QATAR UNIVERSITY

COLLEGE OF ENGINEERING

# PRIVACY-PRESERVING DATA AGGREGATION IN SMART POWER GRID SYSTEMS

BY

FAWAZ AHMAD KSERAWI

A Thesis Submitted to

the College of Engineering

in Partial Fulfillment of the Requirements for the Degree of

Master of Science in Computing

June 2021

# COMMITTEE PAGE

The members of the Committee approve the Thesis of

Fawaz Ahmad Kserawi defended on 19/04/2021.

Dr. Qutaibah m. Malluhi
Thesis Supervisor

Dr. Ali Ghrayeb
Committee Member

Dr. Amr Mohamed
Committee Member

Dr. Khaled Khan
Committee Member

Approved:

Khalid Kamal Naji, Dean, College of Engineering

# ABSTRACT

Name, Fawaz Ahmad Kserawi, Masters : June: 2021, Master of Science in Computing

Title: Privacy-Preserving Data Aggregation in Smart Power Grid Systems

Supervisor of Thesis: Dr. Qutaibah m. Malluhi.

Smart Meters (SMs) are IoT end devices used to collect user utility consumption with limited processing power on the edge of the smart grid (SG). While SMs have significant applications in providing data analysis to the utility provider and consumers, private user information can be inferred from SMs readings. Several methods are developed in the literature that uses perturbation by adding noise to alter user load, hide consumer data, and preserve privacy. Most practices limit the amount of perturbation noise using differential privacy to protect the benefits of data analysis. However, additive noise perturbation may have an undesirable effect on billing. We present a virtual battery model that uses perturbation with additive noise obtained from a virtual chargeable battery. Our model uses fog aggregation with authentication and encryption that employs lightweight cryptographic primitives. We use Diffie-Hellman with a two-way challenge-response method for symmetrical key exchange. A hash-based message authentication code (HMAC) is used for integrity and authenticity, and Advanced Encryption Standard (AES) for encryption. We present our differentially private model with bounding parameters and a dynamic window algorithm to preserve privacy budget loss in infinite time series.

# DEDICATION

*To my family members and mentors who supported me in my studies.*

# ACKNOWLEDGMENTS

My great thanks to my supervisor Dr. Qutaibah Malluhi for his continuous support during my studies. I am grateful for his valuable feedback and advice. I would also want to express my gratitude to all my mentors at Qatar University for their help and guidance.

I would especially like to thank all my family members for their support. My mother for her countless sacrifices and the endless love and care I constantly get from her. I thank my Father, who always pushed me to pursue knowledge and to better myself. My sincere gratitude and thanks to my wife, who has always been supportive and encouraging throughout my studies and provided me with the time I needed to finish this work.

But, first and foremost, I thank God for his blessing and for giving me everything that leads to me finishing my studies and everything that will follow after.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# LIST OF PUBLICATIONS

Fawaz Ahmad Kserawi and Qutaibah M. Malluhi, "Privacy Preservation of Aggregated Data Using Virtual Battery in the Smart Grid", in proceedings of dependsys2020, IEEE Computer Society Press.

# CHAPTER 1: INTRODUCTION

## 1.1.Motivation

With the need for efficient energy, intelligent energy distribution, and renewable energy integration, traditional electric grids are converted into smart grids (SG). Smart grids are implemented for better reliability, performance utilization, and consumer involvement. In some cases, smart grids can be a requirement imposed by governmental policies to achieve environmental and economic objectives such as the EU 20-20-20 goals to increase renewable resources and decrease Co2 emissions [1]. SG systems employ an Advanced Metering Infrastructure (AMI) that provides bi-directional communication between energy or utility providers and consumers to ensure optimization in real-time. Within the AMI, distributed smart meters in homes, businesses or factories, provide on-demand or scheduled data reports to the utility service providers or data aggregators (DA). The provided data can then be used for accurate meter readings, removing randomness in cost estimations, optimizing consumption with dynamic billing and power generation, and transmission planning. Such data provides helpful information for the end consumer to optimize billing prices and utility companies by providing load distribution, load management, generation, consumption monitoring, billing settlements, and price optimization. A critical aspect of SG is gathering or aggregating the smart meters data to predict power usage and future grid planning.

While accurate data readings and updates to service providers or DAs are essential to achieve the previous goals, periodical data updates can cause serious privacy issues. In a fine-grained data consumption aggregation, several attacks can infer useful information from the data by eavesdropping on the communicated packets. Such attacks can reveal personal information and may be exploited by untrusted parties.

Simultaneously, complete hiding of user data may render the collected data useless when considering the SG benefits in billing, advising user consumption, and network load consideration. It is clear that there is a need to protect the data on all SG components, be it the SM data, the aggregators' network, or the utility provider services and servers. Data hiding and consumer privacy protection should be the primary concern for any SG system. Another consideration is the bandwidth and processing costs of implementing a security solution. Processing and network cost are essential as smart meters are IoT (Internet of Things) devices that do not hold high processing capabilities. Bandwidth is another factor to consider as the large amount of data transmitted from all SMs to a single aggregation server can cause network congestion.

Most privacy-preserving methods in the literature involve power load perturbation by adding noise to the consumer load [2][3][4][5][6][7][8][9]. Noise can be generated from a physical chargeable battery by charging or discharging it. When the battery is charging in a physical battery model, the charging amount is considered a negative noise added to the power load. Conversely, when the battery is discharging, the noise is positive with a value equivalent to the discharged amount. Charging and discharging the battery is usually done using a separate hardware component, usually a controller. The chargeable battery model has some limitations, as the battery's size can impact the volume of the added noise and is more costly to implement. Other works consider using low noise with differential privacy while preserving data analytics [3]. However, such models do not consider the loss of cost from the added noise since reconstructing time series with noise may lead to data losses [10].

In this work, we propose secure, lightweight, and cost-effective data aggregation techniques that ensure the consumer's privacy and guarantee safe transmission of smart meter data. We use light cryptographic methods for encryption and authentication, which protect against possible eavesdropping attacks. Additionally, we use perturbation noise from a virtual value to maintain cost and hide fine-grained load data from potential hostile aggregators or utility providers while keeping data analysis. We present a fog aggregation architecture and a dynamic window differential privacy algorithm for a differentially private Gaussian mechanism.

### *1.1.1.Problem Statement*

This work focuses on building a privacy-preserving aggregation framework that protects smart meters privacy while considering billing cost. Since smart meters are IoT devices, a complete understanding of their properties, including security, privacy, and risk, is required for such devices to be used for industries [11]. Lack of a security model in the smart grid can lead to severe threats, some of which are presented in [5]. For example, an attacker or untrusted aggregator may analyze the collected data to conclude the number of residents in a house, residents' availability, and types of appliances used. Such data can be sold, shared, and exploited by third parties such as insurance companies, entertainment companies, and government agencies [12]. A user load profile can reveal various private information such as the time of appliance usage during the day[13] as described in Figure 1.1. Another form of attack on privacy is to reveal power consumption data that may lead to production patterns in industrial factories and businesses to competitors. Other risks are presented when the collected data is used to detect a vulnerability within the grid. Furthermore, sensitive information related to power distribution and energy production patterns might be leaked. To tackle the above issues, several cryptographic methods, data perturbation using differential privacy, and chargeable batteries are proposed in the literature. However, most of these methods do not consider the effect of noise addition on billing and the deterioration of the privacy budget. Another consideration to achieve privacy is to consider the aggregator or the utility provider as a possible adversary since such data can be exploited, sold, or published in a non-private manner. Limited memory constraints, the processing power of end devices, and network bandwidth may limit the effectiveness of privacy-preserving algorithms and cryptographic methods.

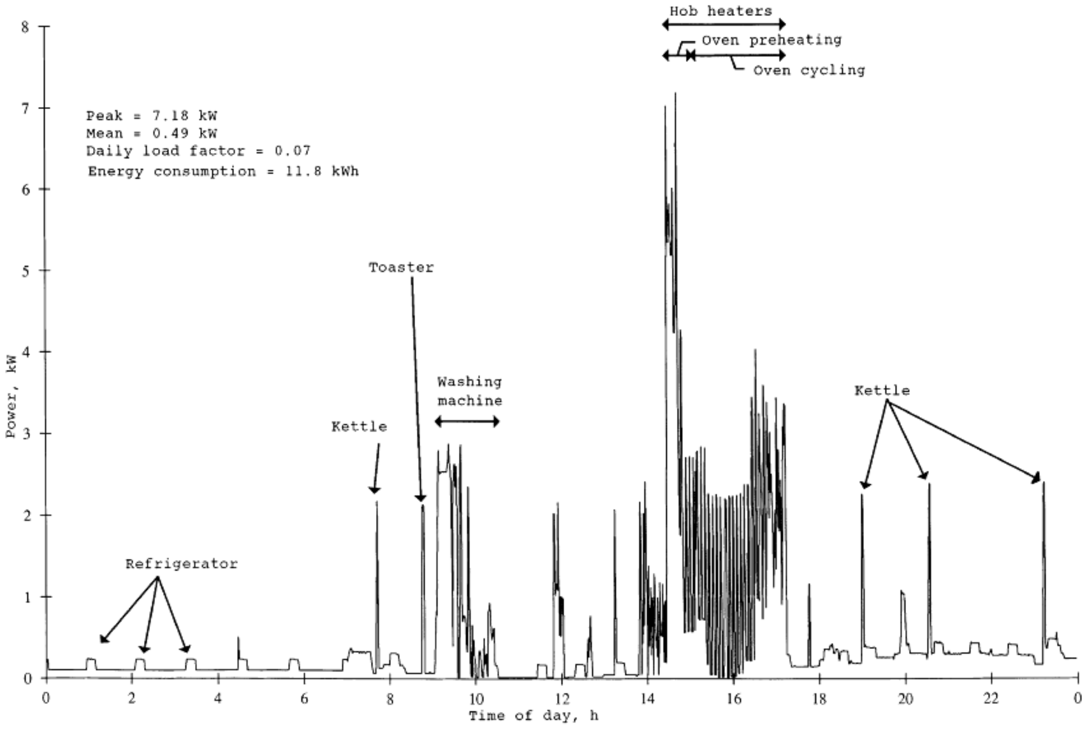

Figure 1.1: Household Electricity Demand Profile [13]

A potential solution to resource limitations is to use Fog aggregation architecture. Fog aggregation can provide better communication and reduce energy consumption as described in [14]. We present a Fog aggregation architecture to overcome the resource limitations and a Virtual Battery model with differential privacy to ensure billing accuracy. The network communication is encrypted to guarantee authentication and integrity of the data. Moreover, we present algorithms and a methodology to preserve the privacy budget, as privacy budget is usually lost over time in infinite time series.

### *1.1.2.Methodology*

We propose a secure, lightweight, and cost-effective data aggregation framework that ensures the consumer's privacy and guarantees the safe transmission of smart meter data. The presented techniques offer a modifiable protection level by specifying the maximum error of perturbed data. We apply differential privacy using an algorithm that modifies the aggregation window to include more data values regarding error constraints for more coarse-grained data and a better privacy budget. A Fog aggregation architecture is presented where aggregators add more privacy guarantees by applying differential privacy over more coarse-grained data obtained from smart meters. Aggregators in our framework also summarize the periodical regional power load of a given region differently. A region contains several smart meters that report relatively fine-grained data to a single aggregator that aggregates the region's overall power consumption; for a faster reporting of regional power load. The virtual battery in our model produces the noise to differentially private algorithms for billing accuracy. Additionally, it limits the consumed power in the perturbed data to a maximum power value to keep the perturbation within a specific error range. We use lightweight cryptographic techniques for encryption and authentication, which protect against possible eavesdropping attacks. Additionally, we use perturbation noise from a virtual value to maintain cost and hide fine-grained load data from potential hostile aggregators or utility providers while sustaining data analysis.

### *1.1.3.Objectives and Contribution*

In this thesis, we have the following major objectives:

1. Create a complete Fog aggregated architecture to distribute processing costs and reduce network congestion. The smart meter owner can consider the Fog aggregators or the utility provider as an adversary.

2. Maintain user's privacy by applying perturbation with possible minor noise to maintain utility and billing accuracy.

3. Use and specify variable parameters that allow for a dynamic level of error.

4. Create algorithms that guarantee our previous objectives to optimize our presented framework and provide the parameters connection.

5. Guarantee authenticity and the integrity of the transmitted packets and encrypt transmitted user data over the network.

Therefore, the contributions can be summarized as follows:

1. We create a Fog aggregated architecture where processing is done on Fog nodes to improve end-user privacy, calculate regional power load and aggregate these values to the utility provider.

2. We maintain user's privacy by applying differential privacy with smaller possible noise to maintain utility and billing accuracy.

3. We use a virtual battery model to maintain the cost of added noise for consistent billing while keeping the noise private.

4. We introduce a dynamic window size, max power loads to guarantee a specified perturbation error rate while maintaining privacy budget loss to a minimum.

5. We use HMAC cryptographic techniques that guarantee authenticity and the integrity of the data over the network and protect the transmitted packets from eavesdropping or man-in-the-middle attacks.

