Programmable Crypto-Control for IoT Networks: An Application in Networked Microgrids

Lizhi Wang, *Student Member, IEEE*, Peng Zhang, *Senior Member, IEEE*, Zefan Tang, *Member, IEEE*, and Yanyuan Qin, *Member, IEEE*

***Abstract*—In collaborative networked microgrids (NMs), distributed energy resources (DERs) utilize intelligent IoT based controllers to coordinately support various smart community functions. Meanwhile, privacy and security issues occur when IoT-based controllers interact with each other. This paper presents a cryptography-based, programmable control (crypto-control) scheme to provably preserve the privacy of DERs while ensuring fast, flexible distributed control in NMs. Specifically, it makes the following contributions: 1) a programmable crypto-control-based NMs (PCNMs) architecture, where crypto-controllers are fully virtualized, is devised; 2) a novel dynamic encrypted weight addition (DEWA) approach, which integrates an enhanced partial homomorphic encryption and a secret sharing scheme, is devised to ensure privacy preserving of distributed controls; 3) the DEWA privacy-preserving property is mathematically analyzed; and 4) a real-time DEWA-based PCNMs testbed is deployed by incorporating DEWA, real software defined networking switches, and IoT devices. The deployable crypto-control scheme is interfaced with and thoroughly verified in a Real-Time Digital Simulator environment, and the experimental results validate the effectiveness, benefits, and superiority of DEWA-based PCNMs. The inherent resilience of the DEWA-based PCNMs is validated by small signal stability analyses.**

***Index Terms*—IoT, privacy, networked microgrids, cryptography, software defined networking, distributed control**

I. Introduction

NETWORKING a group of microgrids to form networked microgrids (NMs) greatly enhances the grid resiliency [1], [2]. With the increasing integration of distributed energy resources (DERs) in NMs, distributed-consensus-based secondary controls are currently attracting more and more attention due to their enhanced flexibility and resiliency over centralized controls [3]. With a distributed-consensus-based secondary control, some information has to be transmitted to and is later processed at neighboring agents. For instance, to achieve certain control functions, some DERs need to share their individual information with their neighbors. However, this inevitably poses a privacy threat, i.e., the information (which can be sensitive) sent from one DER to its neighbors is revealed at the neighbors' sides. In other words, the DER who sends information to its neighbors may not want the information to be revealed; meanwhile, the information should be processed at the neighbors' sides [4], [5].

This work was supported in part by the National Science Foundation under Grant No. OIA-2134840. This work relates to Department of Navy award N00014-22-1-2001 issued by the Office of Naval Research. The United States Government has a royalty-free license throughout the world in all copyrightable material contained herein.

L. Wang and P. Zhang are with the Department of Electrical and Computer Engineering, Stony Brook University, NY 11794, USA (e-mail: p.zhang@stonybrook.edu).

Z. Tang is with Interdisciplinary Science Department, Brookhaven National Laboratory, Upton, NY 11973, USA.

Y. Qin is with the Department of Computer Science, University of Connecticut, Storrs, CT 06269, USA.

A traditional method to preserve the privacy in NMs (i.e., the so-called differential-privacy method) adds a certain noise to each sharing message [6]. By introducing a random perturbation to each sharing message, neighbors cannot obtain true messages [7]. Some examples that have adopted this method include the optimal power flow privacy protection [8] and transmission lines and transformers parameters privacy protection [9]. However, with noises added, the iterated control results are inevitably impacted, and the more the noises are added (which improves the privacy), the more seriously the system operation is affected [10].

An alternative method to address the privacy issue is reducing the number of sensitive sharing parameters through re-designing the control strategy [11], [12]. For instance, in [11], a privacy-preserving distributed control strategy is proposed for active power sharing in islanded microgrids, where DERs only need to exchange their frequencies while other sensitive parameters are kept privately. [12] re-designs the energy management strategy for microgrids, where each customer's schedulable demand remains private (other non-sensitive parameters are shared instead). However, while this method can preserve the privacy of certain parameters, it is based on specific problems, i.e., for a different problem, a different control strategy needs to be carried out. Further, this method still requires certain parameters to be transmitted. As different parameters in NMs have certain correlations, by checking the pattern of the received information, each neighbor can still obtain the pattern of the sensitive information, meaning that the privacy is not fully preserved.

Privacy issues in the Internet of Things (IoT) have been analyzed by a variety of groups. For instance, in [13], some common lightweight cryptographic algorithms used in the IoT are analyzed. Examples include the AES cipher, the TEA cipher, the RSA scheme, and some hash functions. [14] presents a ring signcryption scheme for heterogeneous IoT data transmission between sensors and a server. This scheme achieves confidentiality, integrity, authentication, non-repudiation, and anonymity without the need of certificates. However, the performance of this scheme on different devices has not been analyzed. Similarly, although [15] discusses some existing security and privacy-preserving solutions in the IoT, it does not test cryptographic schemes on different devices and does not provide the practical results of privacy-preserving methods.

Recently, the partial homomorphic encryption (PHE) method has been widely used to preserve privacy [16]–[19]. With PHE, multiple communicating parties send their encrypted messages (namely, ciphertexts) to their neighbors, respectively, and when each party receives all the neighbors' ciphertexts, the product of all the ciphertexts is decrypted. The unique feature of PHE is that the product of multiple ciphertexts decrypts to the sum of their corresponding plaintexts. All of these papers aim to develop and design effective cryptographic protocols and mechanisms to secure IoT applications or services. One of the major common concerns of these papers is that cryptographic operations must be computationally cheap so that they can be implemented in resource-constrained IoT devices. A lightweight and privacy-preserving two-factor authentication scheme for IoT devices, where physically uncloneable functions have been considered as one of the authentication factors, is presented in [20]. However, although existing cryptography-based methods for IoT provide unique solutions to protect each plaintext from being observed directly, the privacy of each communicating party still relies on the honesty of neighbors, as each party has the capability of decrypting the ciphertexts of each neighbor [16]. Moreover, none of the papers talks about the application of cryptography in private cooperative control for IoT devices. The direct application of PHE in the IoT can still leak the privacy of control actions of agents. In addition, PHE relies on long keys to ensure a high-security level; this, however, inevitably leads to heavy computation and communication burdens [19], causing crypto-controllers to work slowly. Further, when any communicating party fails to work, the PHE-based system operates abnormally. A provably private, fast, and flexible cryptography-based control scheme for NMs is therefore needed but does not yet exist.

To bridge the gaps, in this paper, a cryptography-based, programmable control (crypto-control) scheme is designed to provably preserve the privacy of DERs while ensuring fast, flexible distributed control in NMs. Specifically, we devise a programmable crypto-control-based NMs (PCNMs) architecture, where crypto-controllers are fully virtualized (thus, when a crypto-controller fails to work, a backup one can be turned on flexibly) and software defined networking (SDN) is utilized to manage the network. A novel dynamic encrypted weight addition (DEWA) approach, which integrates a new switching-keys-enabled partial homomorphic encryption and a secret sharing scheme, is developed to enable privacy-preserving of distributed IoT controls in real time. Differently from PHE, DEWA uses dynamic, light-weight keys to provide fast, distributed control in NMs (while the security level remains high), and adopts the zero secret sharing technique to preserve each DER's privacy. More specifically, the values of the states, control actions and control gains of each agent are hidden from the rest of the participants. We establish a real-time DEWA-based PCNMs testbed incorporating DEWA, real SDN switches, and IoT devices, validate the hardware testbed in a Real-Time Digital Simulator (RTDS) environment, and mathematically and experimentally analyze the DEWA privacy-preserving property. Test results validate the effectiveness, benefits, and superiority of DEWA-based PCNMs, and the small signal stability result is also provided.

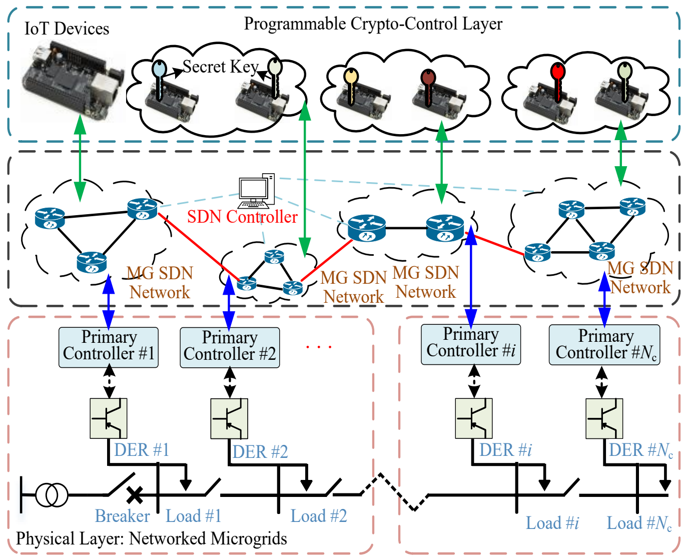

Fig. 1. The PCNMs architecture.

