Detecting False Data Injection in Smart Grid In-Network Aggregation

Lei Yang and Fengjun Li
Department of EECS, The University of Kansas, Lawrence, KS, 66045

***Abstract*—The core of the smart grid relies on the ability of transmitting realtime metering data and control commands efficiently and reliably. Secure in-network data aggregation approaches have been introduced to fulfill the goal in smart grid neighborhood area networks (NANs) by aggregating the data on-the-fly via intermediate meters. To protect users' privacy from being learnt from the fine-grained consumption data by the utilities or other third-party services, homomorphic encryption schemes have been adopted. Hence, intermediate smart meters participate in the aggregation without seeing any individual reading, nor intermediate or final aggregation results. However, the malleable property of homomorphic encryption operations makes it difficult to identify misbehaving meters from which false data can be injected through accidental errors or malicious attacks. In this paper, we propose an efficient anomaly detection scheme based on dynamic grouping and data re-encryption, which is compatible with existing secure in-network aggregation schemes, to detect falsified data injected by malfunctioning and malicious meters.**

I. Introduction

Envisioned as the next-generation power grid, the smart grid is the modernization of the existing power grid with advanced bidirectional communication and pervasive computing capabilities. The fundamental goal is to introduce intelligent electricity generation, distribution, consumption and management into the conventional power systems. An essential component of the smart grid is the two-way communication infrastructure connecting energy consumers and suppliers for more fine-grained meter readings, real-time status reports, dynamic pricing and control. However, along with all the advantages, the smart grid with improved communication and computation capabilities inevitably introduces new security and privacy risks [1], [2]. With forged power consumption data or electricity price, attackers can remotely turn on or off electronic devices in target households or trigger imbalanced power supplement, causing power outages and tremendous damages [3]. Moreover, fine-grained usage data collected by smart meters contains rich information about energy consumers. For example, personal information such as current location or distance traveled can be revealed to the smart meter when charging an electric vehicle [4]; usage patterns of electric appliances derived from high-frequency energy consumption data can be utilized in user profiling attacks to depict a consumer's demand profile [5]. Such information becomes the primary target of adversaries who might launch sophisticated attacks, such as eavesdropping the communication or compromising smart meters to access the power consumption data of the victim. Thus, it is critical to transmit metering data from distributed smart meters to utility's control center in a secure and privacy-preserving manner.

In-network aggregation is widely adopted in wireless sensor networks and proven to be an important primitive to reduce transporting overhead [6]–[9]. Recently, it is extended to smart grid NANs to solve the secure data gathering problem. Many secure in-network aggregation protocols have been proposed to efficiently route metering data through a set of smart meters to the collector device of the utility [10]–[13]. Aggregation operations (e.g., SUM, AVG, etc.) are performed at each intermediate meter on-the-fly, and thus naturally anonymize individual meter readings in the aggregated results. For end-to-end security, homomorphic encryption is adopted. Aggregation operations are directly applied over the encrypted data to ensure that smart meters participating in the aggregation cannot view intermediate or final aggregation results [10], [11].

However, most of the protocols take a *prevention*-based approach to ensure confidentiality, integrity and authenticity in in-network aggregations. Encipherment and digital signature techniques are used to prevent the adversary from eavesdropping or altering messages. Security of such approach heavily relies on the assumption that the adversary can not break cryptography system, but it neglects the fact that the adversary can steal all the associated secret keys to insert or alter output aggregation results to further tamper with critical smart grid functions such as load balancing and smart pricing.

Meanwhile, accidental errors may be incorporated into the aggregation result by malfunctioning smart meters or unreliable wireless transmission channel. Unfortunately, there is very little work that aims at addressing this problem in approaches other than prevention. To the best of our knowledge, [14] was among the first to examine this problem with an attestation-based solution to support an incremental integrity check to verify the aggregate-so-far results based on homomorphic signatures. It also suggested to incorporate anomaly detection at the collector to identify irregular data values that may be potentially caused by accidental errors or malicious attacks. However, the centralized temporal outlier detection approach requires the collector to store a series of individual metering data over time, which is communication and storage costly. More importantly, it violates the initial objective of adopting in-network aggregation – to hide the fine-grained individual metering data from all the involved parities including the collector device that reports to the utility.