### *1.1.4.Thesis Overview*

The remainder of this thesis is organized as follows: Chapter 2 describes the main concepts, terminologies, and the current state-of-the-art models. Chapter 3 Describes the implementation of our model. In chapter 4, we validates the security of our model. Chapter 5 concludes our work and suggests future work.

# CHAPTER 2: BACKGROUND AND RELATED WORK

## 2.1.Background

### *2.1.1.**Fog Aggregation***

Due to the resource constraints usually found in IoT devices (in this thesis, the IoT devices are smart meters), fog computing architecture is used to enhance the framework performance [3]. Fog computing is used to distribute computing over Fog nodes, also known as edge nodes. Fog nodes process data collected from periphery devices, such as smart meters, and forward them to the cloud servers, which in our model are the smart grid's utility provider servers. When handling a large amount of data, the benefit of using Fog computing is to dedicate processing tasks to Fog nodes that do the processing instead of directly forwarding raw data. The use of Fog computing dramatically reduces bandwidth and processing power on the cloud servers or, in our

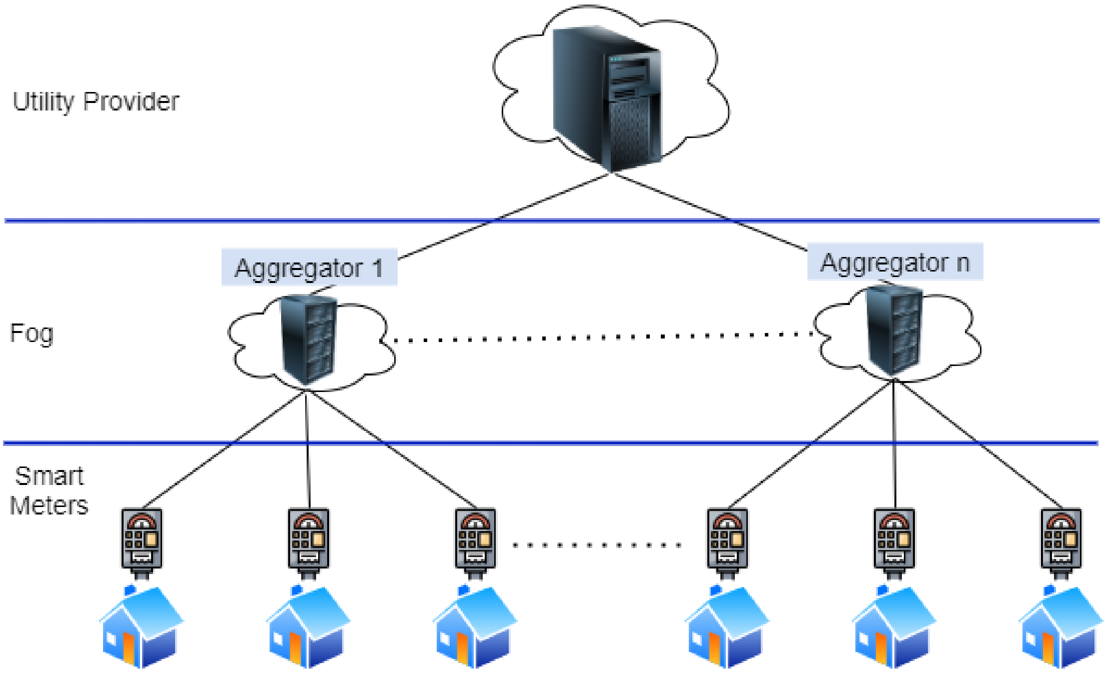

Figure 2.1: Fog Architecture Layers

case, utility provider servers. In a typical Fog-enabled smart grid, the Fog layers is a secondary layer that aggregates several smart meters data to the utility provider. The use of such architecture reduces communication complexity [15]. Figure 2.1 shows a typical Fog architecture. The advantages of Fog computing aggregation with regards to communication and processing are presented in [16] and [3]. In our work, we assume that the aggregator is not to be trusted and cannot infer individual user's data as the received data is perturbed with noise that offers differential privacy guarantees. The Fog aggregators provide statistics for regional consumption in addition to coarse-grained individual consumption statistics over a specified time window. We reduce communication costs since the aggregation is mostly done on Fog nodes rather than sending individual and sensitive raw data to the utility provider. In our article, we refer to Fog nodes as aggregators and end nodes as smart meters; both terms are used interchangeably.

### *2.1.2.**Differential Privacy***

A smart meter privacy model was introduced in [17] where the adversary attempts to obtain accurate smart meter readings in two scenarios. The challenge is for the utility provider to perturb and present the time series data using algorithms that make both scenarios indistinguishable when analyzed by the adversary. An acceptable level of privacy is reached if the adversary cannot correlate which response is given for which scenario with a much better chance than random guessing. However, it was found that the adversary's advantage grew when observing several messages or at periods where the power consumption is large. Thus, the approach is either not accurate enough or not private enough [4]. A guarantee of privacy is achieved with various methods proposed in the literature. It is a common practice to use differential privacy.

Differential privacy is a system first introduced by [18] for sharing public information about a data-set by specifying all the patterns within the data-set without releasing information about individual contributors in the data-set. The leading theory describing differential privacy is that if a single user's data were small enough, a query result would not expose much information about that user. Therefore, the overall result would not change when changing, deleting, or adding the data of any single user, thus providing contributors anonymity. Differential privacy is presently widely trusted in the literature as a powerful concept of privacy. Differential privacy offers a level of confidentiality by perturbing aggregated readings to prove that the value can be differentially private. Differential privacy provides intriguing properties, namely post-processing closure, and composition.

**Definition of Differential Privacy** Assuming a mechanism $M$: $X_n \to Y$. For any two neighbouring data-sets $X, X' \in X_n$ that are different in one entry. We say that $M$ is $\varepsilon$-differentially private if, for all neighboring $X, X'$, and all $T \subseteq Y$, we have: $Pr[M(X) \in T] \leq e^{\varepsilon} Pr[M(X') \in T]$, where $M$ is a randomization mechanism and can be an algorithm that introduces additive noise to the original data $X$. In the literature related to differential privacy, the word mechanism is often used; however, both terms "mechanism" and "algorithm" are used interchangeably.

In the literature, usually, Laplacian or Gaussian noise is introduced. Several algorithms and differential privacy properties are discussed in [19]. The previous definition states that if the effect of making an arbitrary single replacement in the database is small enough, the query result cannot infer a single individual's data. The difference between $X$ and $X'$ is the data related to one person (or entity). Therefore, we can get one data set from the other one by either adding, removing, or changing this person data.

There are algorithms that, by result perturbation, can turn query results into a differentially private one. In a smart grid, we consider a single query to be equivalent to one aggregate. In differential privacy, a parameter $\varepsilon$ shows the privacy strength and is referred to as the privacy budget. The perturbation applied on the data in differential privacy is inversly proportional to $\varepsilon$. This means that a smaller $\varepsilon$ produces better privacy but less accuracy and vise versa. It is a challenge in smart metering to balance this value to tweak the added noise that does not break privacy while preserving utility of the data. A trade-off between privacy and accuracy is presented in [17].

**Approximate Differential Privacy** An algorithm $M : X^n \rightarrow Y$ is $(\varepsilon, \delta)$-differentially private if, for all neighbouring databases $X, X\,' \in X^n$ , and all $T \subseteq Y$:

$$Pr[M(X) \in T] \leq e^{\varepsilon} Pr[M(X\,') \in T] + \delta$$

Here, $\delta$ value is considered to be "cryptographically" small. That is, "$\delta$ should be smaller than the inverse of any polynomial in the size of the data set" [20]. .

**Properties of Differential Privacy** Differential privacy offers several convenient properties that make it reasonable to think about it in a very modular fashion and make it "user friendly."

1. Post-Processing: A valuable property of differential privacy is that once the data

is privatized with differential privacy, the privacy will not be breached if the data is not used again.

Let $M : X^n \rightarrow Y$ be $\varepsilon$-differentially private and $F : Y \rightarrow Z$ be an arbitrary randomized mapping. Then $F\ o\ M = F(M(X))$ is $\varepsilon$-differentially private.

2. Composition: Suppose $M = (M_1, ..., M_k)$ is a sequence of algorithms, where $M_i$ is $(\varepsilon_i, \delta_i)$-differentially private and the algorithms $M_i$'s are potentially chosen sequentially and adaptively. Then M is $(\sum\limits_{i=1}^{k} \varepsilon_i, \sum\limits_{i=1}^{k} \delta_i)$-differentially private.

It is not always clear what the values of $\varepsilon$ should be to maintain privacy [21] since differential privacy is usually added to static data by a trusted curator. In time series and evolving data sets, it is unfeasible to apply differential privacy with a constant $\varepsilon$ as the data is continuously growing. One crucial property to solve this is the composition of differential privacy. In a composition of $T$ independent queries, the privacy parameters $\varepsilon, \delta$ add up.

For example, at each time iteration, $t_i$ with applying differential privacy of the power load at windows $w_j$ and each window contains the sum of power load $X(t)$; for the first period of 10 minutes $t = 0$ to $t = 10$ the $w_0$ value is $w_0 = \sum\limits_{t=0}^{10} X(t)$. Therefore the window $w_j$ between $t = i$ and $t = i\,'$ values can be calculated by:

$$w_j = \sum_{t=i}^{i\,'} X(t_i). \tag{2.1}$$

Say we apply differential privacy with parameter $\varepsilon_1$, $\varepsilon_2$ and $\varepsilon_3$ on time windows $w_1$, $w_2$ and $w_3$ respectively; the composition property states that the overall privacy of all three windows is: $\varepsilon_{total} = \varepsilon 1 + \varepsilon 2 + \varepsilon 3$. The previous equation 2.1 shows that over time, the values of $\varepsilon$ will increase. We found that many solutions in the literature ignore the deterioration of $\varepsilon$ over time, and we will present our solution for this in later sections.

Sensitivity is another parameter to consider in differential privacy. Sensitivity captures the quantity by which a single data entry can affect the mechanism in the worst case and therefore change the perturbation level needed to hide all data. Thus, the sensitivity of a function bounds the perturbation level we must introduce to preserve privacy. For example, in counting queries, it is recognized that the sensitivity is equal to 1. Sensitivity in general metric spaces is defined in [18] as: Let $\mathscr{M}$ be a metric space with a distance function $d_{\mathscr{M}}(.,.)$. The sensitivity $S_{\mathscr{M}}(f)$ of a function $f : D^n \to \mathscr{M}$ is the amount that the function value varies when a single entry of the input is changed.

$$S_{\mathscr{M}}(f) \overset{\text{def}}{=} sup_{x,x\,':d_H(x,x\,'=1)} d_{\mathscr{M}}(f(x), f(x\,')). \tag{2.2}$$

Where $x, x\,'$ in equation 2.2 are two neighboring data sets that differ in only one entry. In the case of real-valued function $f$ in 1-dimensional space, the sensitivity $\triangle f$ can be calculated by: $\triangle f = max|f(x) - f(x\ ')|$. For functions running under multi dimensional data the sensitivity is measured under $\ell_1$ and $\ell_2$ norms. In additive noise differential privacy $\ell_1$-sensitivity is used with Laplace mechanism where $\ell_1$ is calculated from:

$$\triangle(f) = max_{\|x-x\,'\|_1=1} \parallel f(x) - f(x\,') \parallel_1 . \tag{2.3}$$

Another mechanism is the Gaussian mechanism that uses $\ell_2$-sensitivity and is given by:

$$\triangle_2(f) = max_{\|x-x\,'\|_1=1} \parallel f(x) - f(x\,') \parallel_2 . \tag{2.4}$$

The definitions of both Laplace and Gaussian mechanisms with $\ell_1$ and $\ell_2$ sensitivities are given in [19]. In smart metering data, which is a time series, the maximum global value is unknown, making it challenging to measure sensitivity. Substantial values that are unknown to us, since the value can come up in the future, can destroy differential privacy.

## 2.2.Related Work

Various techniques have been developed for private and secure data aggregation in the literature. Such as cryptographic techniques [14][22][23][24][6], noise addition [2][4][5][7][8][9], or hybrid methods [3][5][25][26]. Cryptographic methods are usually used to encrypt user data during network transmission. Cryptographic techniques, such as homomorphic encryption, allows aggregators to perform arithmetical functions on encrypted user data while being oblivious to the actual data. Other cryptographic methods, such as public-key encryption, are used to ensure authentication. Choosing the proper cryptographic method is essential for considering the added processing costs on smart meters and aggregators devices and for maintaining an appropriate privacy level for consumers.