The remainder of this paper is organized as follows: Section II describes PCNMs and privacy requirements. DEWA and the mathematical privacy analysis are presented in Sections III and IV, respectively. Section V provides the DEWA-based PCNMs testbed and test results. Section VI concludes the paper.

II. PCNMs and Privacy Requirements

In this section, we first present PCNMs including its architecture and the control strategy, and then describe the privacy requirements in PCNMs.

A. Architecture of PCNMs

The PCNMs architecture is illustrated in Fig. 1. It consists of three layers: 1) a physical layer, 2) a network layer, and 3) a programmable crypto-control layer. The components in the physical layer contain various DERs (such as solar panels, wind turbines, diesels, and storages), and others like smart meters, loads, and transformers. The measurements of each DER (e.g., power generation, frequency, etc.) are sent to its crypto-controller with a regular frequency via a communication network. The function of the crypto-controller is to realize consensus-based secondary control while preserving DERs' privacy. Each DER communicates with its programmable crypto-controller through an IP-address assigned interface that can adopt different communication protocols.

The network layer utilizes the intelligent SDN switches to achieve a fast and flexible communication environment. The decoupling of control and data planes and the centralization of the control logic in the SDN controller make SDN switches simple forwarding devices, where the SDN controller obtains a global knowledge of network states, enabling the fast development of sophisticated applications. In this study, the communication data paths for crypto-controllers are regulated by the SDN controller, for handling the communication congestion to achieve a reliable communication network [21], [22].

In PCNMs, each crypto-controller is installed in an IoT device. Crypto-controllers communicate with each other through IP-address assigned interfaces. The SDN- and IoT-based

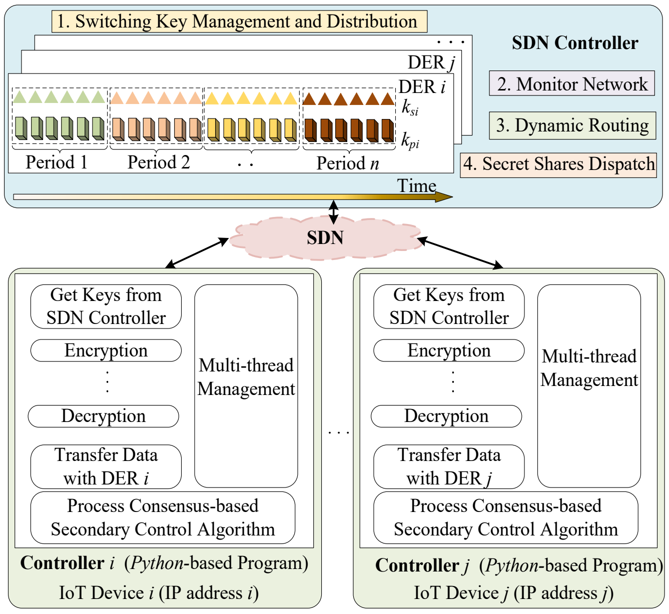
1. Switching Key Management and Distribution
SDN Controller
DER j
DER i
k_{si}
k_{pi}
Period 1
Period 2
Period n
Time
2. Monitor Network
3. Dynamic Routing
4. Secret Shares Dispatch
SDN
Get Keys from SDN Controller
Encryption
Decryption
Transfer Data with DER i
Multi-thread Management
Process Consensus-based Secondary Control Algorithm
Controller i (Python-based Program)
IoT Device i (IP address i)
Transfer Data with DER j
Controller j (Python-based Program)
IoT Device j (IP address j)

Fig. 2. The SDN- and IoT-based crypto-controller structure.

crypto-controller structure is illustrated in Fig. 2, where each crypto-controller is a program developed in Python containing multiple threads. Secure keys are sent from the SDN controller to each crypto-controller every a certain time period. Specifically, each crypto-controller has a private key (i.e., k_{si} for crypto-controller i) and some public keys (i.e., k_{pj}, where $j \in \mathcal{N}_i$ and $\mathcal{N}_i$ denotes the set of crypto-controller i's neighbors). k_{si} is used as a decryption key to decrypt each ciphertext sent from a neighbor. k_{pj} is used to encrypt a plaintext. Each crypto-controller encrypts DER state variables and sends them to neighboring crypto-controllers using the public keys of neighbors. When each crypto-controller receives each message sent from a neighboring crypto-controller, it decrypts the message using its own private key and processes the consensus-based secondary control.

The uniqueness of the PCNMs is that the secondary controllers are pushed to the edge of the system and are virtualized in IoT devices with great scalability and flexibility. Moreover, SDN is leveraged to make the communication network suitable for the deployment of crypto-controllers and capable of handling the heavy communication burden caused by encryption. Meanwhile, instead of introducing a centralized controller, the SDN controller is in charge of the keys management and distribution without affecting crypto-controller operations. Note that the security of SDN itself can be guaranteed by various comprehensive solutions (see [23] for a detailed description).

B. Formulations of Programmable Control

Without loss of generality, a two-layer hierarchical control strategy is adopted in this work. Specifically, the two-layer hierarchical control strategy consists of a droop control and a consensus-based secondary control. The droop control is utilized to regulate the frequency and voltage. For the secondary control, each DER communicates with its neighboring DERs through transmitting required control signals to achieve power sharing and voltage and frequency restorations in PCNMs. This section presents the formulations of this control strategy and demonstrates the privacy leakage issue.

For the droop control, voltage and frequency droop characteristics are given as follows:

$$\begin{cases} \boldsymbol{\omega} = \omega^* - \boldsymbol{m}_p(\boldsymbol{P} - \boldsymbol{P}^*) \\ \boldsymbol{E} = \boldsymbol{E}^* - \boldsymbol{n}_q(\boldsymbol{Q} - \boldsymbol{Q}^*), \end{cases} \tag{1}$$

where $\boldsymbol{\omega}$ denotes a vector containing all the DERs' frequencies, and $\boldsymbol{E}$ denotes a vector containing all the DERs' output voltage magnitudes. $\boldsymbol{P}$ and $\boldsymbol{Q}$ denote two vectors containing the active and reactive power outputs of all the DERs, respectively. ω^*, $\boldsymbol{E}^*$, $\boldsymbol{P}^*$ and $\boldsymbol{Q}^*$ respectively denote vectors containing nominal values of each signal. $\boldsymbol{m}_p$ and $\boldsymbol{n}_q$ are two vectors containing the active and reactive power droop gains of all the DERs, respectively.

When only the droop control is employed, ω^* and $\boldsymbol{E}^*$ are fixed, resulting in that any load change causes voltage and frequency to deviate from their set points. A secondary control that eliminates the frequency and voltage deviations caused by the primary droop control can address this issue. In this study, we introduce a distributed-consensus-based secondary control that allows the voltage, frequency and shared power to converge to their reference values. Specifically, this controller introduces a secondary frequency control variable Ω_i for frequency regulation and a secondary voltage control variable e_i for voltage regulation. Mathematical formulations of this controller are as follows:

With droop and secondary controls, DER i's frequency ω_i can be regulated as follows:

$$\begin{cases} \omega_i = \Omega_i - m_{p,i}(P_i - P_i^*) \\ \Omega_i = \displaystyle\int (k_i^{\omega}[\sum_{j\in\mathcal{N}_i} a_{ij}(\omega_j - \omega_i) + g_i(\omega^{ref} - \omega_i)] \\ \qquad + k_i^P[\displaystyle\sum_{j\in\mathcal{N}_i} a_{ij}(m_jP_j - m_iP_i)])\mathrm{d}t, \end{cases} \tag{2}$$

where $j \in \mathcal{N}_i$ refers to the j^{th} neighboring DER of DER i. ω^{ref} is the reference value of ω_i. a_{ij} is the fixed communication weight for DERs i and j. The finite-time consensus can be achieved through setting the value of a_{ij} [24]. g_i is assigned to 1 if DER i knows the value of ω^{ref}, and otherwise it is set to zero. k_i^{ω} and k_i^P are control gains.

DER i's voltage E_i can be regulated as follows:

$$\begin{cases} E_i = e_i - n_{q,i}(Q_i - Q_i^*) \\ e_i = \displaystyle\int k_i^E[\sum_{j\in\mathcal{N}_i} a_{ij}(E_j - E_i) + g_i(E^{ref} - E_i)]\mathrm{d}t, \end{cases} \tag{3}$$

where E^{ref} refers to the voltage reference, and k_i^E is the control gain.

