrained Internet of Things (IoT) devices such as smart meters especially in our parameter setting.

Contributions:We make the following contributions:
(i) We develop a HE-based anomaly-based attack detection framework for AMIs, that detects attacks over encrypted smart meter data. The fine-grained smart meter readings of individual customers are not revealed to anyone at any stage. The computation of the ratio, requiring a set of fine-grained consumption, is performed at a computational server over encrypted data, followed by the ratio-based computation (in plaintext) for the anomaly detection performed at a utility. We demonstrate that the detection sensitivity is not degraded by the approximation error due to the use of HE.
(ii) We optimize the data encoding of power consumption for the ratio computation, to lower the HE ciphertext size.
(iii) We propose a simple, optimized user-side encryption procedure designed for low-power devices and achieve a 40x speed-up over a naïve method, both with the HE parameter set specific to the anomaly detection for a smart meter dataset.

The rest of this paper is organized as follows. Section II introduces backgrounds of HE and our base anomaly detector while Section III describes the system and threat models. Section IV presents HE-based privacy-preserving anomaly detection. Section V deals with security analysis and Section VI provides performance evaluation. Section VII discusses additional capability of our system, followed by conclusions.

II. Preliminaries

For symbols related to power consumption, the year, date, and timeslot are indicated in the subscript. For simplicity, we will omit the year and date. We use $\mathsf{Enc}(\cdot)$ and $\mathsf{Dec}(\cdot)$ to represent encryption and decryption of the HE, respectively. We respectively denote homomorphic addition and multiplication by $\boxplus$ and $\boxdot$.

A. The CKKS Scheme

The CKKS scheme [7] is an approximate HE scheme specifically designed for real/complex-number arithmetic over encrypted data. At a high level, a real/complex number m is expressed as a fixed-point integer, i.e., m is scaled up by a *scaling factor* $\Delta = 2^p \in \mathbb{Z}$ followed by rounding to its nearest integer, denoted by $m' := \lfloor \Delta m \rceil$. Besides, during encryption process, a small amount of noise e is added, forming a resulting ciphertext $\mathsf{Enc}(m')$ such that the decryption of the ciphertext produces the approximation of the underlying message. That is, $\mathsf{Dec}(\mathsf{Enc}(m')) = m' + e \approx \Delta m$, where the noise part is a source of both security and approximation error. We can obtain the above decryption result as long as a parameter called ciphertext modulus Q is larger than $m' + e$. Meanwhile, as a homomorphic multiplication exponentially increases the amount of message (and the noise), we may require Q to be exponentially large in the number of multiplications, which makes ciphertext size too large. To have Q small, the CKKS scheme maintains the growth by an operation called *rescaling*, that homomorphically scales down the underlying approximate message by reducing the ciphertext modulus. To obtain the correct decryption result, we need to set the initial ciphertext modulus large enough. Typically, Q is determined by the *multiplicative depth* which is the maximum consecutive number of homomorphic multiplications in a computation. Therefore, designing a function with less multiplicative depth is crucial to achieve efficiency in terms of both storage and computational costs.

B. Anomaly Detection Ratio Metric

The *harmonic to arithmetic mean* (HM-AM) ratio is a metric that has been recently demonstrated as an effective indicator for detecting anomalous behavior in smart metering data [6]. We use the HM-AM ratio as the anomaly-based attack detector because (i) it handles various attacks (described in the next section) simultaneously and (ii) it is sensitive to small margins of data falsification attack.

We denote by N the number of smart meters in a neighborhood area network in an AMI. Let $\mathcal{T}$ be a set of timeslots in a day. For each timeslot $t \in \mathcal{T}$, the i-th smart meter has power consumption $p_t^{(i)} \in \mathbb{R}^+$. To maintain the stability of the HM-AM ratio, each power consumption is transformed by taking natural logarithm, yielding $P_t^{(i)} := \ln(p_t^{(i)} + 2)$. For d-th date, the HM-AM ratio Q_d is computed by

$$Q_d = \frac{\sum_{t\in\mathcal{T}} \mathsf{HM}_t}{\sum_{t\in\mathcal{T}} \mathsf{AM}_t}, \tag{1}$$

where

$$\mathsf{AM}_t = \frac{\sum_{i=1}^{N} P_t^{(i)}}{N}, \quad \mathsf{HM}_t = \frac{N}{\sum_{i=1}^{N} \frac{1}{P_t^{(i)}}}. \tag{2}$$

Due to the page limitation, we only give a high-level description of it. (For further details, please refer to the paper [6].) The anomaly detection consists of training and test phases. During the training phase, the utility computes the ratio of daily HM to AM ratio, and establishes the normal range of difference between the observed ratio samples and a safe margin threshold parameter (residual) known, termed as the *standard limit* [6]. Subsequently, the test phase is done by checking that the

residual value derived from the given HM-AM ratios is within the standard limit range. Any residual outside of the standard limit is inferred as an attack, which requires a security audit.