In this paper, we propose a distributed outlier localization scheme. Before data aggregation from the individual meters to

978-1-4799-1526-2/13/$31.00 ©2013 IEEE

the collector, the aggregation tree is partitioned into multiple logical groups through dynamic grouping. Each root stores a queue of historical data for the member meters in the group. Once the collector detects any abnormal in the final aggregation result, it recruits multiple roots to identify the malicious meter within their corresponding sub-tree using a non-parametric-based approach in a distributed manner. One extremely challenging issue with this design is that to verify the validity of each meter reading, the root has to decrypt stored historical data in the queue, which would inevitably require key update after the outlier localization for the sake of privacy. Our scheme only requires sub-trees under suspect to participant in the outlier localization progress and exploits re-encryption to avoid system-wide key update.

The rest of this paper is organized as follows: we summarize related work in Section II, introduce network model, threat model and secure in-network aggregation approach in Section III, and present the main schemes of our solution in Section IV. We evaluate the proposed mechanisms in Section V, and finally, conclude the paper in Section VI.

II. Related Work

Data aggregation is a critical operation in smart grids. Motivated by the in-network aggregation solutions in wireless sensor networks, several aggregation protocols using additively homomorphic encryption schemes [10], [11], [15] have been presented to protect end-to-end data confidentiality and privacy against malicious or "curious" meters en route. Meanwhile, several authentication protocols for smart grid in-network aggregation have been proposed using conventional PKI-based digital signatures [16], [17] or short signature schemes based on bilinear maps [13]. However, these solutions are either not compatible with the privacy-preserving in-network data aggregation or introduce excessive hop-by-hop verification overhead. Recently, Li et al. proposed an authentication scheme that supported batch verification based on a homomorphic signature scheme [14]. However, all these solutions adopt prevention-based approaches, which can detect false data injected by network errors or external attackers with no knowledge of the secrets associated with the encryption and signature schemes. The focus of our work is on a solution for efficient detection of falsified data that are injected into smart grid data aggregation by compromised or malfunctioning meters. To the best of our knowledge, this is the first work proposing a solution of distributed anomaly detection that can be integrated with privacy-preserving in-network aggregation.

III. Background and Motivation

A. Network Model

Wireless mesh has been widely accepted as a promising communication infrastructure for home area networks and neighborhood area networks by most US utilities. In this work, we consider a NAN consisting of hundreds of meter nodes (e.g., $\{N_1, N_2, \cdots, N_t\}$ in Fig. 1) and a collector node (e.g., N_0) which further connects to a utility's wide area network. Similar as the other in-network aggregation approaches for smart grid data collection, a spanning tree (e.g., BFST [10], [11] or MST [13]) is constructed to include all meters in NAN into a logical aggregation tree (as shown in Fig. 1). When the collector initiates an aggregation query, each smart meter will first perform the specified aggregation operation over the inputs from its child nodes, and then submit the aggregated result to its parent node.

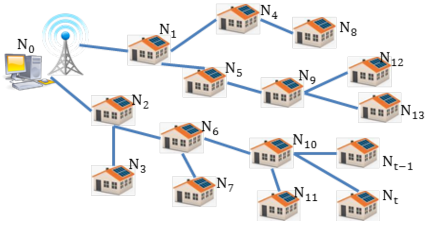

Fig. 1. An example of in-network aggregation in NAN.

B. Secure In-network Aggregation

Privacy-preserving aggregation: As aforementioned, to prevent an intermediate meter from seeing plaintext inputs of its children, homomorphic cryptosystems that support arithmetic operations on the ciphertext domain are employed to encrypt the messages. For most of the aggregation tasks, additive-homomorphism is expected. The addition of two messages can be obtained by directly performing ciphertext additive operation to the encrypted messages followed by a decryption transformation. For example, Paillier cryptosystem [18] was employed in [10] for privacy-preserving in-network aggregation. In particular, [10] assumed a pair of asymmetric aggregation keys $< PK_{agg}, SK_{agg} >$, where each meter node encrypted individual metering data as $C_i = Enc(m_i, PK_{agg})$. The aggregation result ($\prod_{i=1\to t} C_i$) in ciphertext domain can be represented as an encryption transformation of the addition of all messages m_is, and it can only be decrypted by the collector with the private aggregation key SK_{agg}.