Specifically, DER i's neighboring DERs can be divided into two groups, i.e., DERs in the same microgrid (with DER i) and those in other microgrids. Let $\mathcal{N}_i = \mathcal{N}_i^1 \cup \mathcal{N}_i^2$, where $\mathcal{N}_i^1$ denotes the set of DER i's neighboring DERs in the same microgrid (i.e., Intro-MG), and $\mathcal{N}_i^2$ denotes the set of DER i's neighboring DERs in other microgrids (i.e., Inter-MGs). In this study, we only consider the privacy protection of parameters from Inter-MGs. To clearly demonstrate the privacy leakage issue, we reformulate (2) and (3). For simplicity, three control variables (i.e., u_i^ω, u_i^E, and u_i^P) are used to represent the three complex components in (2) and (3), respectively, as follows:

$$\begin{cases} u_i^\omega = k_i^\omega[\sum\limits_{j\in\mathcal{N}_i} a_{ij}(\omega_j-\omega_i)+g_i(\omega^{ref}-\omega_i)] \\ u_i^E = k_i^E[\sum\limits_{j\in\mathcal{N}_i} a_{ij}(E_j-E_i)+g_i(E^{ref}-E_i)] \\ u_i^P = k_i^P[\sum\limits_{j\in\mathcal{N}_i} a_{ij}(m_jP_j-m_iP_i)]. \end{cases} \tag{4}$$

As each DER sends out one data packet (which contains all the states, i.e., ω_i, E_i and P_i) each time, we use an array $\boldsymbol{X}_i$ to represent all the states being sent out, i.e., $\boldsymbol{X}_i=[\omega_i,\ E_i,\ P_i]$. We first rewrite (4) in a compact form as follows:

$$\begin{cases} u_i^\omega = k_i^{\boldsymbol{\omega}}(-\boldsymbol{L}_i\boldsymbol{\omega}+g_i(\omega^{ref}-\omega_i)) \\ u_i^E = k_i^E(-\boldsymbol{L}_i\boldsymbol{E}+g_i(E^{ref}-E_i)) \\ u_i^P = -k_i^P\boldsymbol{L}_i\boldsymbol{P}, \end{cases} \tag{5}$$

where $\boldsymbol{L}_i$ is the i^{th} row of the Laplacian matrix $\boldsymbol{L}$ ($\boldsymbol{L}=[l_{ij}]\subseteq\mathbb{R}^{N_c\times N_c}$ with each element $l_{ij}=\sum_{i=1}^{N_c}a_{ij}-a_{ij}$) and $\boldsymbol{\omega}$, $\boldsymbol{E}$ and $\boldsymbol{P}$ are arrays, i.e., $\boldsymbol{\omega}=[\omega_1,\omega_2,...,\omega_{N_c}]$, $\boldsymbol{E}=[E_1,E_2,...,E_{N_c}]$, and $\boldsymbol{P}=[P_1,P_2,...,P_{N_c}]$, where N_c is the total number of communicating DERs. For simplicity, (5) can be expressed as

$$\boldsymbol{u}_i=\boldsymbol{K}_i[-\widehat{\boldsymbol{K}}\boldsymbol{X}+\boldsymbol{G}(\boldsymbol{X}^{ref}-\boldsymbol{X}_i)], \tag{6}$$

where $\boldsymbol{u}_i=[u_i^\omega,u_i^E,u_i^P]$, $\boldsymbol{K}_i=diag(k_i^\omega,k_i^E,k_i^P)$, $\boldsymbol{X}=[\boldsymbol{\omega},\boldsymbol{E},\boldsymbol{P}]$, $\boldsymbol{X}^{ref}=[\omega^{ref},E^{ref},0]$, $\widehat{\boldsymbol{K}}=diag(\boldsymbol{L}_i,\boldsymbol{L}_i,\boldsymbol{L}_i)$, and $\boldsymbol{G}=diag(g_i,g_i,0)$. To clearly demonstrate the privacy leakage issue, $\boldsymbol{u}_i$ for DER i can be decoupled into $\boldsymbol{u}_i=\boldsymbol{u}_i^1+\boldsymbol{u}_i^2$, where $\boldsymbol{u}_i^1$ and $\boldsymbol{u}_i^2$ respectively represent control variables using the information from Intro-MG and Inter-MGs. Let $\widehat{k}_{ii}$ and $\widehat{k}_{ij}$ ($i,j\in\{1,2,...,N_c\}$) denote the i^{th} and j^{th} element of $\boldsymbol{L}_i$, respectively. $\boldsymbol{u}_i^1$ and $\boldsymbol{u}_i^2$ can be represented as follows:

$$\begin{cases} \boldsymbol{u}_i^1 = \underbrace{-\boldsymbol{K}_i\boldsymbol{G}\boldsymbol{X}_i-\boldsymbol{K}_i\widehat{k}_{ii}\boldsymbol{X}_i-\boldsymbol{K}_i\sum\limits_{j\in\mathcal{N}_i^1}\widehat{k}_{ij}\boldsymbol{X}_j}_{Intro-MG} \\ \quad +\underbrace{\boldsymbol{K}_i\boldsymbol{G}\boldsymbol{X}^{ref}}_{Reference} \\ \boldsymbol{u}_i^2 = \underbrace{-\boldsymbol{K}_i\sum\limits_{j\in\mathcal{N}_i^2}\widehat{k}_{ij}\boldsymbol{X}_j}_{Inter-MGs}. \end{cases} \tag{7}$$

In (7), the variables sent from Intro-MG and Inter-MGs are clearly separated, i.e., the privacy of each neighboring DER in Inter-MGs is contained in $\boldsymbol{X}_j$ (where $j\in\mathcal{N}_i^2$). This setup enables that DEWA (as will be discussed in Section III) can be applied to $\boldsymbol{X}_j$ instead of ω_j, E_j and P_j in different equations.

C. Privacy Requirement in PCNMs

From (7), each crypto-controller, $i\in\{1,2,...,N_c\}$, computes $\boldsymbol{u}_i$ using the states from Intro-MG and Inter-MGs, i.e., it locally aggregates the contributions of its neighbors $j\in\mathcal{N}_i^2$, the privacy of which has to be preserved. Meanwhile, the weights $\widehat{k}_{ij}$ and $\widehat{k}_{ji}$ are commonly the same in the practical applications of microgrids, and this inevitably poses challenges for the privacy-preserving. For PCNMs, we consider that weights are unknown to all the crypto-controllers, and the following privacy requirements need to be satisfied:

- Crypto-controller i can obtain $\boldsymbol{u}_i^0$ and $\sum_{j\in\mathcal{N}_i^2}\widehat{k}_{ij}\boldsymbol{X}_j$ within each time period, while $\boldsymbol{X}_j$, $\widehat{k}_{ij}$, and $\widehat{k}_{ij}\boldsymbol{X}_j$ remain unrevealed.
- Any other participating crypto-controller cannot obtain $\boldsymbol{X}_j$ of other participants in the computation.
- Even if crypto-controller i collaborates with some other crypto-controllers, all the crypto-controllers' $\boldsymbol{X}_j$s cannot be revealed.
- The public-key encryption scheme should be light-weight to achieve a real-time computation with tolerant computation delays, ensuring high stability of PCNMs.

III. Dynamic Encrypted Weighted Addition

DEWA combines PHE with a secret sharing scheme to enable privacy preserving of distributed algorithms. Here PHE is redesigned with switching keys to preserve each DER's privacy while ensuring real-time computations. In PHE scheme, all the communicating DERs send their encrypted messages (i.e., encrypted $\widehat{k}_{ij}\boldsymbol{X}_j$ from DER j to DER i) to their neighbors, respectively, and when each party receives all the neighbors' ciphertexts, the product of all the ciphertexts is decrypted. A unique feature of PHE is that the product of multiple ciphertexts decrypts to the sum of their corresponding plaintexts (i.e., $\sum_{j\in\mathcal{N}_i^2}\widehat{k}_{ij}\boldsymbol{X}_j$, to be used in (7)). However, in PHE, each communicating DER needs to know the communication weights between itself and neighbors, i.e., $\widehat{k}_{ij}$ for DER j to encrypt $\widehat{k}_{ij}\boldsymbol{X}_j$. This makes that although PHE provides a unique solution to protect each plaintext from being observed directly (i.e., only $\sum_{j\in\mathcal{N}_i^2}\widehat{k}_{ij}\boldsymbol{X}_j$ is observed), each party still has the capability of decrypting each neighbor's ciphertexts (i.e., DER i can decrypt the encrypted $\widehat{k}_{ij}\boldsymbol{X}_j$ individually to obtain $\boldsymbol{X}_j$).