### *2.2.1.Preserving Privacy Using Cryptography*

Baloglu U. B. et al. [10] merged encryption using Decisional Diffie-Hellman with perturbation to produce a lightweight data aggregation scheme in which the aggregated data is lossless. A task scheduler is used for data transmission and aggregation by appending perturbed time series data with encrypted noise. Additionally, algorithms are used to continuously monitor user data transmission to two processing nodes for robustness and integrity. The model is scalable with fast and lightweight encryption. However, there is no level of protection against malicious aggregators or utility providers. Lyu et al. [3] used a multi-layered encryption scheme using an efficient stream cipher with public-key crypto in a Fog aggregation architecture. Homomorphic encryption is used with a one-time pad encryption key. However, the application of homomorphic encryption introduces additional processing requirements, and public key exchange requires a third party for reliable distribution of public keys, which adds more cost to the model. A complete security model is presented by Mahmood et al. [27] by introducing a lightweight elliptic curve cryptography (ECC) scheme that allows for mutual authentication. The ECC scheme protects against insider adversary and man-in-the-middle attacks and provides integrity and encryption with good performance. The scheme is explicitly proposed for a smart grid; however, it requires a trusted anchor that will store keys. Adding a trusted anchor to the system will result in additional costs; moreover, trusting an anchor could mean trusting a possible adversary. Gai et al. [28] presented a model that combines blockchain with edge computing technology to preserve smart grid privacy and provide energy security. A hidden authorization channel and group signatures are used for authentication and user validation. In addition, smart contracts are used to devise a security strategy that runs on the blockchain. The model offered optional anonymity and key exchange with a lightweight authentication protocol by utilizing the blockchain properties.

### *2.2.2.Privacy Using a Chargeable Battery*

Adding noise to user data can achieve user privacy by turning the user load using energy from storage devices or rechargeable batteries. While this will not affect long-term billing, altering the user load with a large amount of noise will eliminate the aggregated data analysis's benefits. Furthermore, such methods require the additional cost of power storage devices and may not be applicable for non-electric utilities, for example, gas or water. Kalogridis et al. [2] used power routing to alter the consumers' load signature for masking usage data using a rechargeable battery for power management and a power-mixing algorithm. A method for hiding consumption data is introduced by combining load signature with energy from a rechargeable battery. A tweakable privacy moderation algorithm is developed, allowing consumers to control their privacy and set an evaluation methodology for privacy evaluation. However, using a rechargeable battery requires extra cost, and the battery size offers a limit on how much of the data can be hidden. Varodayan et al. [6] improved on the best effort algorithm introduced in [2] by using a trellis algorithm and stochastic battery policies with 26% less information leakage than the previous work. This percentage was calculated by a methodology that quantifies information leakage. While this improves the rechargeable battery model, the proposed solution suffers from the power storage cost and user data loss for data analysis. In [9], Zhang Z. et al. provided a cost-effective model using a chargeable battery with limited capacity and differential privacy guarantees. A domain-limited noise distribution parameter is used to lessen the lead range on the small capacity battery's limited power. Smart meter readings are perturbed using a battery controller that charges or discharges the battery. A switching method is presented to prevent SM from reporting its reading when violating battery limits. The noise taken from the battery is used as Laplacian noise with narrowed domain to compensate for the limited size of the battery. A multi-armed bandit algorithm is used to reduce cost. This method showed a privacy level nine times better than traditional privacy models; however, some extra cost of setup and future maintenance is not considered.

### *2.2.3.Preserving Privacy by Additive Noise*

Similar to the noise added from a chargeable battery, adding random noise using noise generators can mask user load profiles protecting consumer data. However, adding noise destroys billing accuracy causing coarse-grained billing data to be sent separately. While long-term billing can be sent independently, accurate fine-grained data is needed for dynamic billing when the consumption is relative to the time of the day. Furthermore, the amount of noise added can destroy the benefit of the data for utility and data analysis. Therefore, the amount of noise added must be small enough not to disturb data analysis or billing accuracy.

#### ***Noise Generation***

In [7], Sankar L. et al. presented a theoretical framework for smart meters that ensures utility requirement while maintaining privacy utilizing tools derived from information theory and the Markov model. The presented framework uses power spectra from a high-power, less private appliance as distortion noise and removes the frequency of low-powered elements. In this model, the appliance state's measurement controls the load measurement modeled after random Gaussian variables, where these variables correspond to actual values. The framework uses an interference-aware reverse water filling technique that filters low-power frequency components by summoning a distortion noise from always-on appliances with less privacy concern. The model offers a tunable tradeoff between privacy level and the usefulness of data analysis. Such a model is agnostic to future data collection techniques. However, as the noise load is used from a used appliance continuously, such appliance's privacy is not considered. While it is possible to obtain the noise from a chargeable battery instead of a constantly powered appliance, it suffers from other issues that the previous chargeable battery models suffer from except for keeping user data analysis possible.

### *Differential Privacy*

Another commonly used technique to achieve privacy is using differential privacy. Differential privacy is introduced in [18] and [19] where aggregated data is perturbed in a way that preserves privacy and maintains the overall data attributes for analysis. Here, a privacy algorithm injects noise so that an attacker cannot extract and discover user data from a data set. A privacy parameter $\varepsilon$ determines the privacy level. The added noise can be obtained using various algorithms; however, mainly Gaussian Mechanism and Laplace Distribution are used to introduce the noise for time series data. Traditionally, utility providers or aggregators that end-users already trust use differential privacy to publish statistical data to the public. A distributed form of differential privacy can be used to achieve privacy from multiple data sources to protect against untrusted aggregators. Dwork et al. [20] created a protocol that uses distributed differential privacy (DDP) by shares of randomly generated noise by online entities.

Savi, M et al. [8] investigated the privacy-utility trade-off considering the aggregated set size by calculating an attack success probability against presented perturbation. Colored noise and white noise is introduced and measured against SMART dataset [29]. An $\varepsilon$-Privacy model is proposed with a Gaussian noise perturbation to measure privacy. The $\varepsilon$-Privacy measurement is then used against white and colored noise and the measured perturbation was introduced to the sample dataset. By measuring the $\varepsilon$-Privacy level, it was proven that by introducing the proper colored noise, a lower noise variance could be used on the user data while preserving privacy. Furthermore, the effect of data group size, the interval of observation, and the sampling period on privacy level were measured. Results revealed that no attack on privacy is possible using such a framework.

Eibl et al. [4] implemented differential privacy over actual smart metering data considering applying differential privacy on utility for the large data sets of smart meters. Point-wise privacy is used with epsilon noise for each aggregation period with $\varepsilon = 1$ considering the overall privacy budget from the composition property of differential privacy. Additionally, the perturbed data is smoothed to increase utility while keeping it differentially private exploiting the post-processing property of differential privacy. For the usability of data statistics, it was found that a large amount of data is required, with thousands of smart meters, to achieve a useful utility after applying differential privacy. Such implementation does not consider the deterioration of privacy budget $\varepsilon$. Improving on the work in [4], an advanced algorithm for protecting peak power values for renewable energy resources is introduced by Hassan et al. [30]. Power load is made using a differentially private real-time load monitoring (DPLM) algorithm with Laplacian noise. Point-wise privacy from [4] is used to apply $\varepsilon$-differential privacy for a small periods. Furthermore, the (DPLM) algorithm limited peak power values by trimming them from the current reading and adding excessive energy for the next iteration period. A promising error rate of 1.5% is achieved for specific peak values.; however, the point-wise differential privacy does not account for the deterioration of the privacy budget due to the composition properties.

**Differential Privacy with Fog Aggregation** Lyu L et al. [3] proposed PPFA, a private aggregation system using distributed differential privacy on a fog-cloud architecture. In this work, fog nodes, or aggregators, aggregate data from smart meters and, in turn, send their aggregated data to the cloud. A Gaussian mechanism is used in distributed differential privacy for fog nodes and cloud aggregation. Two-level encryption is used: OTP for noisy measurements and public key encryption for authentication. The Gaussian noise is produced with regards to ($\varepsilon$, $\delta$)-differential privacy in the SM nodes. The measurement plus noise is sent to the fog nodes using homomorphic encryption to enable fog nodes to aggregate results while being oblivious to the actual user power consumption. After aggregation, fog nodes add their own produced noise for all data aggregated at the cloud level. Public key encryption is used to guarantee authentication. The presented framework kept user privacy while maintaining utility. Additionally, the use of fog-cloud architecture reduced the bandwidth and power cost. However, the authors used a third party for public-keys distribution and aggregators aggregate multiple meters only for data analysis, ignoring tariff data. Another framework called RE-ADP, made for aggregating time series of IoT devices, is presented by Huo et al. [31]. It employs an adaptive $w$-event differential privacy by applying differential privacy on dynamic $w$ time stamps over infinite time series in a fog aggregation architecture. RE-ADP uses a privacy-preserving stream data aggregation with an adaptive time window size ($w$) that depends on a proposed metric called quality of privacy (QoP). Additionally, an adaptive sampling scheme is used to improve aggregated data accuracy by using long short-term memory (LSTM) machine learning model. Subsequently, a smart-grouping perturbation is presented using K-means to group IoT sensors and injecting additional noise. While applying group policy did provide better privacy, it may not apply for smart metering billing data unless cost data is sent separately.

# CHAPTER 3: VIRTUAL BATTERY MODEL

In our model, the SMs at the end nodes are assumed to be tamper-free to prevent end-users from changing the virtual battery's values. It is possible to achieve a tamper-free smart meter by running our code inside a secure enclave similar to Intel SGX as explained in [32]. Changing the virtual battery value may result in manipulating billing rates when the billing rate is reliant on the time of consumption. We use a fog distribution model with a virtual battery value that guarantees accurate billing values while offering perturbation from adding noise collected from the virtual battery. Several perturbation models can work with this architecture, such as physical chargeable battery noise perturbation. Using our virtual battery model comes with the benefits of a large battery power capacity as the battery is virtual. Our model follows a distributed fog aggregation where noise is generated at SMs and fog aggregators. We assume that aggregators and utility providers are not trusted and use distributed differential privacy.

## 3.1.Virtual Battery

A virtual battery is an acknowledged value between the SM and the utility provider server. Figure 3.1 shows the architecture for exchanging this value.

Table 3.1 contains a list of symbols used in this section. We use $VB(t)$ to denote the VB value at time $t$. Therefore, $VB(0)$ is the initial VB value at the beginning of period $m$, and $VB(m)$ is the VB value at the end of period $m$. In the first update of the smart meter in period $m$, the value of $VB(0)$ is zero; the smart meter then transmits this value to the utility at the end of $m$ for the next period. The following steps explain the process of applying noise perturbation with a virtual battery on a single SM without encryption:

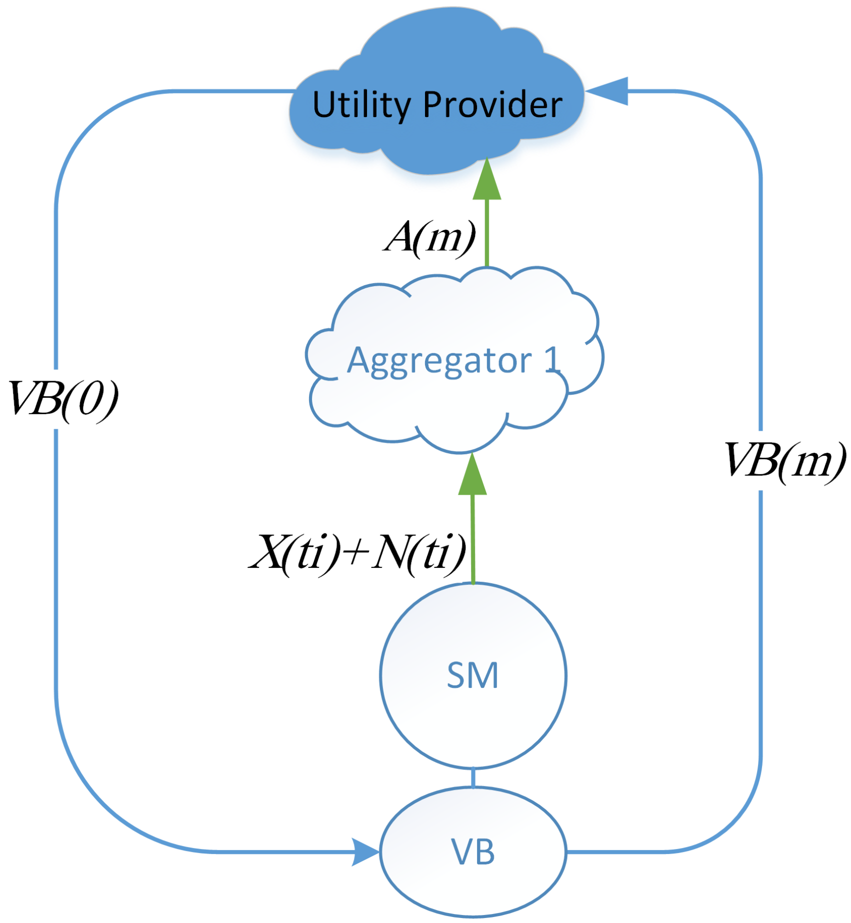

Figure 3.1: Virtual Battery Model Architecture for a Single Smart Meter

1. At the beginning of a period of $m$, typically a month, the value $VB(0)$ is sent to the utility provider for acknowledgment. For the first smart meter update to utility provider of the first period $m_0 : VB(0) = 0$.