Signature-based authentication: External attackers and malicious smart meters can tamper the in-network aggregation with falsified readings. To protect the integrity of the metering data, several signature-based approaches have been proposed [12]–[14]. [14] presented a homomorphic signature algorithm based on a short signature scheme using bilinear pairing. It assumed all the smart meters share a same private signing key SK_{sig} to generate the signature σ_i for message m_i. Such homomorphic signatures can be aggregated in a similar way (e.g., $\prod_{i=1\to t} \sigma_i$) along with the in-network aggregation of the ciphertext messages. In the end, the collector can use the final aggregation signature to perform a batch verification to check the integrity of the final aggregation result.

C. Threat Model

The secure aggregation approaches introduced above can effectively defend against two types of attackers: (1) the *weak external attackers* who can eavesdrop or alter the messages transmitted without knowing the secrets associated with the aggregation or authentication, and (2) the *honest-but-curious internal attackers* who properly follow the protocol with an attempt to sniff confidential data from the relayed messages.

However, when a *strong external attacker* compromises a smart meter, he can take full control of the node. With all the associated secrets, the attacker can insert falsified data or alter the aggregation output. Such falsified outputs are properly encrypted by the homomorphic encryption key and attached with valid signatures, and thus it is extremely difficult, if not impossible, to be detected by the prevention-based approaches. Meanwhile, abnormal readings may be generated by *malfunctioning meters*. Similar as the falsified data, it is indistinguishable from the regular readings after being encrypted. As the goal of the attack is to significantly disrupt the aggregation operations with falsified inputs, it is reasonable to assume the falsified data transmitted by a compromised meter is significantly different from the actual values. Although the attacker might insert a value slightly deviating from the true value to avoid being detected, such falsified data has less impact on the target application.

D. Challenges and our solution

In this work, we aim to efficiently detect falsified data that is intentionally or accidentally inserted into in-network aggregation and further identify the compromised or malfunctioning smart meter. As the metering data is collected at a high frequency, we believe the actual readings from consecutive observations should be highly correlated in time domain. Therefore, we propose an *extended kernel density estimator based mechanism* to detect false inserted data as temporal outliers.

Kernel density estimator [19] originally used for anomaly detection in wireless sensor networks (WSN) cannot be directly applied to smart grid in-network aggregation. Data in WSN is in the cleartext form and accessible by the detector node (i.e., the sink or neighboring sensor nodes) directly, however in our aggregation scenario, data is encrypted and aggregated from neighbor nodes as well as the intermediate nodes. Moreover, even if the collector is allowed to recover the final aggregation result, it is restricted to see the individual metering data as it is hidden by the aggregation.

Therefore, we design a *revised aggregation scheme* to support the transmission and storage of individual metering data at selected verifiers as time-series data for detection while not breaking the security and privacy promises, and a *re-encryption scheme* to preprocess the encrypted data on-demand for the detector where the kernel density estimator is deployed. To reduce the computation and communication overhead at the collector, we also propose a *light-weight dynamic grouping scheme* to divide the aggregation tree into connected logical groups and employ the root as the verifier for each group. Details of the schemes will be explained in Section IV.

IV. Proposed Solutions

In this section, we propose an efficient and reliable anomaly detection protocol for secure in-network aggregation in smart grid NAN. It consists of a *light-weight grouping* scheme, a revised *in-network aggregation and in-group incremental verification* scheme, a *data re-encryption* scheme, and an *group-based anomaly detection* scheme.

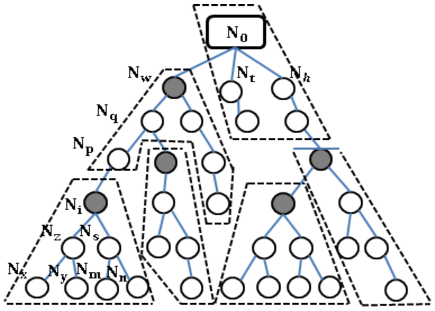

Fig. 2. An example of dynamic grouping: the dotted areas are groups and the dark nodes are the root.