To address this challenge, in DEWA, each communication weight between two DERs is not revealed to DERs. Instead of sending the encrypted $\widehat{k}_{ij}\boldsymbol{X}_j$ from DER j to DER i, DER j sends $\mathcal{E}(\widehat{k}_{ij})^{\boldsymbol{X}_j}\mathcal{E}(s_{ji})$ to DER i, where $\mathcal{E}(\widehat{k}_{ij})$ is the encrypted $\widehat{k}_{ij}$ using DER i's public key, and is sent from the SDN controller to DER j. $\mathcal{E}(s_{ji})$ is the encrypted s_{ji} (i.e., the so-called zero secret share, a random number designed within the SDN controller) using DER i's public key, and s_{ji} is sent from the SDN controller to DER j. Since DER j does not have DER i's private key, it cannot decrypt $\mathcal{E}(\widehat{k}_{ij})$ to obtain $\widehat{k}_{ij}$. When DER i receives $\mathcal{E}(\widehat{k}_{ij})^{\boldsymbol{X}_j}\mathcal{E}(s_{ji})$ from DER j, it cannot obtain $\widehat{k}_{ij}$ and $\boldsymbol{X}_j$, as the random number s_{ji} is not known to DER i. The private information of DER j (i.e., $\boldsymbol{X}_j$) is thus protected from being observed by DER i.

Meanwhile, the Paillier cryptosystem (i.e., a type of PHE) [25] is adopted in DEWA. In addition to having the common feature of a traditional PHE (i.e., the product of multiple ciphertexts decrypts to the sum of their corresponding plaintexts), Paillier has another silent feature, i.e.,

$\mathcal{E}(\widehat{k}_{ij})^{\boldsymbol{X}_j} = \mathcal{E}(\widehat{k}_{ij}\boldsymbol{X}_j)$. With these two features, after decrypting the product of all the neighbors' ciphertexts, DER i obtains $\sum_{j\in\mathcal{N}_i^2} \widehat{k}_{ij}\boldsymbol{X}_j + \sum_{j\in\mathcal{N}_i^2} s_{ji}$. Further, in DEWA, we utilize the additive zero secret sharing technique [26] such that s_{ji} is designed in a way that $\sum_{j\in\mathcal{N}_i^2} s_{ji} = 0$. Therefore, with DEWA, DER i can successfully obtain $\sum_{j\in\mathcal{N}_i^2} \widehat{k}_{ij}\boldsymbol{X}_j$, while both $\widehat{k}_{ij}$ and $\boldsymbol{X}_j$ remain unrevealed.

In this section, we first present some backgrounds of the Paillier cryptosystem and the additive zero secret sharing, and then describe our DEWA algorithm with more details.

A. The Paillier Cryptosystem

1) Overview of The Paillier Cryptosystem: The Paillier cryptosystem contains three components, namely, a key generation part (denoted as Keygen()), an encryption part $c = \mathcal{E}(m)$ (where m and c denote the plaintext and the ciphertext, respectively), and a decryption part $m = \mathcal{D}(c)$. These three components are described in detail as follows:

- Keygen(): The objective of Keygen() is to output a public key k_p (used to encrypt the plaintext) and a private key k_s (used to decrypt the ciphertext). The procedures of Keygen() are given below:
  1) Two large prime numbers (i.e., p and q) with the same bit length are selected in a way that the greatest common divisor of pq and $(p-1)(q-1)$ is one.
  2) Let $n = pq$ and λ be the least common multiple of $p-1$ and $q-1$.
  3) Select a random integer $g \in \{1, 2, ..., n^2 - 1\}$.
  4) Let $\mu = \frac{n}{(g^\lambda \mod \mathrm{n}^2)-1}$.
  5) Then, the public key k_p can be represented by a group of two integers, n and g, i.e., $k_p : (n, g)$, and the private key k_s is the set of λ and μ, i.e., $k_s : (\lambda, \mu)$.
- $\mathcal{E}(m)$: With the public key k_p, $\mathcal{E}(m)$ is used to encrypt the plaintext m and generate the ciphertext c. Specifically, a random integer $r \in \{1, 2, ..., n-1\}$ is first selected, in a way that the greatest common divisor of r and n is one. Then, the ciphertext c is obtained as $c = g^m r^n \mod \mathrm{n}^2$, where the plaintext m has to satisfy $0 \le m < n$.
- $\mathcal{D}(c)$: $\mathcal{D}(c)$ is used to decrypt the ciphertext c and obtain the plaintext m using the private key k_s. Specifically, m can be obtained as $m = \mu\frac{(c^\lambda \mod \mathrm{n}^2)-1}{n} \mod n$.

2) Homomorphism: A great benefit of the Paillier cryptosystem lies in its homomorphic properties, i.e., the product of multiple ciphertexts decrypts to the sum of their corresponding plaintexts and $\mathcal{E}(\widehat{k}_{ij})^{\boldsymbol{X}_j} = \mathcal{E}(\widehat{k}_{ij}\boldsymbol{X}_j)$, as described before. Mathematically, with two plaintexts m_1 and m_2, the two salient features can be represented as follows:

$$\begin{cases} \mathcal{D}(\mathcal{E}(m_1)\mathcal{E}(m_2)) = m_1 + m_2 \\ \mathcal{D}(\mathcal{E}(m_1)^{m_2}) = m_1 m_2. \end{cases} \tag{8}$$

B. Additive Zero Secret Sharing

Directly applying the Paillier cryptosystem in PCNMs is not well suited, as the ciphertext $\mathcal{E}(\widehat{k}_{ij})^{\boldsymbol{X}_j}$ sent from controller j can be inferred with controller i's private key. To address this challenge, the additive zero secret sharing strategy is adopted. With this strategy, a secret message (i.e., zero) is splitted into multiple random shares, which are then sent to a number of parties, respectively. Let s_{ji} denote each zero secret share sent to the j^{th} neighbor of DER i. Instead of sending the encrypted $\widehat{k}_{ij}\boldsymbol{X}_j$ from DER j to DER i, DER j sends $\mathcal{E}(\widehat{k}_{ij})^{\boldsymbol{X}_j}\mathcal{E}(s_{ji})$ to DER i. Mathematically, all the zero secret shares distributed to DER i's neighbors have to satisfy $\sum_{j\in\mathcal{N}_i} s_{ji} = 0$.

C. The DEWA Algorithm

With the Paillier cryptosystem and additive zero secret sharing, DEWA is developed in this paper. The algorithm is given in Algorithm 1, and the procedures are as follows:

Algorithm 1: The DEWA Algorithm

```
1  ▷ Input: The number of crypto-controllers $N_c$, the key pairs
      update period $T$.
2  ▷ Offline key pairs generation:
3  Output: Public key $k_{pi}$ and private key $k_{si}$.
4  for $i \leftarrow 1$ to $N_c$ do
5      Use Keygen() to generate several key pairs;
6  end
7  ▷ Online $u_i$ calculation:
8  for each period $T$ do
9      SDN controller generates zero secret shares $s_{ji}$;
10     SDN controller encrypts $\mathcal{E}_{k_{pi}}(\widehat{k}_{ij})$;
11     Send $k_{pi}$, $\mathcal{E}_{k_{pi}}(\widehat{k}_{ij})$ and $s_{ji}$ to controller $j$;
12     Send $k_{si}$ to controller $i$;
13     while True do
14         Controller $j$ listens to packets of $\boldsymbol{X}_j$ sent from
             RTDS;
15         Controller $i$ calculates $\boldsymbol{u}_i^1$;
16         Controller $j$ calculates $c_j$ using (9), and sends to $i$;
17         Controller $i$ calculates product $\boldsymbol{\alpha}_i$ using (10);
18         Controller $i$ decrypts $\boldsymbol{\alpha}_i$ and computes $\boldsymbol{u}_i^2$ using
             (11);
19         Controller $i$ computes $u_i = u_i^1 + u_i^2$ using (7);
20         Controller $i$ sends $u_i$ to RTDS;
21         if current period ends then
22             Break;
23         end
24     end
25 end
26 ▷ Output: The control signal $\boldsymbol{u}_i$, $(i \in \{1, 2, ..., Nc\})$.
```