2. For a short period $t_1$, the SM applies perturbation by adding a noise value $N$ to its power consumption and sends the perturbed load to the aggregator. Subsequently, the SM subtracts the value of the noise from the virtual battery preserving the cost value of the added noise shown in equation (3.1):

$$VB(t_1) = VB(0) - N(t_1). \tag{3.1}$$

Table 3.1: Table of Symbols for the Virtual Battary Model

**Symbol table**

| **Symbol** | **Meaning** |
|---|---|
| $m$ | long time period for example a month |
| $t_i$ | The time period of smart meter reading at iteration i |
| $X_n(t)$ | The power load reading of smart meter $n$ at time $t$ |
| $L_n(t)$ | The encrypted power load reading plus noise of smart meter $n$ at time $t$ |
| $VB_n(t)$ | The VB value of smart meter $n$ at time $t$ |
| $N_n(t)$ | The noise added at time $t$ to power load of smart meter $n$ |
| $N_a(t)$ | The noise added at time $t$ to power load of an aggregator $a$ |
| $A_n(t)$ | The coarse grained aggregated value of smart meter $n$ load plus noise |
| $TC_n(t)$ | The total consumption for the smart meter $n$ |
| $U(t)$ | The total consumption of all smart meters at time $t$ |

3. At the end of period $m$, the SM sends the value $VB(m)$ to the utility provider server where:

$$VB(m) = \sum_{i=1}^{m} VB(t_i) - \sum_{i=1}^{m} N(t_i). \tag{3.2}$$

4. The aggregator calculates $A(m)$ from equation 3.3 and sends it to the utility provider. The utility provider receives $A(m)$, the aggregated consumption with noise for period $m$ obtained from:

$$A(m) = \sum_{i=1}^{m} (X(t_i) + N(t_i)). \tag{3.3}$$

The utility provider then subtracts the value of VB consumption of period $m$ which is $VB(0) - VB(m)$ to calculate total consumption $TC$:

$$TC(m) = A(m) - (VB(0) - VB(m)). \tag{3.4}$$

5. The security of the VB value transmission is discussed later in section 3.4. The exchange of the finally consumed or added load from the battery is only done after a long period to confirm billing, for example, once per month. The effect of this is that the utility provider cannot infer any useful information from the consumed value of the virtual battery. This effect holds since the operation of subtracting virtually provided power is done only by end nodes (SMs).

For dynamic billing, it is possible to use multiple virtual batteries; for example, when billing is different between daytime and nighttime, we can use $VB_{day}(m)$, $VB_{night}(m)$. Where noise added at daytime is added to $VB_{day}$ and nighttime noise is added to $VB_{night}$. Both values are sent from the smart meter to the utility provider.

In contrast to privacy models that use an actual chargeable battery for the load [2] [6], the use of a virtual battery will cut down the cost of the power storage device and its maintenance. A benefit of using the virtual battery is that our model works with non-electric utility, for example, gas or water, as the battery is virtual. Adding to that, for frameworks that require load from other appliances, such as [7], we can replace the appliance load by load from a virtual battery. Models that use differential privacy [29] [3] can use the Gaussian Mechanism noise from the virtual battery for robust billing. For example, when a user has a sudden substantial power consumption, the volume of noise required to hide it may be too large. This added noise may not be accounted for in billing. We can ensure that the consumer power load does not exceed a certain amount by thresholding the power load at maximum value and adding the subtracted value to the virtual battery. For example, if $P_{peak}$ is exceeded at time $t_{i-1}$ the value of $VB(t_i)$ will be the previous virtual battery value $VB(t_{i-1})$ minus the absolute value of $P_{peak}$ subtracted from $X(t_{i-1})$ as shown in equation 3.5.

$$VB(t_i) = VB(t_{i-1}) - |X(t_{i-1}) - P_{peak}|. \tag{3.5}$$

Additionally, it is possible to use a significant noise and add its consumption to the battery since the battery value will be sent for billing eventually. It is possible to give the SM another level of privacy where the virtual battery's noise is used for a low value of $\varepsilon$ in differential privacy. This guarantees a high level of privacy at the price of limiting data analysis and maintaining billing accuracy. In our model, we use a distributed form of differential privacy.

## 3.2.Aggregation Architecture

Our Fog architecture consists of three layers, as shown in Figures 3.2 and 2.1. The bottom level consists of all smart meters, Fog aggregators in the middle and the utility provider servers at the top. Several delivery algorithms that can be used for the smart meter-aggregator communication are mentioned in [33].

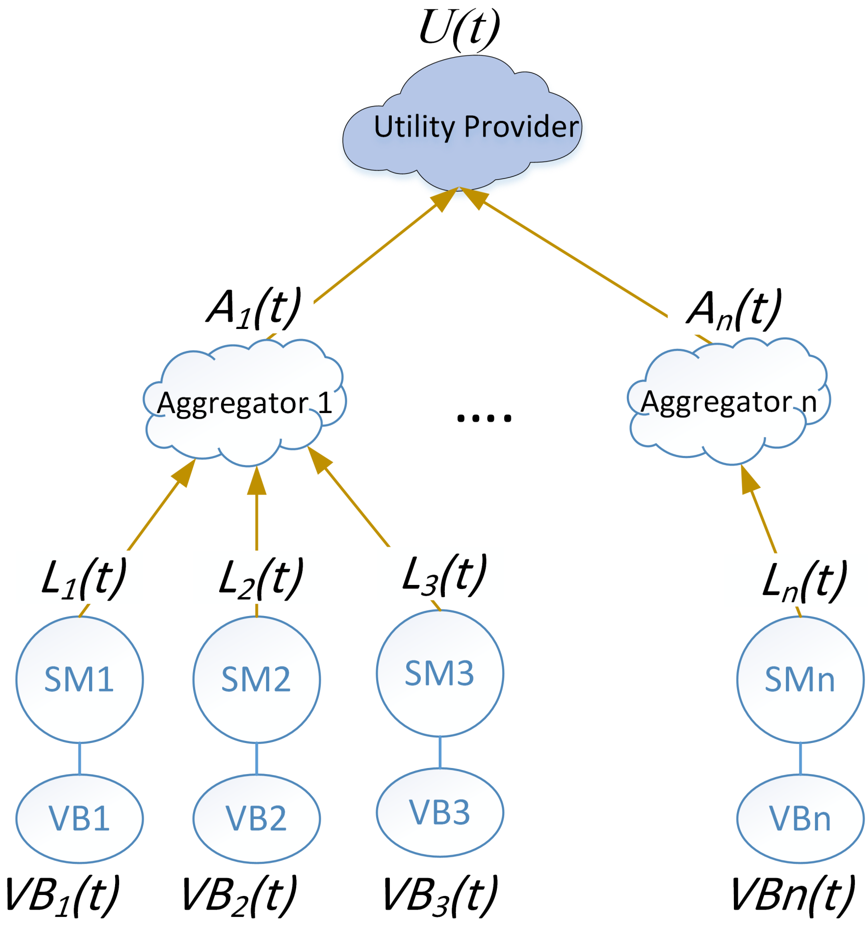

Figure 3.2: Distributed fog aggregation with Virtual Batteries

### *3.2.1.Smart Meters Aggregation*

We consider Fog nodes, or aggregators, to provide better and cheaper service since aggregators are one hop away from smart meters. The transportation cost of all raw data from smart meters to the utility provider is much more than the cost of transmission from the smart meter to aggregator to utility. The cost is reduced, and less data is transferred since measurements are aggregated at the fog nodes, and the data transmitted from the smart meters is coarse-grained. Additionally, regional power loads from several smart meters can be calculated on Fog nodes and then aggregated to the utility provider, saving energy consumption. Excluding setup parameters, our model requires uni-directional communications from smart meters to aggregators and aggregators to the utility provider.

Both the aggregator and the utility provider are untrusted in this architecture and may not discover private information about users. Smart meters are assumed to be tamper-resistant, store their encryption keys and apply differential privacy on readings in a small window of time before sending it to aggregators. The size of this windows can be between previously set parameters $w_{min}, w_{max}$ this is discussed in a later section 3.3. We use fog aggregators for our model, where a collection of SMs connect to several aggregators that collect and aggregate their data to the utility provider. Each SM node perturbs its measurement with noise used from the virtual battery. The noisy measurement is then encrypted and sent to nearby fog aggregators. The aggregator decrypts and aggregates the values, encrypts the results, and sends them to the utility provider; further description of our encryption methodology is described in Section 3.4. Finally, the service provider decrypts the results. Periodically, SMs encrypt and send the virtual battery's value to the utility provider, where the value is compared and added for billing. For example, for a smart meter that updates its power load every minute, we add noise $N(t_i)$ to each reading at $t_i$ where $t_i - t_{i-1} = 1$ minute. At a monthly period $m$ including $t_0$ to $m$ the $VB(m)$ value would include all added noise from $N(t_0)$ to $N(m)$ and the perturbed load $L(m) = \sum_{i=1}^{m}(X(t_i) + N(t_i))$. Only the value of $VB(m)$ is sent from the SM and by the end of $m$ the billing should only account for actual load which would be equal to the noise subtracted from the perturbed load or $ActualLoad(m) = L(m) - VB(m)$. The period for sending the virtual battery's value is typically long enough to prevent the utility provider from subtracting noise from this value. Each smart meter $SM_n$ has a value $VB_n$ with a starting value $VB_n(0)$. Initially, at the start of the smart meter updates for the first period $m_1$, the value $VB_n(0)$ is equal to zero. For the next period $m_2$ , $VB(0)$ is already sent from the smart meter to the utility provider where $VB(0)$ for the period $m_2$ is equal to $VB(m_1)$ at the end of period $m_1$.

Figure 3.2 shows the aggregation architecture and Table 3.1 explains the used symbols.The following steps show the complete aggregation process of a period $m$ where Enc and Dec are encryption and decryption methods discussed in Section 3.4:

1. Virtual batteries $VB_n$ for each smart meter $SM_n$ with values $VB_n(0)$ are encrypted using methods discussed in Section 3.4 and sent to the utility provider by the smart meters. At the first update by smart meters, or when smart meters are first powered on, $VB_n(0)$ is equal to zero.

2. Smart meter $SM_n$ with a stored virtual battery value $VB_n$ communicates with the utility provider and sends the initial virtual battery value $VB_n(0)$. $SM_n$ will perturb its measurement at time period $t_1 : Xn(t_1)$ with noise $N_n(t_1)$. The added noise is taken from $VB_n(0)$ by setting the new value $VB_n(t_1)$ from equation 3.1. $SM_n$ then sends $L_n(t_1)$ the encrypted reading plus noise to the aggregator:

$$L_n(t_1) = Enc[X_n(t_1) + N_n(t_1)]. \tag{3.6}$$

   for a period $m$, say a month, $VB_n(m)$ is calculated by adding all the noise to the virtual battery and the value is then encrypted and sent to the Utility Provider for confirming the $VB$ value of the SM:

$$VB_n(m) = Enc[VB_n(0) - \sum_{i=1}^{m} N(t_i)]. \tag{3.7}$$

3. For an aggregator time window of $w$, an aggregator $A_n$ receives a number of perturbed and encrypted values of consumption $L_{sm_1}$ from equation 3.6 and receives $(\sum\limits_{i \in w} L_{sm_1}(t_i))$ during $w$ from smart meter $SM_1$. Each value of $L_{sm_1}$ in this time period is decrypted and the aggregation is done on the new coarse grained $w$. Aggregator adds its own noise to this load for a parallel differential privacy $N_a(t_{i'})$. With the aggregated value starting at $t_i$ and finishing at the end of $w$ with $t_{i'}$ we calculate and encrypt the consumption of $sm_1$ by:

$$A_n(t_{i'}) = Enc[(\sum_{i}^{i'} Dec(L_{sm1}(t_i))) + N_a(t_{i'})]. \tag{3.8}$$

For example the first time window of 30 minutes has values of $t_i = 0$ to $t_{i'} = 30$ we have $A_n(t_{i'}) = Enc[(\sum\limits_{i=0}^{30} Dec(L_{sm1}(t_i))) + N_a(t_{i'} = 30)]$.

4. The utility provider receives the aggregated values from $n$ number of aggregators. Finally the perturbed power load values are decrypted and summarized by the utility provider for the passed time period:

$$U(t_1) = \sum_{i=1}^{n} Dec[A_n(t_1)]. \tag{3.9}$$

5. For a single smart meter $n$ with the consumption plus noise value of $A_n(m)$ utility provider subtracts the consumption of the virtual battery to calculate total consumption $TC$ of period $m$:

$$TC_n(m) = Dec[A_n(m)] - (Dec[VB_n(0)] - Dec[VB_n(m)]). \tag{3.10}$$

### *3.2.2.Regional Aggregation*

While the previous section discusses private aggregation of end nodes or smart meters, the utility provider needs to have a concept of the overall consumption in a fine-grained manner. Such data is required to optimize power generation by the utility provider. In our model, the algorithm in section 3.3 offers a dynamic window size. If we rely on the aggregator to provide regional power following our algorithm, the max window size may present a long delay. The maximum window size is reached if the smart meter consumption is consistently low, and therefore, the window size is not limited by the maximum error. This section will present a regional aggregation where the aggregator summarizes all smart meters connected to its region and sends them to the utility provider. Regional aggregation is done in a fine-grained manner, limited by the $w_{max}$ of the smart meters, which in our model is relatively a small window size for the aggregator. The summarization does not include smart meters ids and is not correlated with cost or the virtual battery values. Aggregators only send the overall load profile of the region for the connected smart meters; similar work is done in [3].