A. Light-weight dynamic grouping

Previous signature-based authentication protocols [12], [14] are centralized approaches in which the collector is assumed to verify the authenticity of final and intermediate aggregation results. Since intermediate aggregation results are not transmitted to the collector in the aggregation, the incremental verification that checks the integrity of aggregate-so-far results will inevitably increase the computation and communication overhead at the collector. If we further incorporate anomaly detection at the collector, the overhead will become intolerant. As such, we employ a light-weigh dynamic grouping scheme as shown in Fig. 2, which is also widely used in WSN (e.g., [20]), to divide the aggregation tree into multiple groups for distributed verification and anomaly detection.

The idea is to employ a grouping function F such that each node can calculate its probability of becoming a root with its own topological variables and compare with a pre-defined threshold T_r to decide if it is selected as the root of a group. For example, in Fig. 2, node N_i calculates $F_i > T_r$, it then determines itself as the root and all other nodes below itself but not grouped yet as its group members. The grouping decisions are made in a bottom-up fashion. Once the root of a group is determined, an intra-group ID is assigned to each member.

For the grouping, we expect all the groups are of approximately a same size to balance the verification load. We need to store the historical metering data for all group members at the root, therefore a small group size is desired. Considering the communication overhead of multihop transmission, although the transmission frequency is much smaller than the aggregation frequency, we still prefer a short tree. With all these considerations, we suggest a grouping function

$$F_i = \left(1 - \frac{1}{e^{\left(\alpha\theta^{h_i} + \beta n_i\right)}}\right)^{\gamma}$$

where n_i and h_i are the number of current members and the current height of a node i, respectively. α, β, θ, and γ are the parameters to indicate the impact of n_i and h_i to the grouping decision. For a preferred grouping function, when n_i and h_i are too large or too small it should vary slowly, but when n_i and h_i approach to the ideal group size and height, it should give strong indication. Therefore, if we select the ideal $n = 10$ and $h = 3$, the suggested parameter setting is $\alpha = 0.06, \beta = 0.3, \gamma = 6, \theta = 3$. By adjusting the value of these parameters, we can affect the formation of the group.

With the newly constructed logical groups, we expect to distribute the verification and detection load from the collector to the root of the groups. So, the root needs to store time-series data for each group member. The data will be collected during regular aggregations. Since the data is still encrypted with the public aggregation key, if the root does not collude with the collector (to disclose the data or obtain the secret aggregation key), the privacy of individual metering data is retained. When the root receives a request for verification, it first launches an in-group incremental verification based on the aggregation signatures, similar as in [14]. If the intermediate aggregation result is proven to be valid, it calls for in-group anomaly detection to further examine the input of each member. Before sending the to-be-examined input and the historical data to the selected detector node, the data needs to be preprocessed (i.e., re-encrypted) so that it can be recovered by the detector (we will explain this in section IV.C).

B. Revised in-network aggregation

Secure in-network aggregation requires both the encryption algorithm and the signature algorithm are additive homomorphism. In the data pre-processing for anomaly detection, we further require the encrypted data supports re-encryption operation to allow the verifier (i.e., the root of the group) to transform the metering data that is originally encrypted under the public aggregation key into a form that can be recovered by the detector's private key. In this work, we use an ElGamal-based scheme to achieve additive homomorphism, which can also be integrated with the collusion resistant re-encryption scheme [21] based on bilinear maps.

Here we briefly explain the encryption and re-encryption schemes: for an additive cyclic group $\mathbb{G}$ and a multiplicative cyclic group $\mathbb{G}_T$ of prime order q with a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, select a random generator $g \in \mathbb{G}$ of prime order q and $Z = e(g, g) \in \mathbb{G}_T$.

Key Generation

1) Select a random $a \in \mathbb{Z}_q^*$ to generate the aggregation key pair $< SK_{agg} = a, PK_{agg} = g^a >$;
2) Select a random $s \in \mathbb{Z}_q^*$ to generate the signature key pair $< SK_{sig} = s, PK_{sig} = g^s >$;
3) For a detector d_i, select a random $v_i \in \mathbb{Z}_q^*$ to generate the detection key pair $< SK_d = v_i, PK_d = g^{v_i} >$;
4) Generate the re-encryption key $RK_{agg\to d} = g^{v_i/a} \in \mathbb{G}$.