1) Key pairs generation: Each key pair (i.e., k_p and k_s) for each crypto-controller is generated within the SDN controller. Key pairs are distributed to corresponding crypto-controllers, and are updated regularly.
2) Zero secret shares generation: Zero secret shares are generated using $\sum_{j\in\mathcal{N}_i} s_{ji} = 0$ within the SDN controller. They are then distributed to corresponding crypto-controllers, and are updated regularly.
3) Weight encryption: Each communication weight (i.e., $\widehat{k}_{ij}$) is encrypted within the SDN controller using controller i's public key k_{pi}. The encrypted weight $\mathcal{E}_{k_{pi}}(\widehat{k}_{ij})$ is then sent to controller j.
4) When controller j receives k_{pi}, k_{sj}, s_{ji}, and $\mathcal{E}_{k_{pi}}(\widehat{k}_{ij})$ from the SDN controller, it computes the ciphertext $\boldsymbol{c}_j$ as follows and sends it to controller i:

$$\begin{aligned} \boldsymbol{c}_j &= \mathcal{E}_{k_{pi}}(\widehat{k}_{ij})^{\boldsymbol{X}_j}\mathcal{E}_{k_{pi}}(s_{ji}) \\ &= \mathcal{E}_{k_{pi}}(\widehat{k}_{ij}\boldsymbol{X}_j + s_{ji}). \end{aligned} \tag{9}$$

5) When controller i receives all the ciphertexts from its neighbors, it computes their product $\boldsymbol{\alpha}_i$ as follows:

$$\boldsymbol{\alpha}_i = \prod_{j\in\mathcal{N}_i^2} \boldsymbol{c}_j = \prod_{j\in\mathcal{N}_i^2} \mathcal{E}_{k_{pi}}(\widehat{k}_{ij})^{\boldsymbol{X}_j} \mathcal{E}_{k_{pi}}(s_{ji}). \tag{10}$$

6) Controller i then decrypts $\boldsymbol{\alpha}_i$ and computes the control signal $\boldsymbol{u}_i^2$ (see (7)) as follows:

$$\begin{aligned}\boldsymbol{u}_i^2 &= -\boldsymbol{K}_i \mathcal{D}_{k_{si}}(\boldsymbol{\alpha}_i) \\ &= -\boldsymbol{K}_i(\sum_{j\in\mathcal{N}_i^2} \widehat{k}_{ij}\boldsymbol{X}_j + \sum_{j\in\mathcal{N}_i^2} s_{ji}) = -\boldsymbol{K}_i \sum_{j\in\mathcal{N}_i^2} \widehat{k}_{ij}\boldsymbol{X}_j.\end{aligned} \tag{11}$$

7) Error correction: During the key pairs dynamic updating process, it is likely to have a mismatch issue, i.e., k_{pi} and k_{si} are unmatched. To address this issue, we further implement an error correction scheme in DEWA. Specifically, when a mismatch issue is detected (e.g., an abnormal decrypted value is observed), the private key from the previous updating cycle is utilized.

IV. DEWA Privacy Analysis

The DEWA privacy-preserving property is mathematically analyzed in this section. Specifically, we classify the attacks into two categories, namely, internal attacks and external attacks. A major type of internal attack is the so-called honest-but-curious attack. We present how the privacy in DEWA-based PCNMs can be preserved against these attacks and also analyze the security level through a security level index.

A. Privacy Analysis

Following formal privacy analysis procedures [27], we present the DEWA privacy analysis in this subsection. We demonstrate that the private information, including crypto-controllers' states $\boldsymbol{X}_j$ and communication weights $\widehat{k}_{ij}$ cannot be inferred under the honest-but-curious and external attacks.

According to the formal privacy analysis, privacy can be defined as a typical cryptographic game involving two players, namely, a challenger and an adversary. The challenger encrypts messages and the adversary tries to break the ciphertext. The game rules are described as follows:

- The adversary creates two messages with the same length and sends them to the challenger.
- The challenger randomly selects one of the messages, encrypts it, and sends the ciphertext to the adversary.
- When the ciphertext is received by the adversary, the adversary determines which message it is. If the probability that the adversary obtains the correct result is larger than 50%, the privacy is not preserved, and otherwise, the privacy is well protected.

Scenario 1: In this scenario, we consider the honest-but-curious attack (i.e., executing operations normally without an active attack launched) in DEWA-based PCNMs. We denote the set of corrupted (i.e., under the honest-but-curious attack) and uncorrupted crypto-controllers as $\mathcal{C}$ and $\mathcal{U}$. For a corrupted crypto-controller $i \in \mathcal{C}$, some of its neighbors are also corrupted, while some are not. Crypto-controller i can collaborate with its corrupted neighbors, trying to obtain $\boldsymbol{X}_j$ and $\widehat{k}_{ij}$ from an uncorrupted neighboring crypto-controller $j \in \mathcal{U}$.

For a corrupted crypto-controller $i \in \mathcal{C}$, it knows the following information: k_{pj}, k_{pi}, k_{si}, $\{s_{ji}\}_{j\in\mathcal{C}}$, and $\{\widehat{k}_{ij}\}_{j\in\mathcal{C}}$. Note that $\sum_{j\in\mathcal{U}} s_{ji} = -\sum_{j\in\mathcal{C}} s_{ji}$ (where crypto-controller j is the j^{th} neighbor of crypto-controller i). Following the formal privacy analysis, the adversary (i.e., the corrupted crypto-controller i) sends the challenger (i.e., the uncorrupted crypto-controller j) two messages (denoted as $\boldsymbol{X}_j^0$ and $\boldsymbol{X}_j^1$, where $j \in \mathcal{U}$). The challenger randomly selects a message, encrypts it (i.e., $c_j = \mathcal{E}(\widehat{k}_{ij}\boldsymbol{X}_j^b)\mathcal{E}(s_{ji})$, where $b \in \{0,1\}$ and $j \in \mathcal{U}$), and sends the ciphertext to the adversary.

When the adversary receives the ciphertext, its private key k_{si} is used to decrypt the ciphertext, and the adversary obtains $\widehat{k}_{ij}\boldsymbol{X}_j^b + s_{ji}$. Then, the adversary will determine which message ($\boldsymbol{X}_j^0$ or $\boldsymbol{X}_j^1$) is encrypted. Denote the adversary's output as $b' \in \{0,1\}$, which is the guess of b. The probability of successfully determining the message is denoted as $Pr[b' = b|i \in \mathcal{C}]$. As the zero secret shares $\{s_{ij}\}_{j\in\mathcal{U}}$ are random and unknown and the probability of breaking the zero shares scheme is negligible, the probability that the adversary obtains the correct result is as follows:

$$Pr[b' = b|i \in \mathcal{C}] \leqslant \frac{1}{2}. \tag{12}$$

From the aforementioned analysis, we can observe that in DEWA-based PCNMs, the honest-but-curious attack is not able to break the privacy of an uncorrupted neighboring crypto-controller. Privacy is thus well protected.

Scenario 2: In this scenario, we consider a typical external attack, i.e., an external attacker eavesdrops on a communication channel and intercepts the information on the channel.

In this case, the adversary only knows public keys (e.g., k_{pi} for crypto-controller i). The adversary can use a public key k_{pi} to encrypt an arbitrary message. Following the formal privacy analysis, the adversary sends the challenger (i.e., an uncorrupted crypto-controller j) two messages (denoted as $\boldsymbol{X}_j^0$ and $\boldsymbol{X}_j^1$). The challenger randomly selects a message, encrypts it (i.e., $c_j = \mathcal{E}(\widehat{k}_{ij}\boldsymbol{X}_j^b)\mathcal{E}(s_{ji})$, where $b \in \{0,1\}$), and sends the ciphertext to the adversary. Denote the adversary's output as $b' \in \{0,1\}$, which is the guess of b. The probability of successfully determining the message (denoted as $Pr[b' = b]$) is higher than 50% only when the adversary finds a polynomial time distinguishable for the n^{th} residue modulo n^2 (where $n = pq$ as in Keygen()) [25], which is believed to be computationally hard, and therefore the probability of successfully determining the message is not higher than 50%.

In summary, it is computationally infeasible for the external attacker to infer a crypto-controller's states $\boldsymbol{X}_i$ and weights $\widehat{k}_{ij}$ from ciphertexts, and the privacy is well protected.