1. For a window specified for regional aggregation $w_{reg} = Max[w_{SM}]$ , an aggregator $A$ receives a number of perturbed and encrypted values of consumption for $n$ number of smart meters $L_{sm_n}$ from equation 3.6 and receives $(\sum\limits_{i \in w_{reg}} L_{sm_x}(t_i))$ during $w_{reg}$ from each smart meter; where $x$ is the all connected smart meters. Each value of $L_{sm_x}$ in this time period is decrypted and the aggregation is done on all $sm_x$ and the noise added for perturbation is applied on the sum of all smart meters consumption during $w_{reg}$. With the aggregated value starting at $t_i$ and

finishing at the end of $w_{reg}$ with $t_{i\,'}$ we calculate regional consumption of $A$ by:

$$A(t_{i\,'}) = Enc[(\sum_{n=1}^{x}\sum_{i}^{i\,'} Dec(L_{SM_n}(t_i))) + N_a(t_{i\,'})]. \tag{3.11}$$

2. The utility provider receives the aggregated values from $n$ number of aggregators. Each aggregated value can be analysed for power distribution and prediction. The summarized value for all regional aggregation is decrypted and summarized by the utility provider:

$$U(t) = \sum_{i=1}^{n} Dec[A_i(t)]. \tag{3.12}$$

## 3.3.Differential Privacy

### *Gaussian Mechanism*

The decomposition of the Gaussian mechanism uses noise generated from all parties. Each party generates a small amount of noise, and consumer privacy can be guaranteed if the sum has a standard deviation of $\sigma$. From theorem A.1 in [19], we have:

Let $f : \mathbb{N}^{|x|} \to \mathbb{R}^d$ be an arbitrary d-dimensional function, and define its $\ell_2$ sensitivity to be $\triangle_2 f = max_{adjacent_{x,y}}||f(x) - f(y)||_2$. The Gaussian Mechanism with parameter $\delta$ adds noise scaled to $N(0, \sigma^2)$ to each of the $d$ components of the output.

**Theorem A.1.** Let $\varepsilon \in (0, 1)$ be arbitrary. For $c^2 > 2ln(1.25/\delta)$ the Gaussian Mechanism with parameter $\sigma \geq c\triangle_2 f/\varepsilon$ is $(\varepsilon, \delta)$-differentially private.

Following the parameters of distributed differential privacy, we have:

$$\widehat{x}(t) = \sum_{i=1}^{n}(\widehat{x}_i(t)) = \sum_{i=1}^{n}(x_i(t) + r_i(t)). \tag{3.13}$$

Where $n$ is the number of users, $x_i$ represents the measurement of user $i$ at time $t$ and $r$ is the added noise, and $\sigma$ satisfies the theorem A.1.

### *Distributed Differential Privacy*

Distributed differential privacy is a form of differential privacy where an aggregated noise is collected from multiple sources. Differential privacy is already discussed in many models and frameworks in the literature. Our contribution is the fact that added noise can be subtracted from our virtual battery for billing accuracy. Following Theorem A.1 from section 3.3 in our model, the added noise $r$ is reduced from the virtual battery value guaranteeing accurate billing. Additionally, for a higher level of privacy, where the user does not want to allow for his data analysis by the utility provider, we may use an even higher noise level without affecting utility billing on the utility provider side. A separate SM application can provide statistical data analysis to the consumer as the SM has access to noise from $VB(t)$ values without sending such values to utility or aggregators. In our demonstration of the model, we used a single virtual battery with a single method. It is, however, possible to use the virtual battery load for noise generation in multiple methods of perturbation.

**Sensitivity** From theorem A.1. in section 3.3 we have the sensitivity $S$ of our algorithm $M$ is:

$$S(M) = \triangle_2 f = max_{adjacent_{x,y}} ||f(x) - f(y)||_2. \tag{3.14}$$

Several literature methods use a pointwise sensitivity; we argue that this does not guarantee differential privacy as the sensitivity should be on the entire data-set level. Sensitivity is usually unpredictable in the smart grid due to the unpredictability of future power load consumption. A sensible way is needed to determine the sensitivity in a private manner[4]. We use several parameters to ensure bounds on the future sensitivity. We allow smart meters to select a dynamic window size bounded from $w_{min}$ to $w_{max}$ that guarantees differential privacy and peak power limit $P_{peak}$ with consideration to maximum error $err_{max}$. An explanation of the use of these parameters is in section 3.3 and Table 3.2. Following up on the window size bounds $w_{min}, w_{max}$, we can already calculate sensitivity. We know that the lowest consumption windows are zeros perturbed by the most minor differentially private noise $N$ applied on the smallest window $w_{min}$. On the contrary, the maximum consumption is the peak power $P_{peak}$ applied on the maximum window size $w_{max}$. Therefore, the sensitivity in our model can be calculated by:

$$S(M) = \triangle_2 f = P_{peak}/w_{max}. \tag{3.15}$$

Table 3.2: Table of Symbols for The Differential Privacy Model

Differential Privacy Model Symbol Table

| Symbol | Meaning |
|---|---|
| $w_{min}$ | minimum window size |
| $w_{max}$ | maximum window size |
| $P_{peak}$ | peak power allowed in $w_{max}$ with $err < err_{max}$ |
| $err_{max}$ | maximum allowed error |

**Standard Deviation** From Theorem A.1. we can calculate the standard deviation $\sigma$ of the Gaussian mechanism for a single dimension where $\varepsilon \in (0,1)$ and $\delta \in (0,1)$:

$$\sigma^2 = 2ln(1.25/\delta).(\triangle f)^2/\varepsilon^2 = 2ln(1.25/\delta).S(M)^2/\varepsilon^2. \tag{3.16}$$

**Privacy Budget** The composition theorem states that the privacy budget parameters $\varepsilon$ and $\delta$ adds up. Therefore, applying $(\varepsilon, \delta)$-differential privacy on each window $(w_1, w_2, ...w_k)$ in a space $k$ number of windows. We apply the following differentially private parameters $(\varepsilon_1, \varepsilon_2...\varepsilon_k, \delta_1, \delta_2, ..., \delta_k)$. Therefore the overall privacy budget are: $\varepsilon = \sum\limits_{i=1}^{k} \varepsilon_i$ and $\delta = \sum\limits_{i=1}^{k} \delta_i$ meaning for the first window we loose from the privacy budget $\varepsilon_1 = \varepsilon/k$ and $\delta_1 = \delta/k$. This is why it is important to increase the window size when possible.

### *Dynamic Window Differential Privacy*

Due to the deterioration of the privacy budget over time, many bounding algorithms exist in the literature [31][30]. To our knowledge, no algorithm exists that considers our chosen parameters. Here we give further explanation to the used parameters shown in Table 3.2.

1. $w_{min}$: Utility provider selects the minimum window size, which is the number of points in the time series of the fine-grained smart meter or aggregator readings. It should not be fine-grained as that could waste the privacy budget, and it should be more significant for the aggregator. In our model, the perturbation algorithm will increase the number of the included values starting at the $w_{min}$ until the error parameter $err_{max}$ or $w_{max}$ is reached.

2. $w_{max}$: Is the maximum number of points included in the time series. This value is required and set by the utility provider to limit the windows of periodical updates of data statistics. For example, at night with zero power consumption, our algorithm may increase $w$ and not be bounded by the error for an extended amount of time. Here the more significant the value of $w_{max}$, the less privacy budget is lost; therefore, this value should consider privacy vs. utility provider updates requirements.

3. $P_{peak}$: Is the peak power load, which is calculated by applying noise on increasing similar power values up to maximum allowed power in $w_{max}$ until $err_{max}$ value is reached. The $P_{peak}$ value is used to achieve better performance in the algorithm since if adding the total consumption over $w$ exceeds $P_{peak}$ value, we already know that $w$ will break $err_{max}$ without adding noise. Here, it may also be possible for a single reading to exceed $P_{peak}$ we threshold all such readings and add the trimmed values to the virtual battery value.

4. $err_{max}$ Is the maximum error allowed for perturbation by the utility provider. The value of $err$ for each $w$ is calculated by the mean absolute error:
$MAE = \frac{\sum\limits_{i=1}^{n} |X\,'_i - X_i|}{n}$

$w$ event $\varepsilon$-differential privacy was introduced in [34] and applied in [31] for $\varepsilon$-differential privacy to protect event sequence occurring in a window of $w$ time. We expand on $w$-event differential privacy for time series data in the smart grid with minimum privacy budget loss.

**Dynamic Window Differential Privacy Algorithm** $w$-event differential privacy presents a solution for the infinite data stream; here, privacy is applied at sliding windows of size $w$ in the smart grid. However, the fixed window size means sacrificing the privacy budget unnecessarily due to the time series data dynamically changing nature in the smart grid. For example, the power load may change dramatically when using heavy appliances such as heaters; on the other hand, night power consumption is usually low. Therefore, we use a dynamically changing window size with specific bounds that with differential privacy guarantees.

In $w$-event differential privacy, the perturbation of data in a single $w$ leads to a consumption of a fixed privacy budget taken from the overall privacy budget. Increasing the window size will lead to a lower privacy budget loss; though, this may lead to a high perturbation and more significant error value. We present a dynamic window differential privacy algorithm that uses a dynamic window size $w$ but limits the window size by $P_{peak}$, $err_{max}$ and $w_{max}$. Our algorithm ensures that the error resulting from the perturbation $err$ does not exceed a certain amount as it is bounded by $P_{peak}$, $err_{max}$. At the same time, we bound $w$ by $w_{max}$ for constant reporting to the aggregator and utility provider. The algorithm 1 explains the process of our model.

**Algorithm 1:** Dynamic Window Differential Privacy Algorithm

```
Input: $w_{min}, w_{max}, err_{max}$
current window $w = w_{min}$
Calculate $P_{peak}$
while $err \leq err_{max}$ do
    while $w \leq w_{max}$ do
        while $P(t) \leq P_{peak}$ do
            Calculate $P(t_i) = \sum_{i=0}^{i} X(t_i)$;
        end
        w++;
    end
    Calculate error: $err = P(t_i)/L(t_i)$;
    Apply perturbation on $w$: $L(t_i) = P(t_i) + \sum_{i=0}^{i} N(t_i)$;
end
Calculate new privacy budget parameters: $\varepsilon = \varepsilon - \varepsilon_i$, $\delta = \delta - \delta_i$;
Output perturbed value for $w_{i-1} : L(t_{i-1})$;
```

Both smart meters and aggregators use the same algorithm; however, the parameters' values will be different for the aggregator. $w_{min}$ and $w_{max}$ values will be larger for the aggregator as we expect the aggregators to collect more coarse-grained data. Smart meters perturb their data only to protect it from the aggregator as we consider aggregators to be a possible adversary. It is reasonable to assign larger values of $\varepsilon$ the privacy budget for the smart meters since it is less likely for the aggregator to be an adversary. Additionally, the data will be perturbed again by the aggregator before forwarding it to the utility provider. Another benefit of increasing window size is to lower network communication costs since we summarize the windows' power load to a single value. Figure 3.3 shows the flow of our algorithm.

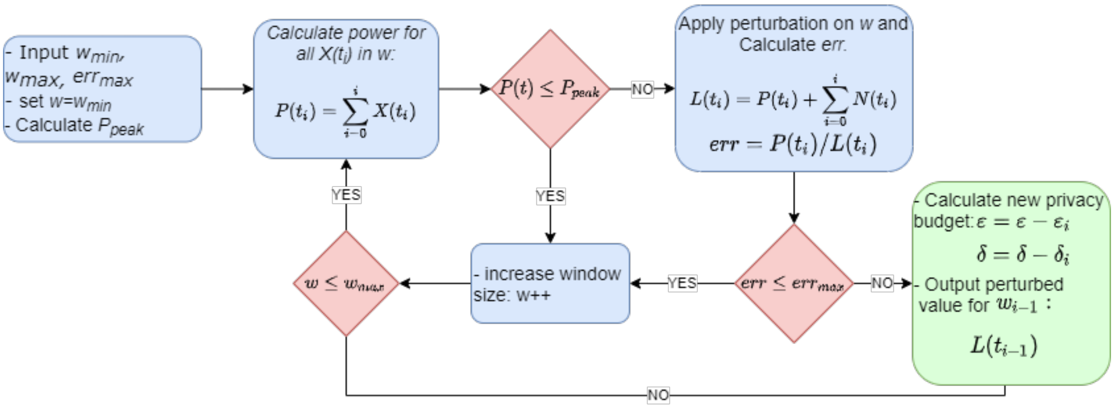
- Input $w_{min}$, $w_{max}$, $err_{max}$
- set w=$w_{min}$
- Calculate $P_{peak}$
Calculate power for all X($t_i$) in w:
$P(t_i) = \sum_{i=0}^{i} X(t_i)$
$P(t) \leq P_{peak}$
NO
Apply perturbation on w and Calculate err.
$L(t_i) = P(t_i) + \sum_{i=0}^{i} N(t_i)$
$err = P(t_i)/L(t_i)$
YES
YES
- increase window size: w++
YES
$w \leq w_{max}$
$err \leq err_{max}$
NO
- Calculate new privacy budget: $\varepsilon = \varepsilon - \varepsilon_i$
$\delta = \delta - \delta_i$
- Output perturbed value for $w_{i-1}$ :
$L(t_{i-1})$
NO

Figure 3.3: Dynamic Window Differential privacy Algorithm

## 3.4.Cryptographic Methods

### *Encryption*

**Diffie Hellman Key Exchange** As SMs are IoT devices with typically low processing power, we try to use lightweight encryption primitives. For each smart meter, a pair of symmetric keys are required. One is to communicate to the utility provider to exchange the virtual battery value and communicate to the aggregator to send load consumption. Aggregators must also exchange symmetric keys with all connected SMs in addition to utility keys. At the same time, the utility provider must hold the symmetric keys for all aggregators and SMs. A Diffie–Hellman key exchange [35] is used for any key exchange between any two parties. Any symmetric key encryption method may work with the distributed keys. We use AES in our model [36]. Usually, public-key encryption is used for authentication. However, it usually requires a trusted third party authority for key distribution and is more processing intensive for devices such as SMs than symmetric encryption. For our authentication, we use a simple two-way challenge-response authentication as shown in Figure 3.4.