Encryption

1) Select a random $r \in \mathbb{Z}_q^*$;
2) For a message m, map m to $M = Z^m \in \mathbb{G}_T$;
3) Encrypt the message with PK_{agg}: $C = (Z^r \cdot M, g^{ra})$.

Decryption

1) Decrypt the ciphertext C with SK_{agg}: compute $Z^r \cdot M/e(g^{ra}, g^{1/a}) = Z^r \cdot M/e(g, g)^r = Z^r \cdot M/Z^r$;
2) Reversely map $M \in \mathbb{G}_T$ back to m.

Re-encryption

1) With $RK_{agg\to d_i} = g^{v_i/a}$, any root can change a ciphertext under PK_{agg} into a ciphertext for a detector d_i: from $C = (Z^r \cdot M, g^{ra})$, compute $e(g^{ra}, g^{v_i/a}) = e(g, g)^{rv_i} = Z^{rv_i}$;
2) The new ciphertext $C' = (Z^r \cdot M, Z^{rv_i})$ can be decrypted by d_i under v_i as $M = Z^r \cdot M/(Z^{rv_i})^{1/v_i}$.

Signing

1) For a plaintext m, create the signature $\sigma = (g^m)^s$.

Verification

1) Check if $e(\sigma, g) = e(g^m, g^s) = e(g^m, PK_{sig})$ based on the bilinearity property.

With the above scheme, we slightly change the in-network aggregation and data signing process in [14] to make it fit into our scenario. Here we briefly illustrate the process with an example: as shown in Fig. 2, the leaf node N_x encrypts its own reading into C_x, generates the signature σ_x for m_x, and sends the tuple $< C_x, C_{o_x}, \sigma_{o_x} >$ to the parent meter N_z. Here, as N_x is the leaf node, we have $C_{o_x} = C_x$ and $\sigma_{o_x} = \sigma_x$.

The intermediate node N_z gets the to-be-aggregate ciphertexts C_x, C_{o_x}, C_y and C_{o_y} as well as the corresponding signatures from its children. Therefore, N_z can compute $C_{o_z} = C_x \cdot C_y \cdot C_{o_x} \cdot C_{o_y}$ and $\sigma_{o_z} = \sigma_z \cdot \sigma_{o_x} \cdot \sigma_{o_y}$, and outputs the tuple $< C_z, C_{o_z}, \sigma_{o_z} >$ to node N_i. Note that, for anomaly detection purpose, each intermediate node needs to store the individual data from its child nodes for a pre-defined period of time.

So on so forth, when the collector receives the tuple $< C_w, C_{o_w}, \sigma_{o_w} >$ from N_w and the ones from N_t and N_h, it aggregates the final result $C = C_w C_{o_w} \cdots C_h C_{o_h}$ and $\sigma = \sigma_{o_w}\sigma_{o_t}\sigma_{o_h}$. After decrypting C to get cleartext m the collector checks the integrity using batch verification as follows:

$$e(\sigma, g) \overset{?}{=} e\left(g^m, PK_{sig}\right)$$

If the batch verification succeeds, the collector will report the aggregated data in the NAN to the utility. However, if the aggregated result significantly deviates from the normal range, it indicates some falsified data has been inserted. Thus, it will follow the incremental verification scheme [14] to identify logical groups with invalid outputs.

C. Group-based anomaly detection

Through the incremental verification, we can identify the groups whose intermediate aggregate result has been tampered by the falsified data in a top-down pattern. Once locating such group, we will call the group-based anomaly detection to examine the input from each group member. The anomaly detection is *pairwise*: it involves the root of the group, serving as the *verifier*, and a randomly selected neighbor node with light-weight kernel density estimator installed, serving as the *detector*.

1) Data collected at the verifier: According to the revised data aggregation scheme, individual metering data of each member node is transmitted to the group verifier, and stored in a fixed-size queue. It creates a random sample set R for the values collected in a recent period of time t where $|R| = t$, and will be used as historical data in anomaly detection. Accordingly, the size of the verifier's dataset T is $t * gSize$, where $gSize$ is the group size.