B. Security Level Analysis

The setting of key length affects the security level in DEWA-based PCNMs. For the switching-key management system, the security level is presented as an index [28], which indicates the computational cost of decrypting the private keys. For the DEWA scheme, the security level index can be represented as:

$$\mathbb{I}_N = \frac{1}{NT} \sum_{j\in\mathcal{I}_t} L_p^j(\theta,\zeta) + \eta, \tag{13}$$

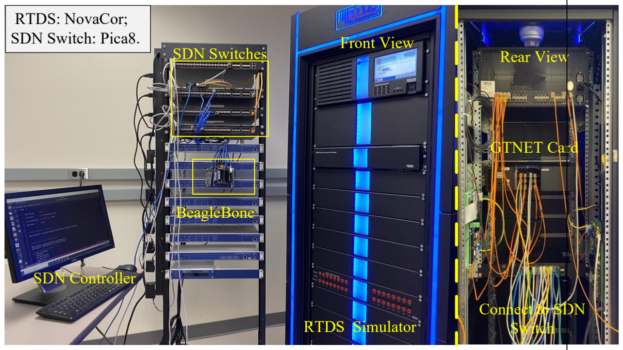

Fig. 3. The PCNMs testbed.

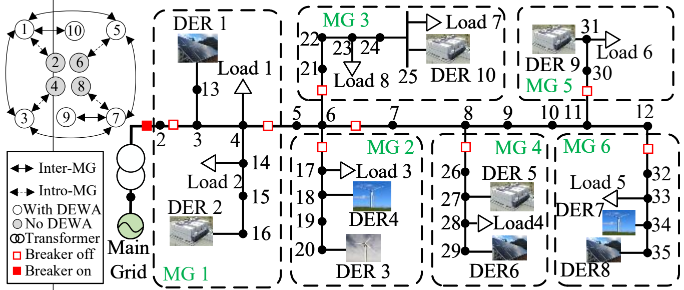

Fig. 4. One-line diagram of the NMs model.

TABLE I
POWER LOADS AT EACH BUS IN FIG. 4

Load 1	Load 2	Load 3	Load 4
$P1$ = 10 kW	$P2$ = 5 kW	$P3$ = 10 kW	$P4$= 5 kW
$Q1$= 5 kVAR	$Q2$= 2 kVAR	$Q3$= 5 kVAR	$Q4$= 2 kVAR
Load 5	Load 6	Load 7	Load 8
$P5$= 5 kW	$P6$ = 15 kW	$P7$ = 10 kW	$P8$ = 15 kW
$Q5$ = 2 kVAR	$Q6$ = 7 kVAR	$Q7$ = 5 kVAR	$Q8$ = 7 kVAR

where $T \in \mathcal{Z}_N := \{1, 2, 3, ..., N\}$ is the update period of key pairs, $\mathcal{I}_t := \{1, 2, 3, ..., l_t\}$ (l_t is the number of key pairs), and $\eta \geq 0$ is the cost incurred to identify the keys update. $L_p(\theta, \zeta)$= exp$\{(\theta + o(1))(\log p)^{\zeta}(\log\log p)^{1-\zeta}\}$, where p is the prime used in the cyclic group for the DEWA scheme, and θ and ζ are determined by an algorithm to solve a discrete logarithm problem.

The index (13) shows that $\exists T \leq N$, $\mathbb{I}_N \geq \frac{1}{N} L_p(\theta, \zeta)$, $\forall N \in \mathcal{Z}$, which means that the proposed DEWA is more secure than the static-key management in terms of the computational cost of decrypting the private keys as long as the selected update period T satisfies $\mathbb{I}_N \geq \frac{1}{N} L_p(\theta, \zeta)$.

V. CASE STUDIES

A. The PCNMs Testbed

The real-time PCNMs testbed is illustrated in Fig. 3. It consists of six IoT devices (i.e., six BeagleBone devices, where programmable crypto-controllers are installed, respectively), four real SDN switches (i.e., Pica8), an SDN controller, GTNET×2 cards, and RTDS hardware. Specifically, the NMs system is developed and compiled in RSCAD, which is designed to interact with the RTDS hardware. The six BeagleBone devices are utilized to virtualize and install the crypto-controllers. These BeagleBone devices are respectively connected to different ports of SDN switches. SDN is used to manage the whole network. It allows for real-time packet monitoring and network configuration, enabling a programmable, reliable, and efficient network management environment. Four SDN switches, managed by an SDN controller through a traditional switch, are used in this study to form a realistic network environment. GTNET×2 cards are utilized for the RTDS hardware to communicate with crypto-controllers. The DEWA algorithm is running within each crypto-controller. The switching key pairs are managed and distributed to each crypto-controller by the SDN controller.

To validate the effectiveness, benefits, and superiority of DEWA-based PCNMs, experimental results produced with the PCNMs testbed are reported. A six-microgrid NMs system is considered in this study. The one-line diagram of the system is shown in Fig. 4. It contains six microgrids including ten DERs and eight power loads, and can operate in either grid-connected or islanded mode depending on whether the circuit breaker between buses 2 and 3 is closed or open, and the communication topology of the NMs is shown on the left side of Fig. 4. In this study, it operates in the islanded mode. The DEWA algorithm is implemented in each BeagleBone device. The real-time communication between each BeagleBone device and the RTDS hardware adopts the User Datagram Protocol (UDP). In this study, a ring-shaped communication topology in the cyber-layer is designed. The sampling rate in the RTDS is set at 35 Hz. The number of key pairs for each controller and the update period T of generated key pairs of k_s and k_p are set at 20 and 50, respectively.

The following subsections are organized into five studies. In the first study, we demonstrate that DEWA has little impact on the system's normal operations, while the traditional PHE method has a larger impact. In the second study, we illustrate that the delay produced by DEWA is only slightly higher than that without encryption. We also conduct the eigenanalysis for both DEWA-based PCNMs and NMs without encryption, and results demonstrate that the DEWA-based PCNMs maintain high stability. In the third study, three types of ciphertexts (i.e., V_1, V_2, and V_3) are recorded respectively with and without DEWA to demonstrate DEWA's effectiveness. The fourth case study illustrates a benefit of using SDN in PCNMs, i.e., the dynamic routing function enabled by SDN greatly improves the resilience of PCNMs. In the fifth study, we present the superiority of DEWA-based PCNMs over the traditional differential-privacy method-based NMs.

B. The Impact of DEWA

In this case, we compare the impacts posed by DEWA and the traditional PHE method on the system's normal operations. The system configurations are the same as in Tables I and II, and the communication topology is the same as in Fig. 4.

1) The Impact of DEWA: Fig. 5 gives frequency and active power responses of different DERs (i.e., DERs 1, 3, 5, 7, 9, and 10) with DEWA, where at time $t = 3s$, Load 1 (see Fig. 4)

TABLE II
DROOP CONTROL AND DEWA PARAMETERS

$m_{P\{1,3,5,7,9\}}$	$3e^{-5}$	$m_{P\{2,4,6,7,10\}}$	$1.5e^{-5}$
$n_{P\{1,3,5,7,9\}}$	0.04	$n_{P\{2,4,6,8,10\}}$	0.02
$k^{\omega}_{\{1,2,\ldots,10\}}$	0.3	$k^{E}_{\{1,2,\ldots,10\}}$	0.3
$k^{P}_{\{1,2,\ldots,10\}}$	0.3	ω^{ref} = $60Hz$	$E^{ref} = 253.9V$

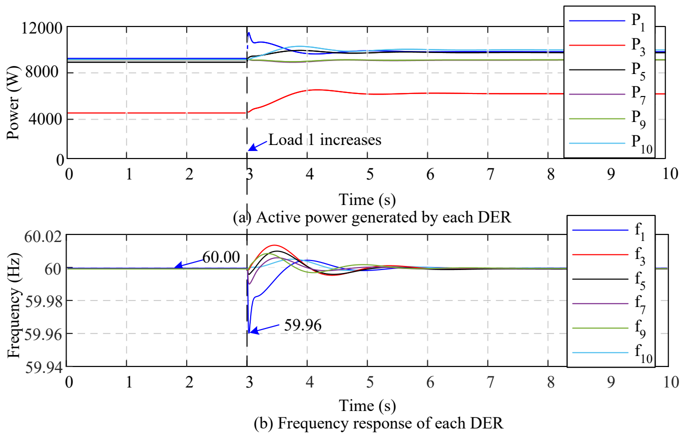

Fig. 5. Frequency and active power responses of DERs with DEWA.

increases from [10 kW, 5 kVAR] to [20 kW, 10 kVAR]. It can be observed that

- Before Load 1 changes, the system has a rated frequency, i.e., 60Hz, meaning that the frequency is regulated well in the steady state.
- After experiencing a disturbance at time $t = 3s$, the system is able to go back to the steady state, and the active powers from different DERs are regulated.

The above observations demonstrate that DEWA has little impact on the system's normal operation.