III. System Goals

We show our architecture in Figure 1. Without loss of generality, we consider three types of stakeholders: a computational server (operated by a third-party), a utility, and N smart meters. The goal of our system is to securely perform anomaly-based attack detection without revealing the fine-grained power consumption data of the end user to the server and the utility. Namely, we enable the server to securely compute the HM-AM ratio from N homomorphically encrypted power consumption data at each timeslot, while ensuring that the server does not learn anything about each power consumption and the HM-AM ratio. Besides, the utility only learns the HM-AM ratio but not anything beyond what can be inferred from them. Using the HM-AM ratio, the utility performs the anomaly detection in the clear.

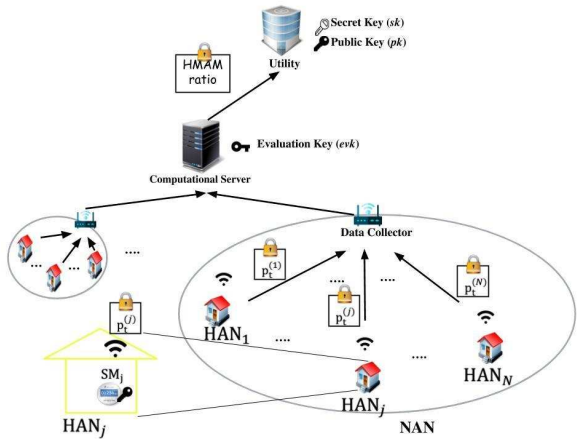


Figure 1. Overview of our system

Privacy Threat Model: Our system strives to protect the privacy of users against both the server and the utility. We assume that the all of the stakeholders are semi-honest (honest-but-curious), i.e., they follow the protocol but attempt to infer as much information as possible from the messages exchanged during the protocol. Moreover, we assume that the utility (the secret-key holder) and the server do not collude. The privacy requirement can be ensured as long as these assumptions hold. The detailed analysis is provided in Section V. Note that we do not restrict the collusion among smart meters and between smart meters and the server. That is, when the server and a subset of the smart meters are colluded, only the meter readings of the smart meters are revealed but not those of the other meters.

Security Threat Model: We consider the data integrity threat, which involves an adversary performing a data falsification attack on meter readings, assumed to occur before smart meters encrypt their power consumption. Specifically, we consider the attack model for AMI networks proposed in [6], [8]. It has four features that characterize the attack model: the fraction of compromised meters, the margin of false data, attack types, and the falsification strategy, as described below:

Compromised Meter Fraction: Let $M(\leqslant N)$ be the number of compromised meters and S_{all} be a set of compromised meter ids. The id_j-th ($id_j \in S_{\text{all}}$) smart meter in the area is compromised for each $j \in [M]$, such that the fraction of compromised meters is treated as a parameter ρ_{mal}.

Margin of False Data: The attacker samples M uniformly random variables $\{\delta_t^{(j)}\}_{j\in[M]}$, each from a discrete range $[\delta^{(\min)}, \delta^{(\max)}]$ for $\delta^{(\min)}, \delta^{(\max)} \in \mathbb{Z}$, whose steady-state average is denoted by δ_{avg}, quantifying the stealth of the attack.

Attack Types: We consider deductive, additive, and camouflage attacks. In *deductive attacks*, the consumption data is reduced from its original values to incur losses to the utility and errors in forecast. For $j \in [M]$, the compromised meter reading is given by $p_t^{(id_j)} - \delta_t^{(j)}$. Whereas, in *additive attacks*, it is $p_t^{(id_j)} + \delta_t^{(j)}$, intended to incur loss of business confidence that users experience owing to rival companies. In *camouflage attacks*, each half of the set of the compromised meters launch additive and deductive attacks simultaneously, with an intention to keep the mean consumption same, while favoring one set of customers at the expense of other half.