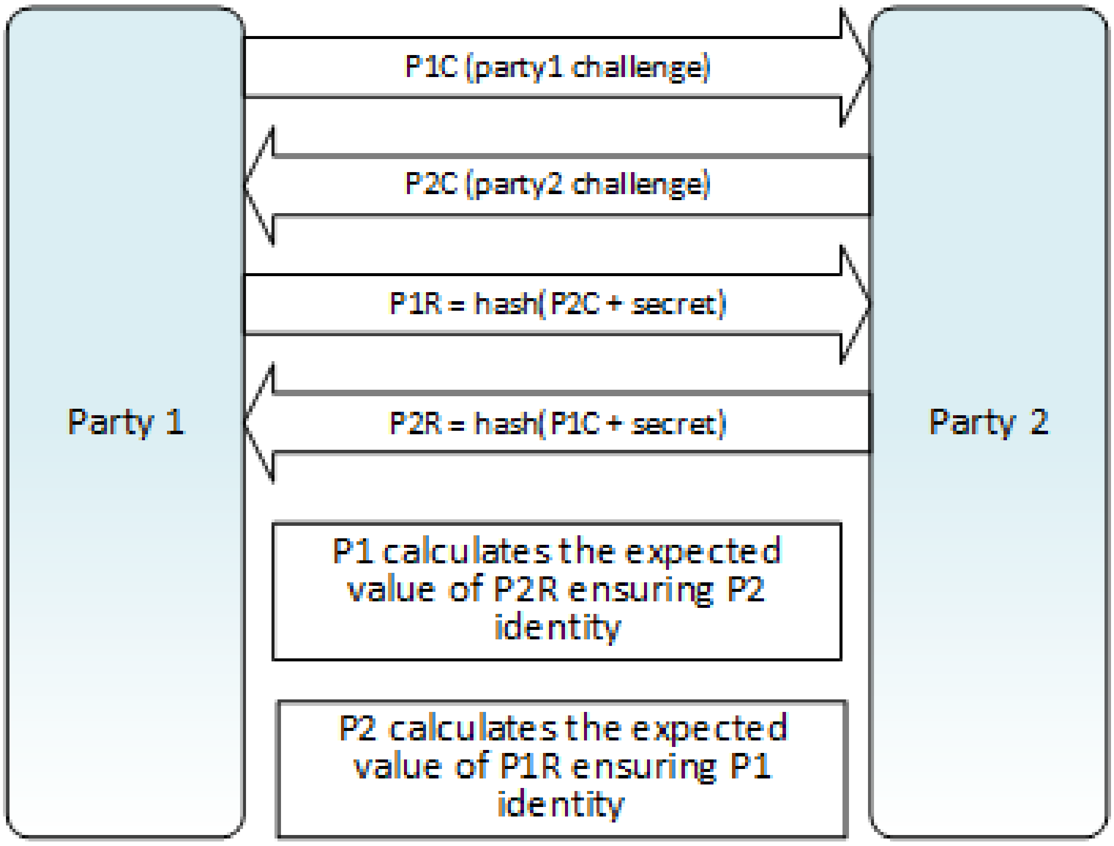
Party 1
Party 2
P1C (party1 challenge)
P2C (party2 challenge)
P1R = hash( P2C + secret)
P2R = hash( P1C + secret)
P1 calculates the expected value of P2R ensuring P2 identity
P2 calculates the expected value of P1R ensuring P1 identity

Figure 3.4: Two way challenge-response authentication

The secret is a concatenation of the exchanged Diffie Hellman key with another secret string. The secret string is a fixed id set by the vendor or utility provider before the smart meter distribution. In the case of Fog aggregator keys exchange, we assume that the utility provider already shares an aggregator's secret string. The two-way challenge-response is used only once for establishing the authenticity of the symmetric key and is not used for data authentication. The purpose of using two-way challenge-response is to protect from man-in-the-middle (MITM) attacks. In MITM attacks, an adversary can mimic each of the parties' behavior to possess separate keys that may be used with each party, allowing an adversary to break the data integrity and authenticity. Keys are stored at each architecture component, on smart meters, aggregators and the utility provider. After the key's initial establishment, there is no need for further communication regarding a key exchange. Such a method is lighter and more cost-effective than other methods such as public key exchange that may require a trusted third party.

**Keyed-Hash Message Authentication Code (HMAC)** HMAC is a variation of the Message Authentication Code (MAC) that uses secret keys and hash functions and provides data integrity and authenticity. HMAC is an improved version of MAC since MAC is known to suffer from length-extension attack where an attacker can append data to the message without knowing the key. The implementation methodology and definitions are presented in [37] and [38]. A shared secret is used in HMAC implementation that does not require a third party's involvement in key distribution, as is the case with public-key cryptography. Following up on the previous section, we assume that the key is already exchanged between parties:

1. The key is used to acquire two separate keys referred to as the inner and outer keys.
2. Two hash rounds are applied; the first round produces a hash from the inner key with the already encrypted message.
3. The resulting hash is hashed again, with the outer key producing the final HMAC code.

Usually, HMAC applies iterative hashing functions such as SHA-256 or SHA-512 over multiple fixed-sized blocks; for example, SHA-256 works on blocks of 512-bit.

Since we can not guarantee our communication block size, we truncate our data blocks to the proper size.

The encrypted message is then sent alongside the HMAC code to the other party. The other party will then hash the message again, and the computed hashes should match the received hash authenticating the message. The definition of HMAC from [37] and [39]

$$HMAC(K, m) = H((K' \oplus opad) \parallel H((K' \oplus ipad) \parallel m))$$

When $K$ is larger than block size then $K' = H(K)$ otherwise $K' = K$. $K'$ is a block-sized key obtained from the secret key $K$ either by padding with zeroes up to the block size or by hashing down to $\leq$ block size and later padding with zeros. Table 3.3 contains an explanation of the used symbols.

Table 3.3: Table of Symbols for the HMAC model

HMAC Symbol Table

| Symbol | Meaning |
|---|---|
| $H$ | a cryptographic hash function |
| $K$ | the secret key |
| $K\ '$ | block-sized key |
| $\parallel$ | concatenation |
| $\oplus$ | XOR |
| $opad$ | the block-sized outer padding |
| $ipad$ | the block-sized inner padding |

# CHAPTER 4: VALIDATION

## 4.1.Security Analysis

### *4.1.1.Confidentiality*

Authentication: authentication is the process of verifying a party's identity and associating it with incoming messages; in our model, a party is a smart meter, aggregator, or utility provider. Since we transmit our data over HMAC we guarantee data integrity and authenticity as HMAC guarantees both properties. Therefore, unauthorized hostile nodes can not inject or change the sent data in any way as any changes in sent packets will be detected. We rely on HMAC authentication, where the integrity of the data relies on distributed keys; therefore, any malicious changes or man-in-the-middle attacks can be detected.

Encryption: We choose to use AES for our encryption as it is a lightweight symmetric cryptography method. AES was first announced in [36]. AES is a well known standard encryption method that is immune to brute force attacks.

### *4.1.2.Integrity*

Integrity represents data accuracy and completeness guarantees by preventing data from being altered by an adversary. HMAC preserves integrity in our model as the encrypted message is sent with the HMAC hash. The information recipient can verify the message's integrity by calculating its HMAC hash and comparing it with the received hash. As the HMAC hash calculation is done with the secret key, the adversary can't break our model's integrity.

### *4.1.3.Availability*

So far, we have assumed that all smart meter and aggregator nodes are always online. However, some nodes may go offline due to network or hardware failure affecting regional differential privacy calculation. For example, a Fog node may not receive readings from a specific smart meter, or a utility provider may not receive an update from an aggregator. Updates must ideally happen before reaching the maximum window $w_{max}$. However, we will not always have ideal updates due to the previously mentioned failures.

Smart meters that do not receive an acknowledgment from the Fog node would log the timestamp and set the flag for the reading to update later ($ul$). Once the Fog node is available, the smart meter will send all stored values with the $ul$ flag to the aggregator. In the same scenario, Fog aggregators would log the data and the time stamp, set the absent smart meters readings as the minimum perturbation level considering the maximum error value, and set the flag for missing smart meters as $ul$ to update this value later. Once the smart meter is back online, the aggregator will receive all the $ul$ flagged values, reapply the noise on the data at the flagged time stamp, and sends them to the utility provider. The utility provider will then reconstruct the data and update it, replacing the flags from $ul$ to valid $v$. We argue that using this method will still guarantee differential privacy as the absent value is still perturbed, and the utility provider is aware of not valid values while still able to apply data statistics. Furthermore, the cost will be updated once the smart meter is back online. This scenario also applies if both aggregator and utility provider are offline.

A similar scenario will occur once a Fog node goes offline. Smart meters without a response from aggregators after the maximum window is reached will update their timestamp, flags, and store the data until the aggregator returns online. After the aggregator goes online, smart meters will send all their data and update their flags to valid $v$; the aggregator updates all the missing values, apply perturbation, and sends them to the utility provider.

Another scenario may occur if the utility provider is offline; aggregators will set their flags, timestamp, and store the data until the utility provider returns online. Once the utility provider is online, all flags for the aggregator's sent data will be set to valid $v$.

An important point to consider is: what if the smart meter's or the aggregator's $ul$ flagged data going over multiple periods exceeding storage capacity. In this case, we will exceed the maximum window and store only the sum of all previous windows as one with the timestamp instead of the fine-grained data, and the flag is set to none valid ($nv$) for statistical analysis. Here, we did not breach privacy; however, the statistics may not be valid for the maximum window size. Such a scenario is only in extreme emergencies where an aggregator or utility provider is offline for an extended period. However, the utility provider or aggregator can still summarize all windows of valid smart meters or aggregators readings to the single large window of none valid reading, thus giving coarse-grained statistics that are still valid.

In a scenario where the smart meter is not disconnected from the network but is completely turned off due to framework issues or power issues, it is impossible to register the smart meter readings. Unfortunately, as we guaranteed privacy by protecting individual data, predicting the exact power consumption during such an event accurately is impossible. However, a rough estimation of user consumption over a long period can be reached, for example, a month. Thus, coarse-grained data analysis for extensive periods can still be accurate to a certain extent. But, it is not possible to reach an exact accurate billing for when a smart meter is entirely down.

Our model's previous scenarios allow all failures to be addressed while still maintaining a one-way data stream from the smart meter to the aggregator to the utility provider.

### *4.1.4.Adversary Models*

***Eavesdropper/MITM***

An eavesdropper is an attacker that intercepts information by sniffing packets on the network. Typically, an eavesdropper is considered a passive adversary that does not interrupt service or inject any new information. On the other hand, a man-in-the-middle (MITM) is an adversary that may modify communications between connected parties. MITM attackers try to impersonate one or both of communicating party members by making independent connections between them, controlling the conversation. Both eavesdropper and MITM must be capable of listening to relevant messages being transmitted between two parties to inject new messages. Therefore, in our model, we consider the eavesdropper's capabilities to be covered by the MITM capabilities.

**Assumptions** We assume that this model's adversary has access to the network between smart meter-aggregator or aggregator-utility provider and possesses strong processing power. Additionally, the adversary can inject packets over the network.

**Goals** A goal of the adversary is to collect smart meter data. Learning such information has an economic impact in addition to breaching privacy. An example of this is learning factory production patterns or inferring appliance usage patterns; competitor or malicious data collection companies can exploit such data. Another goal is to inject false information to poison the data about consumption. Injecting incorrect information can significantly impact the grid since transmitting false information may impact the grid power distribution. For example, sending false low power consumption may cause the utility provider to produce less power for a region, causing power blackouts.

**Capabilities** Assuming the adversary can access the data stream over the network, the network's transmission contains the encrypted messages and the HMAC code. For the adversary to access the encrypted message, he must have access to the Diffie-Hellman key. While a MITM attack is possible on the standard Diffie-Hellman key exchange model, we included the secret string concatenated with a two-way challenge-response method for authenticating the key owner. Therefore, the adversary can't decrypt the messages and is unable to infer user data. Another attack is where the adversary tries to inject messages over the network trying to impersonate an aggregator, smart meter, or utility provider. Such an attack is impossible since HMAC provides data authenticity and integrity; an attacker cannot break HMAC's authenticity and integrity without owning the keys.