2) Comparison of A Controller Without Encryption, DEWA, and PHE: With an encryption algorithm, a certain amount of time typically needs to be consumed to encrypt and decrypt data. In this case, the impact of the key size is demonstrated. Specifically, private key sizes are selected as suggested by the NIST recommendation [29]. For the static encryption system, a larger key size commonly leads to improved security; in the meantime, it results in longer computation time, potentially affecting the real-time operation in PCNMs.

Fig. 6 gives the comparison results of $\sum_{j\in\mathcal{N}_1^2} a_{1j}(m_jP_j - m_1P_1)$ under the traditional control (i.e., without encryption), DEWA, and PHE (with a short key and a long one). DEWA has a key size of 160 bits and the update period T is set at 50 cycles. The short key for PHE has 512 bits and the long key has 1024 bits. The value of $\sum_{j\in\mathcal{N}_1^2} a_{1j}(m_jP_j - m_1P_1)$ directly affects the power sharing performance, and the smaller the deviations between $\sum_{j\in\mathcal{N}_1^2} a_{1j}(m_jP_j - m_1P_1)$ from a crypto-control (i.e., DEWA or PHE) and the traditional control (i.e., without encryption) is, the smaller the impact posed by the crypto-control will be. From Fig. 6, it can be seen that:

- The results obtained from the traditional control and DEWA are almost the same, indicating that DEWA has little impact on the system's normal operation.
- The result obtained from PHE with either a short key or a long one largely deviates from that obtained from the traditional control.
- For PHE, the increase of the key size largely affects the system's performance.

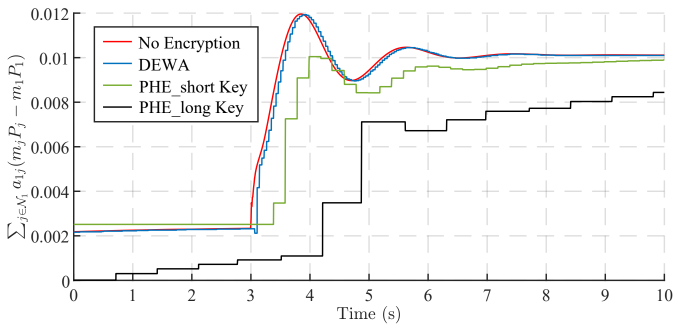

Fig. 6. $\sum_{j\in\mathcal{N}_1} a_{1j}(m_jP_j - m_1P_1)$ in different scenarios.

C. DEWA Delays and Eigenanalysis of DEWA-based PCNMs

In this subsection, we first test the packet loss in the PCNM system, and compare the delays produced by DEWA-based PCNMs and those without encryption. We also conduct the eigenanalysis for both DEWA-based PCNMs and NMs without encryption.

1) Packet loss test: In this case, we test the packet loss using UDP in the PCNMs system. As the microgrid measurements from DERs to crypto-controllers are not encrypted, we only consider the impact of packet loss between crypto-controllers. In this test case, the crypto-controller of DER 3 sends packets to the crypto-controller of DER 1. We record the number of packets sent (n_s) and received (n_r) for different time periods (T). Table III shows the number of packets between crypto-controller 3 and crypto-controller 1. We use Wireshark to capture packets for different time periods. It can be seen that the packet loss rate ranges from 0 to 5.62%, and PCNMs can work well even when the packet loss rate reaches 5.62%.

We also investigate the impact of packet loss on DEWA. We use a timestamp to generate a switching sequence of the keys. An advantage of using the timestamp is its simplicity, in that no extra information is communicated between different crypto-controllers to realize the dynamic key-encrypted control. In Step 7) of the DEWA algorithm, we consider the error correction process to handle the key mismatch problem in the case where packet loss and short delay occur at the time of key pairs change. Specifically, when a mismatch issue is detected (e.g., an abnormal decrypted value is observed), the private key from the previous updating cycle is utilized. Fig. 7 shows the control signal generated in crypto-controller 1 with the key pairs changing every time period (T). It can be seen that error correction occurs with T = 5 and 10 in the test case, and the DEWA algorithm still works for PCNMs.

2) Comparison of Delays from DEWA and Those Without Encryption: In this case, we compare the delays produced by DEWA-based PCNMs and those without encryption. Specifically, the delays investigated include measurement and compu-

TABLE III
THE NUMBER OF PACKETS BETWEEN CRYPTO-CONTROLLER 3 AND CRYPTO-CONTROLLER 1 FOR DIFFERENT TIME PERIODS

T	n_s	n_r	Packet loss rate
5s	178	168	5.62%
10s	322	322	0
20s	698	680	2.58%
30s	1039	1037	0.19%
50s	1720	1718	0.12%
60s	2095	2078	0.14 %

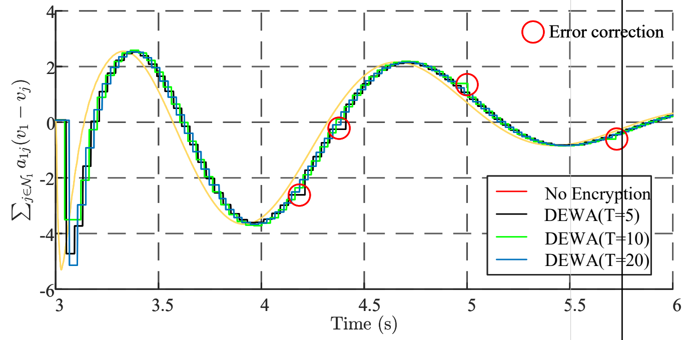

Fig. 7. $\sum_{j\in\mathcal{N}_1} a_{1j}(v_1-v_j)$ with different dynamic key update periods.

tation delays. In DEWA-based PCNMs, each crypto-controller measures the required local physical state variables (i.e., output power, voltage and frequency) through its measurement unit, which introduces measurement delays τ_m. Within each crypto-controller, control signals are encrypted and are sent to its neighbors. When each control signal is received by a neighbor, the signal is decrypted. Both the encryption and decryption processes inevitably introduce delays, i.e., the computation delays τ_c.

To clearly illustrate τ_m and τ_c, we add an incremental number, which starts from zero and increases by 1 every 0.02 seconds. This incremental number is sent from DER 3 with a frequency of 35 Hz, and is received by DER 1. At first, the incremental number is not encrypted. At around time $t = 1.55s$, DEWA is enabled, meaning that the incremental number is first encrypted on the DER 3's side and is later decrypted on the DER 1's side. At around time $t = 1.93s$, DEWA is removed. Fig. 8 illustrates the incremental numbers sent from DER 3 (see the blue line) and those received by DER1 (see the red line) with and without DEWA. From Fig. 8, it can be seen that, the computational delay produced by DEWA is about $\tau_c = 38$ ms, and the measurement delay is about $\tau_m = 6$ ms. Moreover, we measure the measurement and computational delay between controllers 3 and 7 using Beaglebone devices, and Dell servers separately. The experimental results show that the measurement delay is 6.2 and 12 ms, respectively, and the computational delay is 39.9 and 25.1 ms, respectively.

3) Eigenanalysis for DEWA-based PCNMs and NMs Without Encryption: With measured τ_m and τ_c, we further conduct the eigenanalysis for both DEWA-based PCNMs and NMs without encryption. Specifically, the dynamic model of the NMs considering heterogeneous delays can be found in our previous work [30]. As the equations of the small-signal stabil-

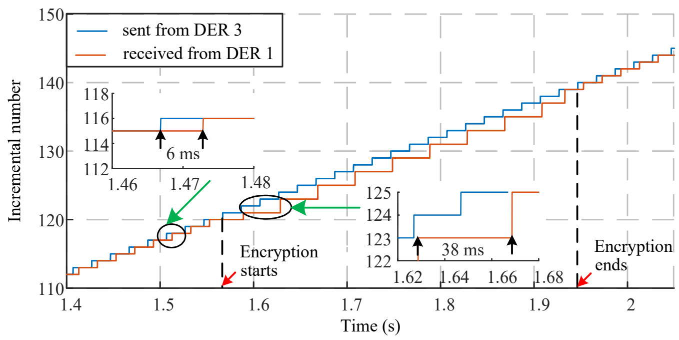

Fig. 8. The incremental numbers sent from DER 2 and those received by DER1 with and without DEWA.