At each aggregation round, it is assumed to update the historical data with the latest value. However, as the metering data is collected at a high frequency, it may not be necessary to update the data frequently. Moreover, per-round update will cause undesired communication overhead that we want to avoid by in-network aggregation. Therefore, we propose a slow update scheme that allows only one member node to submit an update per aggregation round. Hence, the communication overhead caused by anomaly detection is only doubled (i.e., two output readings at each node) compared to the original aggregation scheme. In particular, at round n, every node computes $(n \mod gSize)$. Only the node whose intra-group ID matches $(n \mod gSize)$ is allowed to send the update in terms of C_i in the output tuple.

2) Data Re-encryption: To identify the abnormal data, we need to analyze the historical data distribution, which is encrypted under the aggregation key. However, it is absolutely insecure for the collector to share the secret aggregation key a with the verifier, not only because this will allow the verifier to view all the historical readings of its group members, but more importantly, the verifier can abuse this secret to recover all future encrypted data in the aggregation. Key refreshing may also not be a good solution, as it will lead to inconsistency in the encryption form of the historical data (i.e., part of the data is encrypted by PK_{agg} and part of them is encrypted by new aggregation key), and further causes difficulty to key management.

Therefore, we propose a data re-encryption scheme: the verifier (with the help of the collector) randomly selects a neighbor node, say N_d, as its detector, and obtains the re-encryption key $RK_{agg \to d_i} = g^{v_i/a}$ from the collector. It then performs the aforementioned re-encryption operation over the data stream (e.g., data stream of N_p at time t_p is $\{C_p(t_p - t), \cdots, C_p(t_p - t - i), ..., C_p(t_p - 1)\}$) where $C_p(t_p - i)$s are node N_p's t ciphertexts before time t_p, and sends the re-encrypted stream to the detector, which can recover $M_p = \{m_p(t_p - t), ...m_p(t_p - 1)\} = \{m_{p_1}, ..., m_{p_t}\}$ with its private detection key v_i. The security of individual metering data is guaranteed: for the verifier that stores the data, it will never be chosen as detector and thus it cannot obtain a re-encryption key for itself; for the detector, although it can recover the individual metering data in t rounds, but it doesn't know to whom the data belongs. Therefore, we believe the security and privacy is well maintained in the re-encryption approach.

3) Group-based anomaly detection: We consider the metering data in smart grid aggregation as a temporal streaming data without spatial correlation. That is, data distribution changes only over time, and we neglect the similarity in neighboring meter data associated with different households. We further assume the time when anomaly occurs is unpredictable, that is, we don't have priori knowledge about data distribution at a given time. Therefore, we need an anomaly detection scheme that can efficiently model distribution for streaming data and effectively approximate an unknown data distribution. It should also be light-weight, which is computationally efficient and requires very small memory.

In this work, we adopt the outlier detection model used wireless sensor networks [19] and modify it to fit in our scenario. This model uses a distance-based anomaly definition: "*a point p in a dataset T is a (D, r)-anomaly if less then D points in T lie within distance r from p.*" It approximates the data distribution in dataset T based on the *kernel density estimator*, and then computes the density of the data space around the value which needs to be detected. If the number of neighboring data is less than D, the distance-based outlier is identified. We briefly introduce the method as follows.

The detector first maps the sample set $R = M_p = (m_{p_1}, \cdots, m_{p_t})$ into the interval $[0, 1]$. Let $k_p(x)$ be the kernel function for N_p, where $\int_0^1 k_p(x)\, dx = 1$ for all values in M_p. The underlying distribution $f_p(x)$ for T can be approximated according to the values in the sample set M_p by the following function

$$f_p(x) = \frac{1}{t} \sum_{m_{p_j} \in M_p} k\left(x - m_{p_j}\right).$$

Let us select the Epanechnikov kernel that is easy to integrate as the kernel function, then we have

$$k(x_p) = \begin{cases} \frac{3}{4}\frac{1}{B}\left(1 - \left(\frac{x_p}{B}\right)^2\right) & , \quad \left|\frac{x_p}{B}\right| < 1 \\ 0 & , \quad otherwise \end{cases}$$

where B is the bandwidth of the kernel function. With a similar setting as in [19], we set B as $\sqrt{5}\sigma \left|M_p\right|^{-\frac{1}{5}}$, where σ is the standard deviation of the values in M_p. With the distribution function $f_p(x)$ for N_p, the detector estimates the number of values that are within the distance r from m_p as

$$N(m_p, r) = \int_0^r f_p(x)\, d_x.$$

If this number is less than the threshold D, m_p is identified as an outlier.