Falsification Distribution Strategy: We pick data order aware as it is the stealthiest of all strategies that has lowest deviation from the original data distribution compared to other known strategies such as scaling attacks. More specifically, the random variables $\{\delta_t^{(j)}\}_{j\in[M]}$ are sorted in ascending and descending orders for the *deductive* and *additive* attacks, respectively. We consider that the attacks occur between two timeslots, and the evaluation uses different attack start points and averages them to remove performance bias.

IV. Privacy-preserving Anomaly Detection

We propose a privacy-preserving anomaly detection by adopting HE to our previous work [6] based on the HM-AM ratio. The HM-AM ratio requires several HE-unfriendly operations. Thus, naïve adoption of the HE leads to inefficiency in terms of both storage/communication and computational costs. We optimize both encoding and encryption procedures to mitigate these. In what follows, we describe our protocol in sequence.

A. Initial Setup

The utility performs key generation and obtains a secret and public key pair $(\boldsymbol{sk}, \boldsymbol{pk})$ and an evaluation key $\boldsymbol{evk}$. Then, $\boldsymbol{pk}$ is installed manually in each smart meter which encrypts the power consumption data to form an HE ciphertext, while $\boldsymbol{evk}$ is sent from the utility to the computational server. $\boldsymbol{sk}$ is kept secret at the utility for the decryption of the ratio.

B. Data Encoding & Encryption via User-side Precomputation

At first glance, the ratio can homomorphically be computed from a set of encrypted power consumption $\mathsf{Enc}(p_t^{(j)})$. As shown in Eqn. (1) and (2), computation of the ratio consists

of a set of HE-unfriendly operations such as log-transformation for AM and its inverse for HM. Evaluating these over encrypted data is expensive because of the large multiplicative depth. Moreover, the encryption process gets expensive especially in a resource constrained IoT device such as smart meter due to the lack of computational resources. We address these problems.

Optimized Data Encoding: In Eqn. (2), the log-transformation and its inverse are applied on each power consumption. Thus, each of the smart meters can compute these operations locally in plaintext, and then encrypts three values. This can lower the HE parameter set compared with the naïve solution where the server homomorphically evaluates these operations. More specifically, the i-th smart meter locally performs natural log transformation, and computes its inverse, yielding $P_t^{(i)} = \ln(p_t^{(i)} + 2)$ and $P_t^{'(i)} = 1/P_t^{(i)}$, respectively. Then, three ciphertexts $\mathsf{Enc}(p_t^{(i)}), \mathsf{Enc}(P_t^{(i)})$ and $\mathsf{Enc}(P_t^{'(i)})$ are generated by the i-th smart meter for each $i \in [N]$.

Optimized Encryption: The encryption process of the CKKS scheme is broken into two steps: 1) generating an encryption of zero, and 2) adding a message to the encryption of zero. As later shown in Table II, 1) takes 97% of the encryption time. Meanwhile, it can be performed independently of the message. Thus, instead of adopting the CKKS encryption directly by performing the two steps after $p_t^{(i)}$ is known, our method first pre-computes the encryption of zero, before $p_t^{(i)}$ is known. Namely, the i-th smart meter first generates three encryptions of zero $z_0 \leftarrow \mathsf{Enc}(0), z_1 \leftarrow \mathsf{Enc}(0), z_2 \leftarrow \mathsf{Enc}(0)$. After the log transformations on $p_t^{(i)}$ are performed, the resulting ciphertexts are generated via $\mathsf{Enc}(p_t^{(i)}) \leftarrow z_0 + P_t^{(i)}$, $\mathsf{Enc}(P_t^{(i)}) \leftarrow z_1 + P_t^{(i)}$ and $\mathsf{Enc}(P_t^{'(i)}) \leftarrow z_2 + P_t^{'(i)}$. Note that each z_0, z_1 and z_2 must be re-generated after the encryption process is completed in every timeslot. This is because subtracting the two ciphertexts generated from the same encryption of zero reveals the difference between the two underlying messages.

C. Invariant Time Series Computation over Encrypted Data

For each date, the computational server homomorphically evaluates Eqn. (1) via Algorithm 1. We use the homomorphic division algorithm Inv presented in [9]. The encrypted ratio is sent to the utility.