***Aggregator and Utility Provider***

We combine aggregators and utility providers as one adversary since their capabilities are similar, and both might be one entity. An aggregator or utility provider can be an adversary trying to infer a smart meter's fine-grained power consumption. This data might be shared with third parties such as insurance companies, entertainment companies, and government agencies [12] for profit. Usually, security policies posed by government institutions prevent this unauthorized sharing of data. However, assuming a disgruntled employee or social engineering cyberattacks, it may be possible to leak the smart meter's aggregated data.

**Assumptions** An adversary in this model has access and privileges to the smart meter aggregated data. In this case, the aggregator poses a more serious threat as aggregators collect more fine-grained smart meter data.

**Goals** The main goal of an adversary is to leak end-users or smart meters accurate load profiles.

**Capabilities** Assuming an aggregator tries to infer user temporal power load accurately, the promise of differential privacy [19] prevents the adversary from inferring individual data readings as the received data is already perturbed with differential privacy. Smart meters summarize the power load for the minimum window $w_{min}$; therefore, the aggregated data is coarse-grained bound by $w_{min}$ and differentially private. Another valuable property of differential privacy is that it is unaffected by post-processing, preventing the adversary from inferring accurate information from the perturbed data. Here we assume that a security policy appropriately sets the Dynamic Window Algorithm parameters; auditing and policy procedures are required to ensure that the parameters are not set in a way that may break differential privacy.

## 4.2.Performance Analysis

This section evaluates our model's performance accuracy and compares it with the traditional Gaussian Mechanism. We apply our model to an actual smart home data set. Our dynamic window model is applied to the individual household electric power consumption data set [40]. The data set contains electric power consumption measurements in a household with a one-minute sampling rate for almost four years. We apply our perturbation with the dynamic window differential privacy algorithm on smart meter then aggregator nodes. For every window $w$, we aggregate the values from the smart meter and apply noise, losing an amount of privacy budget $\varepsilon_w$. Next, the aggregator receives the perturbed data and applies another perturbation level; the noise added to achieve data perturbation is added to the virtual battery of each smart meter. Error is calculated over the time series real-time values for each time window $w$. The values for $w_{min}$, $w_{max}$, $err_{max}$ and $P_{peak}$ are set for the smart meter and aggregator. Naturally, the values of parameters $w_{min}$, $w_{max}$ are selected to be larger for the smart meter than those used for the aggregator. Choosing a larger window size for smart meters hides the smart meter's data from the aggregator and consumes less privacy budget for the smart meter windows as the smart meter processes more fine-grained data. On the other hand, $err_{max}$, $P_{peak}$ values are chosen to be larger for the aggregator as aggregators aggregate larger values for smaller window sizes. The accuracy is visualized and presented by the mean relative error (MRE) for a time period

$T$ in percentage is defined below:

$$MRE = \frac{100}{T} \sum_{i=1}^{T} \left( \frac{\sum_{i=1}^{T} L(t_i) - \sum_{i=1}^{T} X(t_i)}{\sum_{i=1}^{T} X(t_i)} \right). \tag{4.1}$$

Figure 4.1 shows the aggregation of time series for a single smart meter. The original data is shown in blue, the dotted green line shows traditional Gaussian Mechanism, and the dotted orange line represents our model's result. The perturbation is applied by consuming values of $\varepsilon$ per window size $w$ with fixed $w$ used in traditional Gaussian Mechanism and dynamic $w$ size in our model. We can see from the results that we were able to control the error value and achieve a specific error value while consuming less privacy budget since we are using a dynamic window size. The accuracy in our model is evident by the larger range of the traditional Gaussian Mechanism signal or the number of outliers. Note that the values are trimmed at the highest values by $P_{peak}$ and at lowest value by minimum noise in our model. These values are trimmed and sent to the virtual battery and do not affect differential privacy due to the differential privacy post-processing property. Similarly, Figure 4.2 shows the aggregation of time series for the aggregator. In both results, the values of differential privacy parameters per $w$ are set as $\varepsilon = 1$ and $\delta = 1/n^2$ where $n$ is the number of readings in the data set.

Table 4.1 shows a comparison between the MRE difference between using our model and the traditional Gaussian Mechanism. Note that the error rate in our model is much lower than the traditional Gaussian Mechanism while we kept the differential privacy guarantees. The error is low since we choose the perturbation window size optimally with the $err_max$ and trim the power between $P_{peak}$ and noise applied on zero values. The window size also improved upon the loss of privacy budget as the added noise is

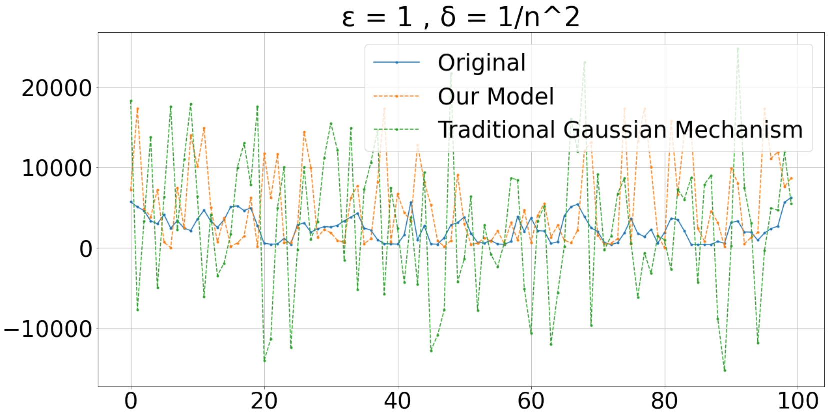

Figure 4.1: Smart Meter Aggregation Our Model VS Traditional

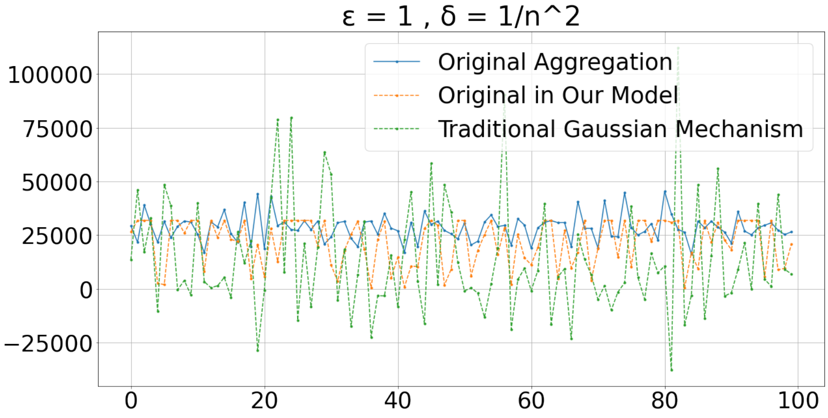

Figure 4.2: Aggregator Aggregation in Our Model VS Traditional

applied on a larger window size when possible. We argue that this is better for data analysis and that the only drawback is using more coarse-grained data dynamically. It is important to note here that we could specify $err_{max}$ to achieve better accuracy; however, depending on the application and the required level of accepted error in the data, it is possible to increase the amount of $err_{max}$ as desired. An initial value of $err_{max}$ can be set on the smart meter running as a closed enclave with the ability only to increase this value given to the utility provider.

Figure 4.3 shows smart meter perturbation error MRE over various values of differential privacy budget $\varepsilon$. Our test outcomes support differential privacy's theoretical

Table 4.1: MRE Comparison Between Our Model and Traditional Method

| Layer | Our Model | Gaussian Mechanism |
|---|---|---|
| Smart Meter | 0.0899 | 0.1861 |
| Aggregator | 0.0312 | 0.0805 |

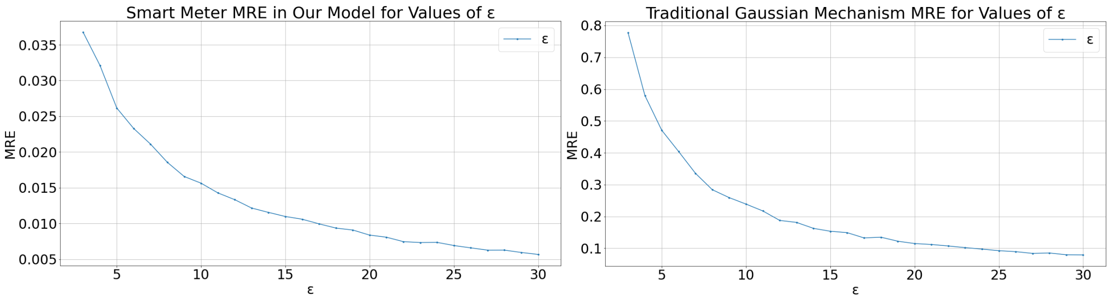

Figure 4.3: Variable $\varepsilon$ values Effect on MRE in Our Model VS Traditional

principles; Lower values of privacy budget $\varepsilon$ results in more noise and larger error values. Furthermore, we can see that the error value does not exceed 3.1% for $\varepsilon = 3$ while MRE is 79% in Gaussian Mechanism with the same $\varepsilon$ value.

Figure 4.4 represents the effect of increasing the window size $w$ on the MRE applied over a monthly period. We can see that increasing the window size reduces the value of MRE. However, there are some inconsistencies in the graph; for example, at $w = 210$, this is due to the randomness introduced when applying noise from a Gaussian distribution. We can see that after a certain amount of window size:$w = 190$ increasing the window size does not decrease the MRE value by much; This is because the MRE is limited by the value of $err_{max}$. $err_{max}$ value is used on the window size $w$ taken from the overall consumption; therefore, the overall MRE will reduce to a certain level. However, even after reaching this value of $w$, the increase of $w$ still benefits our model

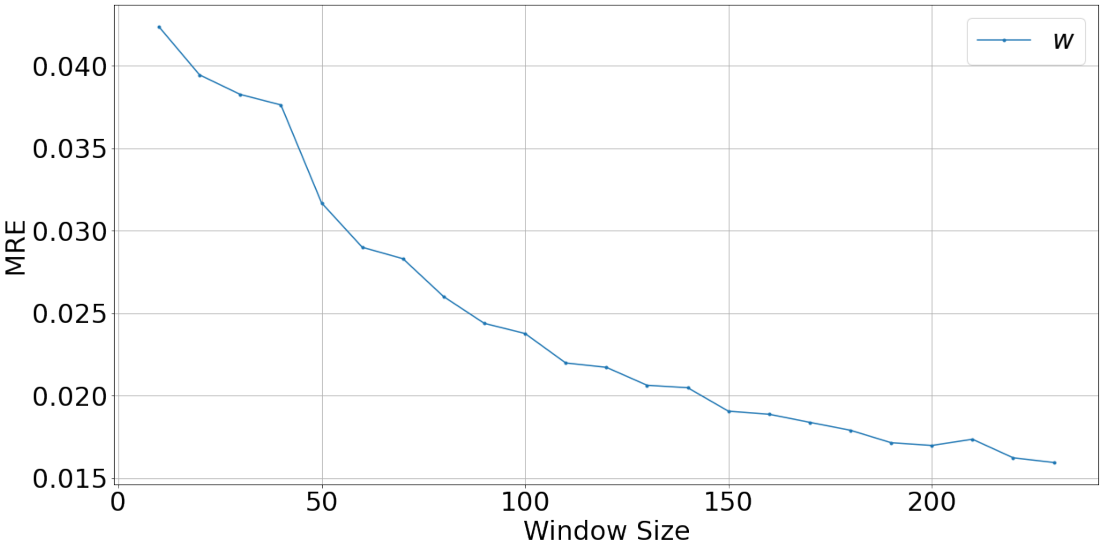

Figure 4.4: Variable Windows Size Effect on MRE in Our Model

in reducing the consumed privacy budget $\varepsilon$.