V. Evaluation

To support distributed group-based anomaly detection, the proposed solution requires the root of each group to store individual data for its members and preprocess the historical data for the detector when necessary. The root is randomly selected in the dynamic grouping, so it can be any smart meter in the system. Therefore, we need to evaluate the additional computation and storage overhead that the group-based anomaly detection introduces to the root.

Computation overhead: In each aggregation round, a member node needs to perform one encryption, one aggregation, and one signing operation, while the group root needs to do an additional re-encryption and the detector needs to do an additional decryption over the re-encrypted data. We implemented the schemes with Java pairing-based cryptography (JPBC) library on a 2.8 GHz processor PC. The implementation chooses a 160-bit order elliptic curve group $\mathbb{G}$ based on $y^2 = x^3 + x$ over a 512-bit finite field. We measured the processing time for each operation as below:

Operation	ENC	DEC	AGG	Re-encryption	DEC (for Re-Enc)
Time (ms)	15.89	23.61	0.09	10.04	1.4

Storage overhead: As the verifier of the anomaly detection scheme, the root of each group needs to store a queue of individual metering data for the recent t rounds for each group member. A larger t indicates a higher accuracy in anomaly detection. However, t should be bounded by the storage capacity of the smart meter. Given the average group size as 10 (adjustable by the parameters and threshold in the grouping function), in order to achieve a reasonably high detection accuracy (more than 90%), as also assumed in [19], the size of the sample set (the queue length) is 1024, and thus the dataset size for each member node is 10240. Since each ciphertext size is 128B in our experiment, the total memory needed for storage is 1.3×10^6B. In practice, a communication module of a smart meter [22] has 4MB RAM and 8MB flash memory. Therefore, only 16% memory is required for storing the historical data, and the storage overhead introduced by the anomaly detection is far below the capability of the current smart meter.

Communication overhead: In the revised aggregation scheme, we let the group members periodically send their individual readings to the root of the group to update their historical data. Hence, only one member node reports in each aggregation round. This individual data is included in the aggregation message, resulting in the increase of message size by one ciphertext size (e.g., 128B). Consider the historical data will be transmitted to the detector. If we distribute this overhead to each aggregation round, it indicates a cost of one additional message from the group root to the detector. Therefore, the overall communication overhead will be $2N * 128B$, in a network of N smart meters.

VI. Conclusion

Existing prevention based secure in-network information aggregation mechanism for smart grid systems cannot effectively detect accidental errors and falsified data injection by malfunctioning or compromised meters. In this paper, we first introduce a light-weigh anomaly detector based on kernel density estimator to localize false data injected into the the aggregation. To reduce the overhead at the collector, we design a dynamic grouping scheme to divide meters into multiple connected groups and distribute the verification and detection load among the root of the groups. A novel data re-encryption scheme based on bilinear mapping is further proposed to transform the data previously encrypted under the aggregation key into a form that can be computed by the detector to ensure the security and privacy of individual metering data, which are critical for anomaly detection. Performance of the anomaly detector is evaluated in terms of the memory usage and communication overhead, and proved to be light-weight for the current smart meter configuration.

References

[1] P. McDaniel and S. McLaughlin, “Security and privacy challenges in the smart grid,” *Security & Privacy, IEEE*, vol. 7, no. 3, 2009.

[2] H. Khurana, M. Hadley, N. Lu, and D. Frincke, “Smart-grid security issues,” *Security & Privacy, IEEE*, vol. 8, no. 1, pp. 81 –85, 2010.

[3] X. Li, X. Liang, R. Lu, X. Shen, X. Lin, and H. Zhu, “Securing smart grid: cyber attacks, countermeasures, and challenges,” *Communications Magazine, IEEE*, vol. 50, no. 8, pp. 38–45, 2012.