Algorithm 1 Homomorphic Evaluation of the Daily Ratios

Input:
- Encrypted log power consumption in an area $\{\mathsf{Enc}(P_t^{(i)})\}_{i\in[N],t\in\mathcal{T}}$
- Encrypted inverse log power consumption in the area $\{\mathsf{Enc}(P_t^{'(i)})\}_{i\in[N],t\in\mathcal{T}}$

Output: Encrypted HM-AM ratio

$\mathsf{HM} \leftarrow 0$, $\mathsf{AM} \leftarrow 0$
for $t \in \mathcal{T}$ **do**
  $\mathsf{fracsum}_t \leftarrow 0$, $\mathsf{sum}_t \leftarrow 0$
  for $i \leftarrow 1$ to N **do**
    $\mathsf{sum}_t \leftarrow \mathsf{sum}_t \boxplus \mathsf{Enc}(P_t^{(i)})$
    $\mathsf{fracsum}_t \leftarrow \mathsf{fracsum}_t \boxplus \mathsf{Enc}(P_t^{'(i)})$
  end for
  $\mathsf{HM} \leftarrow \mathsf{HM} \boxplus (\mathsf{Inv}(\mathsf{fracsum}_t) \boxdot N)$
  $\mathsf{AM} \leftarrow \mathsf{AM} \boxplus (\mathsf{sum}_t \boxdot \frac{1}{N})$
end for
return $\mathsf{HM} \boxdot \mathsf{Inv}(\mathsf{AM})$

D. Anomaly Detection from the HM-AM ratio (in Plaintext)

Upon receiving the ciphertext, the utility decrypts it and learns the ratio Q_d. Then, the utility performs anomaly detection in plaintext. Algorithm 2 presents the overall protocol, except the initial setup.

Algorithm 2 Privacy-preserving Anomaly Detection Protocol

1) **Data Transmission**: In a year y on the d-th date at each timeslot t in an area, the i-th smart meter SM_i does the following:
   - Obtain the power consumption $p_{y,d,t}^{(i)}$
   - Compute $P_{y,d,t}^{(i)} := \ln(p_{y,d,t}^{(i)} + 2)$ and $P_{y,d,t}^{'(i)} := \frac{1}{P_{y,d,t}^{(i)}}$
   - Encrypt $p_{y,d,t}^{(i)}$, $P_{y,d,t}^{(i)}$ and $P_{y,d,t}^{'(i)}$ using pre-installed $\boldsymbol{pk}$
   - Send $\mathsf{Enc}(p_{y,d,t}^{(i)}), \mathsf{Enc}(P_{y,d,t}^{(i)})$ and $\mathsf{Enc}(P_{y,d,t}^{'(i)})$ to the computational server through a data collector in NAN and other higher-level network
2) **Homomorphic Evaluation of HM-AM Ratio:** Upon receiving the ciphertexts, the computational server does the following:
   - Call Algorithm 1 and obtain $\mathsf{Enc}(Q_d)$ if the daily amount of ciphertexts is available ($\mathsf{Enc}(p_{y,d,t}^{(i)})$ is used for other tasks as discussed in Section VII)
   - Send $\mathsf{Enc}(Q_d)$ to the utility
3) **Anomaly Detection:** The utility does the followings:
   - Decrypt $\mathsf{Enc}(Q_d)$ using $\boldsymbol{sk}$, and locally save Q_d
   - Perform training if a sufficient amount of Q_d's is available
   - Perform test if the training phase has been performed, and obtain a decision bit

V. Security Analysis

In our system, joint privacy and accuracy of attack detection must be ensured. We show how to guarantee privacy while also preserving the correctness of anomaly detection metrics, such that sensitivity to attack detection is not degraded. Since the correctness is ensured by setting the ciphertext modulus large enough which is described in the next section, we only describe the privacy analysis below.

We show that the privacy of fine-grained power consumption is preserved against both the computational server and the utility. Each smart meter sends $\mathsf{Enc}(p_t^{(i)}), \mathsf{Enc}(P_t^{(i)})$ and $\mathsf{Enc}(P_t^{'(i)})$ at each timeslot to the computational server. The server calls Algorithm 1 to homomorphically evaluate the HM-AM ratio for each date, and the resulting ciphertext is sent to the utility. Subsequently, the utility decrypts and learns the HM-AM ratio. As each of the ciphertexts is generated under the semantically secure CKKS encryption scheme, no information about the underlying messages is revealed to the semi-honest server. The HM does not have a closed-form expression itself [6]; thus, its ratio with AM makes it impossible to infer the individual customer consumption as the number of participating meters becomes more than three. Therefore, it is reasonable to conclude that the individual privacy is preserved against the semi-honest utility and the computational server as long as they do not collude.