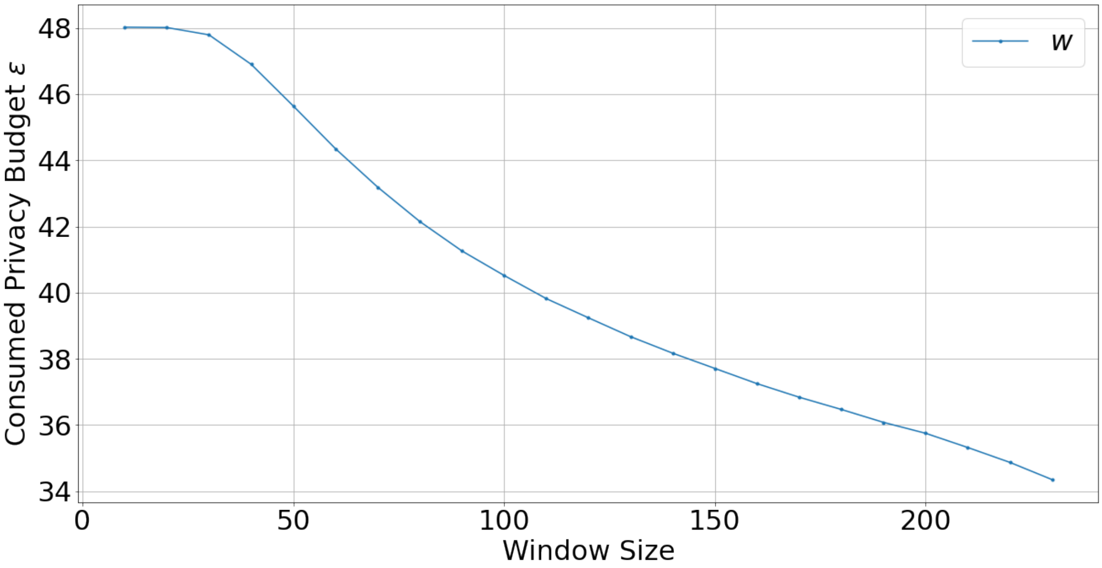

Figure 4.5: Variable Windows Size Effect on Consumed Privacy Budget $\varepsilon$

In the used data set, we have readings of the power load for every minute a fixed window size will be, for example, 10 minutes. The applied $\varepsilon$ on a fixed window for the monthly period will be the addition of all $\varepsilon$ applied on each window by using the total composition property of differential privacy. Therefore, it is trivial that using a larger window size will consume less privacy budget overall. As we discussed previously, larger $\varepsilon$ yields less privacy; therefore, it is beneficial to reduce the privacy budget loss. Our model used differential privacy with additive noise from the Gaussian Distribution on a dynamic window size $w$. In our model, the window size $w$ is set to $w_{min}$ and increased until we reach $P_{peak}$, $err_{max}$ or $w_{max}$ whichever comes first. Figure 4.5 represents the loss in the privacy budget $\varepsilon$ when applied to the overall monthly period. In this figure, the value of $w = w_{max} - w_{min}$. It is important to note that the actual value of $w$ will vary between $w_{min}$ and $w_{max}$ as mentioned before. Nonetheless, increasing $w_{max}$, and subsequently $w$, will consume less privacy budget $\varepsilon$ for the overall data.

# CHAPTER 5: CONCLUSION AND FUTURE WORK

This thesis introduced a virtual battery model for achieving user data privacy in the smart grid. Our model allows for a form of differential privacy while preserving data statistics. It is possible to combine our model with several perturbation methodologies that are already found in the literature or developed in future works. Subtracting noise added to user load for perturbation from the virtual battery ensures billing accuracy regardless of obfuscation level. It is possible to use the proposed model with non-electrical systems, such as gas or water utility distribution, as our battery storage is virtual. We avoided using heavy cryptographic primitives that require a trusted third party or heavy cryptographic processing. Instead, we used a simple Diffie-Hellman key exchange and AES for lightweight symmetric encryption of transferred data. All forms of communication are encrypted with the exchanged symmetric keys using AES. For authentication, we use HMAC, which is another lightweight cryptographic method. Fog aggregation is used where middle fog nodes aggregate SM data for a more coarse-grain aggregation. Our perturbation is done using differential privacy with the Gaussian mechanism over infinite time series data. We present model setup parameters that can control the level of privacy and maintain a certain level of error using a dynamic window algorithm. The dynamic window algorithm uses a window size of the smart meter power readings to preserve the privacy budget. Comparing our model with traditional differentially private models showed several enhancements, namely controlling the error level with values much less than the normal Gaussian Mechanism, the consumption of less privacy budget, and the accuracy of billing. We found that introducing specific parameters can give us the ability to control the error level. Additionally, we found that increasing the size of the window of the selected data for perturbation reduced the

MRE and reduced the amount of consumed privacy budget. However, choosing a fixed window size might affect the error rate; therefore, we introduced a dynamic window size limited by a maximum value to guarantee updates to the utility.

## 5.1.Future Work

While we presented a Framework that guarantees privacy, it is essential to consider a large amount of data collected on an actual smart metering date. Moreover, some variables may change with the addition of renewable resources. Future work will be to estimate the privacy budget loss over a long period. A possible direction to solving privacy loss is to use temporally discounted differential privacy for evolving data sets presented in [41]. Additionally, an evaluation against actual smart metering data and simulating attacks on such data needs further investigation. On the other hand, several data analysis methods should be tested on the perturbed data to estimate the loss of analytic benefits after applying differential privacy. Another direction would be to consider the effect of gathering data from multiple users on privacy, for example, collecting data from an entire neighborhood. Moreover, a possible enhancement is to study the impact of adding data from resources other than power consumption. Such information can be exploited for breaching privacy, for example, weather data might affect consumption as users tend to turn on the heating or cooling appliances. Finally, the constantly evolving science of machine learning might introduce new challenges to keeping an individual's privacy while maintaining viable data for analysis.

# REFERENCES

[1] Z. Fan, G. Kalogridis, C. Efthymiou, M. Sooriyabandara, M. Serizawa, and J. McGeehan, “The new frontier of communications research: Smart grid and smart metering,” in *Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking*, 2010, pp. 115–118.

[2] G. Kalogridis, C. Efthymiou, S. Z. Denic, T. A. Lewis, and R. Cepeda, “Privacy for smart meters: Towards undetectable appliance load signatures,” in *2010 First IEEE International Conference on Smart Grid Communications*, IEEE, 2010, pp. 232–237.

[3] L. Lyu, K. Nandakumar, B. Rubinstein, J. Jin, J. Bedo, and M. Palaniswami, “Ppfa: Privacy preserving fog-enabled aggregation in smart grid,” *IEEE Transactions on Industrial Informatics*, vol. 14, no. 8, pp. 3733–3744, 2018.

[4] G. Eibl and D. Engel, “Differential privacy for real smart metering data,” *Computer Science-Research and Development*, vol. 32, no. 1-2, pp. 173–182, 2017.

[5] G. Ács and C. Castelluccia, “I have a dream!(differentially private smart metering),” in *International Workshop on Information Hiding*, Springer, 2011, pp. 118–132.

[6] D. Varodayan and A. Khisti, “Smart meter privacy using a rechargeable battery: Minimizing the rate of information leakage,” in *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2011, pp. 1932–1935.

[7] L. Sankar, S. R. Rajagopalan, S. Mohajer, and H. V. Poor, “Smart meter privacy: A theoretical framework,” *IEEE Transactions on Smart Grid*, vol. 4, no. 2, pp. 837–846, 2012.

[8] M. Savi, C. Rottondi, and G. Verticale, “Evaluation of the precision-privacy tradeoff of data perturbation for smart metering,” *IEEE Transactions on Smart Grid*, vol. 6, no. 5, pp. 2409–2416, 2015.

[9] Z. Zhang, W. Cao, Z. Qin, L. Zhu, Z. Yu, and K. Ren, “When privacy meets economics: Enabling differentially-private battery-supported meter reporting in smart grid,” in *2017 IEEE/ACM 25th International Symposium on Quality of Service (IWQoS)*, IEEE, 2017, pp. 1–9.

[10] U. B. Baloglu and Y. Demir, “Lightweight privacy-preserving data aggregation scheme for smart grid metering infrastructure protection,” *International Journal of Critical Infrastructure Protection*, vol. 22, pp. 16–24, 2018.

[11] L. Da Xu, W. He, and S. Li, “Internet of things in industries: A survey,” *IEEE Transactions on industrial informatics*, vol. 10, no. 4, pp. 2233–2243, 2014.

[12] B. J. Murrill, E. C. Liu, and R. M. Thompson, “Smart meter data: Privacy and cybersecurity,” Congressional Research Service, Library of Congress, 2012.

[13] E. L. Quinn, “Privacy and the new energy infrastructure,” *Available at SSRN 1370731*, 2009.

[14] F. Li, B. Luo, and P. Liu, “Secure information aggregation for smart grids using homomorphic encryption,” in *2010 first IEEE international conference on smart grid communications*, IEEE, 2010, pp. 327–332.

[15] R. K. Barik, S. K. Gudey, G. G. Reddy, M. Pant, H. Dubey, K. Mankodiya, and V. Kumar, “Foggrid: Leveraging fog computing for enhanced smart grid network,” in *2017 14th IEEE India Council International Conference (INDICON)*, IEEE, 2017, pp. 1–6.

[16] J.-N. Liu, J. Weng, A. Yang, Y. Chen, and X. Lin, “Enabling efficient and privacy-preserving aggregation communication and function query for fog computing-based smart grid,” *IEEE Transactions on Smart Grid*, vol. 11, no. 1, pp. 247–257, 2019.

[17] J.-M. Bohli, C. Sorge, and O. Ugus, “A privacy model for smart metering,” in *2010 IEEE International Conference on Communications Workshops*, IEEE, 2010, pp. 1–5.

[18] C. Dwork, F. McSherry, K. Nissim, and A. Smith, “Calibrating noise to sensitivity in private data analysis,” in *Theory of cryptography conference*, Springer, 2006, pp. 265–284.

[19] C. Dwork, A. Roth, *et al.*, “The algorithmic foundations of differential privacy.,” *Foundations and Trends in Theoretical Computer Science*, vol. 9, no. 3-4, pp. 211–407, 2014.

[20] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor, “Our data, ourselves: Privacy via distributed noise generation,” in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Springer, 2006, pp. 486–503.

[21] C. Clifton and T. Tassa, “On syntactic anonymity and differential privacy,” in *2013 IEEE 29th International Conference on Data Engineering Workshops (ICDEW)*, IEEE, 2013, pp. 88–93.

[22] K. Kursawe, G. Danezis, and M. Kohlweiss, “Privacy-friendly aggregation for the smart-grid,” in *International Symposium on Privacy Enhancing Technologies Symposium*, Springer, 2011, pp. 175–191.

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

[24] T.-H. H. Chan, E. Shi, and D. Song, “Privacy-preserving stream aggregation with fault tolerance,” in *International Conference on Financial Cryptography and Data Security*, Springer, 2012, pp. 200–214.

[25] V. Rastogi and S. Nath, “Differentially private aggregation of distributed time-series with transformation and encryption,” in *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, 2010, pp. 735–746.

[26] Z. Erkin and G. Tsudik, “Private computation of spatial and temporal power consumption with smart meters,” in *International Conference on Applied Cryptography and Network Security*, Springer, 2012, pp. 561–577.

[27] K. Mahmood, S. A. Chaudhry, H. Naqvi, S. Kumari, X. Li, and A. K. Sangaiah, “An elliptic curve cryptography based lightweight authentication scheme for smart grid communication,” *Future Generation Computer Systems*, vol. 81, pp. 557–565, 2018.

[28] K. Gai, Y. Wu, L. Zhu, L. Xu, and Y. Zhang, “Permissioned blockchain and edge computing empowered privacy-preserving smart grid networks,” *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 7992–8004, 2019.

[29] S. Barker, A. Mishra, D. Irwin, E. Cecchet, P. Shenoy, J. Albrecht, *et al.*, “Smart*: An open data set and tools for enabling research in sustainable homes,” *SustKDD, August*, vol. 111, no. 112, p. 108, 2012.

[30] M. U. Hassan, M. H. Rehmani, R. Kotagiri, J. Zhang, and J. Chen, “Differential privacy for renewable energy resources based smart metering,” *Journal of Parallel and Distributed Computing*, vol. 131, pp. 69–80, 2019.

[31] Y. Huo, C. Yong, and Y. Lu, “Re-adp: Real-time data aggregation with adaptive-event differential privacy for fog computing,” *Wireless Communications and Mobile Computing*, vol. 2018, 2018.

[32] V. Costan and S. Devadas, “Intel sgx explained.,” *IACR Cryptol. ePrint Arch.*, vol. 2016, no. 86, pp. 1–118, 2016.

[33] L. Lyu, J. Jin, S. Rajasegarar, X. He, and M. Palaniswami, “Fog-empowered anomaly detection in iot using hyperellipsoidal clustering,” *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1174–1184, 2017.

[34] G. Kellaris, S. Papadopoulos, X. Xiao, and D. Papadias, “Differentially private event sequences over infinite streams,” *Proceedings of the VLDB Endowment*, vol. 7, no. 12, pp. 1155–1166, 2014.

[35] W. Diffie and M. Hellman, “New directions in cryptography,” *IEEE transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[36] N.-F. Standard, “Announcing the advanced encryption standard (aes),” *Federal Information Processing Standards Publication*, vol. 197, no. 1-51, pp. 3–3, 2001.

[37] H. Krawczyk, M. Bellare, and R. Canetti, *Hmac: Keyed-hashing for message authentication*, 1997.

[38] M. Bellare, R. Canetti, and H. Krawczyk, “Keying hash functions for message authentication,” in *Annual international cryptology conference*, Springer, 1996, pp. 1–15.

[39] Wikipedia contributors, *Hmac — Wikipedia, the free encyclopedia*, `https://en.wikipedia.org/w/index.php?title=HMAC&oldid=1009547100`, [Online; accessed 7-March-2021], 2021.

[40] G. Hebrail and A. Berard. (). “Individual household electric power consumption data set,” [Online]. Available: `https://archive.ics.uci.edu/ml/datasets/individual+household+electric+power+consumption#`. (accessed: 08.11.2020).

[41] F. Farokhi, “Temporally discounted differential privacy for evolving datasets on an infinite horizon,” in *2020 ACM/IEEE 11th International Conference on Cyber-Physical Systems (ICCPS)*, IEEE, 2020, pp. 1–8.