[4] K. Budka, J. Deshpande, J. Hobby, Y.-J. Kim, V. Kolesnikov, W. Lee, T. Reddington, M. Thottan, C. White, J.-I. Choi, J. Hong, J. Kim, W. Ko, Y.-W. Nam, and S.-Y. Sohn, “Geri - bell labs smart grid research focus: Economic modeling, networking, and security and privacy,” in *IEEE SmartGridComm 2010*, oct. 2010, pp. 208 –213.

[5] G. Wood and M. Newborough, “Dynamic energy-consumption indicators for domestic appliances: environment, behaviour and design,” *Energy and Buildings*, vol. 35, no. 8, pp. 821 – 841, 2003.

[6] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, “Tag: a tiny aggregation service for ad-hoc sensor networks,” *SIGOPS Oper. Syst. Rev.*, vol. 36, no. SI, pp. 131–146, 2002.

[7] B. Krishnamachari, D. Estrin, and S. B. Wicker, “The impact of data aggregation in wireless sensor networks,” in *ICDCSW ’02*.

[8] H. Chan, A. Perrig, and D. Song, “Secure hierarchical in-network aggregation in sensor networks,” in *ACM CCS ’06*.

[9] K. B. Frikken and J. A. Dougherty, IV, “An efficient integrity-preserving scheme for hierarchical sensor aggregation,” in *WiSec ’08*.

[10] F. Li, B. Luo, and P. Liu, “Secure information aggregation for smart grids using homomorphic encryption,” in *SmartGridComm, 2010 First IEEE International Conference on*, oct. 2010, pp. 327 –332.

[11] ——, “Secure and privacy-preserving information aggregation for smart grids,” *Int. J. Secur. Netw.*, vol. 6, no. 1, pp. 28–39, 2011.

[12] M. Fouda, Z. Fadlullah, N. Kato, R. Lu, and X. Shen, “A lightweight message authentication scheme for smart grid communications,” *Smart Grid, IEEE Transactions on*, vol. 2, no. 4, pp. 675–685, 2011.

[13] D. Li, Z. Aung, J. R. Williams, and A. Sanchez, “Efficient authentication scheme for data aggregation in smart grid with fault tolerance and fault diagnosis,” in *Innovative Smart Grid Technologies, 2012 IEEE PES*.

[14] F. Li and B. Luo, “Preserving data integrity for smart grid data aggregation,” in *Smart Grid Communications (SmartGridComm), 2012 IEEE Third International Conference on*, 2012, pp. 366–371.

[15] F. D. Garcia and B. Jacobs, “Privacy-friendly energy-metering via homomorphic encryption,” in *Proceedings of the 6th international conference on Security and trust management*, ser. STM’10, 2011, pp. 226–238.

[16] Y. Yan, Y. Qian, and H. Sharif, “A secure and reliable in-network collaborative communication scheme for advanced metering infrastructure in smart grid,” in *Wireless Communications and Networking Conference (WCNC), 2011 IEEE*, 2011, pp. 909 – 914.

[17] P. Deng and L. Yang, “A secure and privacy-preserving communication scheme for advanced metering infrastructure,” *Innovative Smart Grid Technologies, IEEE PES*, vol. 0, pp. 1–5, 2012.

[18] P. Paillier, “Public-key cryptosystem based on composite degree residuosity classes,” in *Proceedings of Eurocrypt ’99*, 1999, pp. 223–238.

[19] S. Subramaniam, T. Palpanas, D. Papadopoulos, V. Kalogeraki, and D. Gunopulos, “Online outlier detection in sensor data using non-parametric models,” in *VLDB*.

[20] Y. Yang, X. Wang, S. Zhu, and G. Cao, “Sdap: a secure hop-by-hop data aggregation protocol for sensor networks,” in *Proceedings of the 7th ACM international symposium on Mobile ad hoc networking and computing*.

[21] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, “Improved proxy re-encryption schemes with applications to secure distributed storage,” *ACM Trans. Inf. Syst. Secur.*, vol. 9, no. 1, Feb. 2006.

[22] “Communications module for electricity meters,” http://www.silverspringnet.com/pdfs/SilverSpring-Datasheet-Communications-Modules.pdf.