VI. Performance Evaluation

In this section, we examine the feasibility of our protocol with a real dataset in terms of accuracy and efficiency. First, we compare the accuracy of our protocol with that of the method in plaintext (i.e., non-private). Then, we report the runtime at each user and the one at the server, and the ciphertext sizes.

A. Experimental Setup

We used a smart grid dataset collected from Pecan Street Project[1], which consists of a dataset from 200 households in Texas, USA over three years (2014–2016). The number of timeslots per day is 24. The data in 2014–2015 was used for the training phase, wheres the data in 2016 was used for the test phase. For the data pre-processing, we performed winsorizing as in the original work [6] by setting the lower and upper limits of each power consumption to 50W and 6000W, respectively. That is, values smaller (resp. greater) than 50 (resp. 6000) are replaced by 50 (resp. 6000).

We implemented our system in C++ using g++ 7.4.0 with "-O2" option. We used Raspberry Pi 3 equipped with an ARMv7 processor (1.2 GHz), 1GB RAM, and 64GB SD card for an experiment on user-side computation. The other experiments about the computational server and the utility were ran on a machine running Ubuntu 16.04, equipped with Intel Xeon E5-1620 v4 3.50 GHz in single-threaded mode.

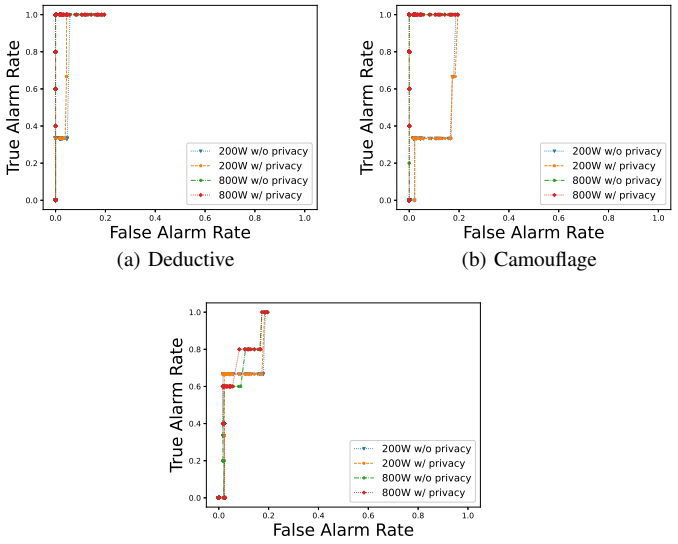
We used the HEAAN library[2]; an open-source implementation of the CKKS scheme, and chose a parameter set as $(n, \log Q, p) = (2^{15}, 491, 35)$, in which a fresh ciphertext size is calculated as $2n \log Q$ bits. The other parameters were default values such that the parameter set provides the correctness and an 80-bit security level according to the online security estimator [10]. Let L be the multiplicative depth. To maintain the precision and message growth throughout the computation, the ciphertext is *rescaled* after every homomorphic multiplication. As a result, we chose $\log Q = L \cdot 2^p + (2^{p+1})$ such that the size of the ciphertext the utility obtains is $2n(p+1)$ bits, i.e., during decryption, the utility can obtain a noisy HM-AM ratio as a $(p+1)$-bit fixed-point integer with the integral part and the fractional part being 1-bit and p-bit, respectively. Algorithm 1 requires $L = 13$ mostly due to the Inv operation. We found p from the experiment.

B. Attack Detection Accuracy and Correctness

We compare the detection accuracy from anomaly detection with and without HE via receiver operating characteristics (ROC) curve. We calculate the ROC by tuning different *standard limits*, and then examine trade-off between false alarms and true attack detection as a way of evaluating the performance of the anomaly detector. Specifically, we attempt to show how close our HE-based anomaly detector performs with its plaintext counterpart and show the extent of performance degradation across varying thresholds. Fig. 2(a) shows the ROC curve for deductive attack with two extreme δ_{avg} of 200W and 800W to show that the sensitivity scales well across margins. Similarly, Fig. 2(b) and (c) respectively show the same for the camouflage and additive attack. From the close proximity of both ROC curves with and without HE for all attack types, it is shown that the attack detection performance is preserved under HE.

[1]https://www.pecanstreet.org/

[2]https://github.com/kimandrik/HEAAN



(a) Deductive (b) Camouflage

(c) Additive

Figure 2. Sensitivity of Anomaly Detection Performance

C. Computation and Communication Performance Results

We show how our framework performs for smart metering, in terms of *time to encryption completion*, the computation time, and the data size given that the HE is known to be computational and resource-intensive.

1) User-side Computation Speed-up: Table I shows the complexity reduction of the cost by our method from the naïve generation of the CKKS ciphertext. It is known that any subroutine in the CKKS is bounded by polynomial multiplications. With our technique, all necessary polynomial multiplications for the encryption are offloaded in the pre-computation phase, and ciphertext generation can be done with only a single polynomial addition.

We measured the actual runtime for encrypting three ciphertexts by each household at each timeslot to assess the time to encryption completion. We compared the timing result of the naïve CKKS encryption method with that of our proposed method. Table II shows the speed-up of the timing result for the encryption completion at the user side.

Table I
RUNTIME COMPLEXITY FOR USER-SIDE ENCRYPTION

Method	# Poly. Mult.	# Poly. Add.
CKKS $\mathsf{Enc}(m)$	2	3
Pre-computation ($\mathsf{Enc}(0)$)	2	2
Optimized Enc ($\mathsf{Enc}(0) + m$)	**0**	**1**

Table II
RUNTIME FOR USER-SIDE ENCRYPTION COMPLETION IN MILLISECOND

Method	RAM	SD Card
CKKS $\mathsf{Enc}(m)$	4525	N/A
Pre-computation ($\mathsf{Enc}(0)$)	4512	7964
Optimized Enc ($\mathsf{Enc}(0) + m$)	**112**	**4151**

In our experiment, we have two options to store the encryption of zero; viz., a RAM or SD card in the Raspberry Pi. The primary difference between using the RAM and SD card is the existence of a file I/O. With the SD card, the pre-computed

$\mathsf{Enc}(0)$ is written. Once the message m is known, it is added to $\mathsf{Enc}(0)$ read from the card. The file write took 3479ms while its read took 4040ms, indicating almost the same cost as the original $\mathsf{Enc}(m)$ with the RAM. The usage of the RAM shows that the pre-computation takes most of the time in the entire encryption process while the addition of m is cheap. Therefore, our encryption time is accelerated especially with the RAM. As smart meters are typically online in the smart grid, using the RAM-based pre-computation is a valid optimization.

2) Computation Time at Server and Utility: We measured two different computational time at the server, and the one at the utility, as shown in Table III. First, server side computation time per timeslot was measured, i.e., average computation time in each iteration of the outer loop in Algorithm 1. Next, we measured time for computing the last iteration of the outer loop followed by completing the ratio computation. Finally, the time for decrypting the HM-AM ratio at the utility was measured. Because the minimum requirement of AMI is to perform the test phase of the anomaly detection on a daily basis, we achieved significantly faster protocol.

Table III
ANOMALY DETECTOR TURNAROUND TIME IN SECOND

Step	Running Time
Server Computation / timeslot	9.935
Server Computation (last timeslot)	18.342
Utility Decryption	0.022

3) Ciphertext Size: Table IV shows concrete cost of ciphertexts size. In our protocol, CKKS ciphertexts are transferred in the following paths: 1) from each household to the server and 2) from the server to the utility. The size of the CKKS ciphertexts gets smaller as the *rescaling* is performed. Therefore, the size of ciphertexts from the sever to the utility is smaller than that from each smart meter to the server.

Table IV
CIPHERTEXT SIZE FOR ANOMALY DETECTION

A Household $\rightarrow$ Server	$3 \cdot 2 \cdot 2^{15} \cdot 491$ bit = 11,784KB
Server $\rightarrow$ Utility	$2 \cdot 2^{15} \cdot 36$ bit = 288KB

VII. DISCUSSION ON SMART GRID FUNCTIONALITY

While we showed the applicability of the anomaly detection in a smart gird in a privacy-preserving manner, the primary requirements in a smart grid are automated billing and load monitoring. In this section, we discuss the capability of our proposed system to these tasks:

Load Monitoring: Load monitoring and forecasting are performed at different scales, and data aggregation (homomorphic addition) is sufficient for these tasks.

Automated Billing: Automated billing is performed for individual customers. It can be accomplished via data aggregation of power consumption from a specific customer over timeslots (e.g., per day). Thus, simple homomorphic addition is sufficient. Our CKKS parameter is capable of handling these tasks since they do not require homomorphic multiplications.

Handling Each Tasks in Different Sectors: Moreover, we provide another option in case each task is performed in different organizations or different sectors of the utility, i.e., the anomaly detection center C_1, load monitoring center C_2, and the billing center C_3. Accordingly, we consider a setting where the secret key is generated by a *cryptographic service provider (CSP)* instead of the utility, which also does not collude with the server. The only difference from the system in Figure 1 is that each center will receive its appropriate value from both the server and the CSP. For example, suppose the server holds $\mathsf{Enc}(r_1)$ and C_1 is willing to obtain r_1. The server first generates a uniformly random number r_1' in some integer ring $\mathbb{Z}_q$ that the CKKS scheme natively supports. It sends $\mathsf{Enc}(r_1 - r_1')$ and r_1' to the CSP, and C_1, respectively. Then the CSP decrypts and sends $r_1 - r_1' \mod q$ to C_1. Finally, C_1 only knows r_1 by adding the two received values over $\mathbb{Z}_q$.

VIII. CONCLUSION

In this paper, we developed a privacy-preserving anomaly-based attack detection framework using HE, that does not sacrifice the functionality of smart grid and sensitivity of the anomaly detector. The quantification of the information leakage of the fine-grained power consumption from the HM-AM ratio will be part of our future work.

Acknowledgments: This work was supported by JST CREST Grant #JPMJCR1503, Japan-US Network Opportunity (JUNO2) funded by NICT and NSF; and also by NSF grants DGE-1433659, CNS-1818942, CNS-2030611, CNS-2030624.

REFERENCES

[1] M. R. Asghar, G. Dán, D. Miorandi, and I. Chlamtac, "Smart Meter Data Privacy: A Survey," *IEEE Communications Surveys Tutorials*, vol. 19, no. 4, pp. 2820–2835, 2017.

[2] G. Danezis, C. Fournet, M. Kohlweiss, and S. Zanella-Béguelin, "Smart meter aggregation via secret-sharing," in *Proceedings of the First ACM Workshop on Smart Energy Grid Security*. ACM, 2013, pp. 75–80.

[3] M. A. Mustafa, S. Cleemput, A. Aly, and A. Abidin, "A secure and privacy-preserving protocol for smart metering operational data collection," *IEEE Transactions on Smart Grid*, 2019, (To Appear).

[4] F. Li, B. Luo, and P. Liu, "Secure information aggregation for smart grids using homomorphic encryption," in *Proceedings of IEEE International Conference on Smart Grid Communications*, 2010, pp. 327–332.

[5] X. Liang, X. Li, R. Lu, X. Lin, and X. Shen, "UDP: Usage-Based Dynamic Pricing With Privacy Preservation for Smart Grid," *IEEE Transactions on Smart Grid*, vol. 4, no. 1, pp. 141–150, 2013.

[6] S. Bhattacharjee and S. K. Das, "Detection and Forensics against Stealthy Data Falsification in Smart Metering Infrastructure," *IEEE Transactions on Dependable and Secure Computing*, vol. to appear, 2020.

[7] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *Advances in Cryptology – ASIACRYPT 2017, Proceedings, Part I*, vol. 10624 of LNCS, 2017, pp. 409–437.

[8] S. Bhattacharjee, A. Thakur, and S. K. Das, "Towards fast and semi-supervised identification of smart meters launching data falsification attacks," in *Proc. of ACM ASIA CCS*, ser. ASIACCS, 2018, pp. 173–185.

[9] J. H. Cheon, D. Kim, D. Kim, H. H. Lee, and K. Lee, "Numerical Method for Comparison on Homomorphically Encrypted Numbers," in *Advances in Cryptology – ASIACRYPT 2019*, vol. 11922 of LNCS. Springer International Publishing, 2019, pp. 415–445.

[10] M. Albrecht, R. Player, and S. Scott, "On the concrete hardness of Learning with Errors," *Journal of Mathematical Cryptology*, vol. 9, no. 3, pp. 169–203, 2015.