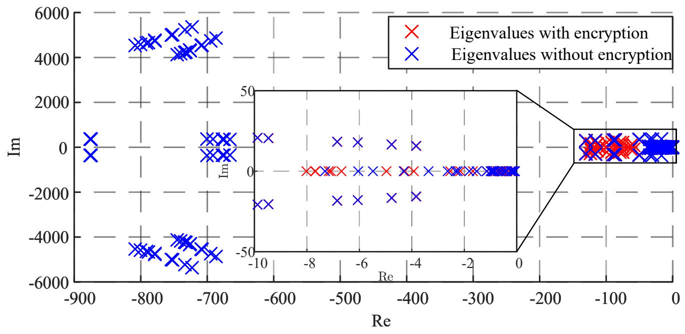

Fig. 9. Eigenvalues of NMs with/ without the encrypted control architecture.

ity model in delayed NMs become transcendental equations, an ODE-SOD [30] method is adopted to calculate the rightmost eigenvalues. Fig. 9 illustrates the eigenvalues of DEWA-based PCNMs and NMs without encryption. It can be seen that the eigenvalues of DEWA-based NMs (see the red points) all have negative real parts (i.e., Re), indicating that, with $\tau_m = 6ms$ and $\tau_c = 38ms$, DEWA-based NMs can maintain the stability. This eigenanalysis result is consistent with the dynamic response in Fig. 5.

D. Privacy Evaluation for DEWA

The effectiveness of DEWA is validated in this subsection. In this case, the system configuration is the same as in Fig. 5. Three types of ciphertexts (i.e., V_1, V_2, and V_3) are recorded. At time $t = 3s$, Load 1 (see Fig. 4) increases from [10 kW, 5 kVAR] to [20 kW, 10 kVAR]. The recorded voltages without DEWA and with DEWA are illustrated in Figs. 10 and 11, respectively. It can be observed that,

- Without DEWA, each voltage eventually converges to a fixed reference value (see Fig. 10). This results in that the voltages can be readily identified.
- With DEWA, each encrypted voltage (i.e., ciphertext) is always varying (see Fig. 11). This makes identifying each voltage more difficult, and thus greatly enhances the privacy-preserving capability.

E. Benefit of SDN-Enabled Crypto-Control

In this subsection, we illustrate a benefit of using SDN in PCNMs, i.e., the dynamic routing function enabled by SDN greatly improves the resilience of PCNMs. Specifically, the

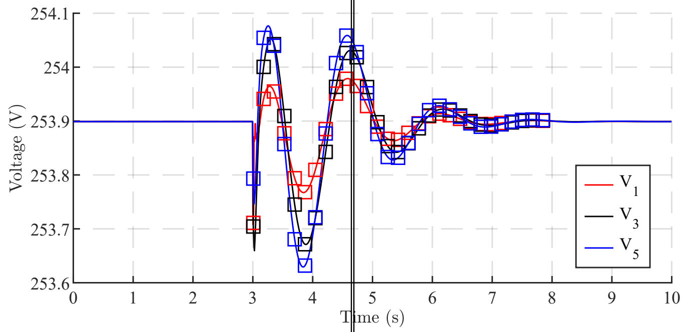

Fig. 10. The recorded voltages without DEWA.

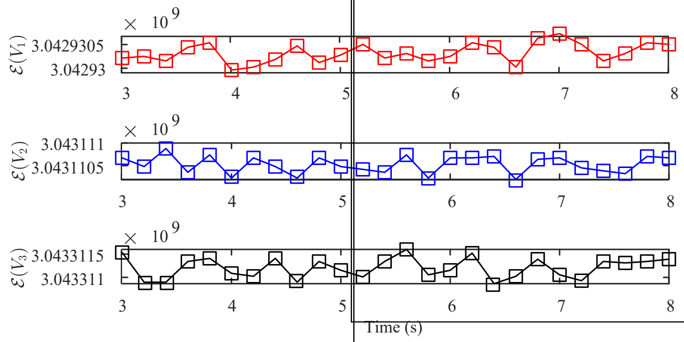

Fig. 11. The encrypted voltages (i.e., ciphertexts) with DEWA.

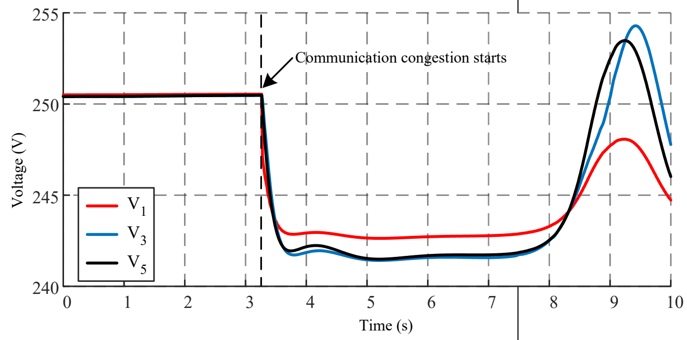

Fig. 12. The responses of V_1, V_3, and V_5 before and after a communication congestion occurs when the dynamic routing function is disabled.

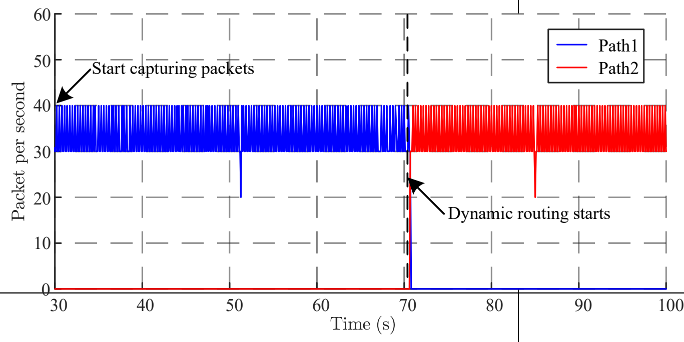

Fig. 13. Data packets in the two paths before and after a communication congestion occurs at around time $t = 70.165s$.

system configuration is the same as in Fig. 5. Two subcases are developed. In the first subcase, at around time $t = 3.25s$, communication congestion occurs in the channel between DER 1 and DER 3. This is achieved by adding a delay of $3s$ in the DEWA algorithm. Fig. 12 illustrates the responses of V_1, V_3, and V_5 before and after the congestion occurs when the dynamic routing function is disabled. It can be seen that the system soon collapses. This is due to the fact that the communication congestion causes the mismatch of public and private keys in the switching key management system.

In the second subcase, we implement the SDN-enabled dynamic routing function in PCNMs. With dynamic routing, when a data path between DER 1 and DER 3 is subjected to congestion, another path is used in real-time. Wireshark is utilized to capture the traffic between DER 1 and DER 3. Fig. 13 illustrates the data packets in the two paths before and after a communication congestion (i.e., the same with that in Fig. 12) occurs at around time $t = 70.165s$. The responses of V_1, V_2, and V_3 with the dynamic routing function are given in Fig. 14. It can be seen from Figs. 12 and 14 that, with SDN, the dynamic routing function has been successfully accomplished and the system performance remains unaffected.

F. Comparison of DEWA with existing scheme

In this subsection, we present the superiority of DEWA-based PCNMs over the traditional differential-privacy method-based NMs and compare DEWA with existing privacy-preserving methods for IoT devices.

1) Comparison with differential-privacy method

The traditional differential-privacy algorithm is adopted from [31]. Its basic idea is that a random noise is added to each data message. This is widely used to preserve privacy in the control area. The system configuration is the same as in Fig. 5, where at around time $t = 3s$, Load 1 increases from [10 kW, 5 kVAR] to [20 kW, 10 kVAR].

The response of V_1 in three different scenarios, i.e., without encryption, with the differential-privacy method, and with DEWA are shown in Fig. 15. It can be observed that,

- The differential-privacy method enhances the privacy during the transient process, as the variation of each parameter becomes larger when random noises are added. However, each varying parameter still eventually converges to a fixed reference value.
- With DEWA, each encrypted voltage is always varying, making identifying each voltage more difficult, and thus greatly enhancing the privacy-preserving capability.

2) Comparison with existing schemes for IoT devices

To show the effectiveness of our proposed scheme, we compare the proposed scheme with three existing encryption schemes for general IoT devices. We choose and compare the DEWA scheme with the Paillier's homomorphic encryption scheme and the ring signcryption scheme [14]. Table IV presents our experimental results for selected privacy-preserving techniques. We compare the homomorphic encryption performance, which is suitable for cooperative control, and measure the time of main operations/phases such as the encryption time ($\mathcal{E}$), the decryption time ($\mathcal{D}$). We give the maximum and minimum time values from 30 iterations. It can be seen that although the Paillier scheme is a partial homomorphic encryption, it cannot be directly applied to a

TABLE IV
MINIMAL AND MAXIMAL COMPUTATION TIMES (IN MS) FOR EVALUATING DEWA AND EXISTING SCHEMES ON THE PCNM PLATFORM

Alg.	PHE	Cooperative Control		C1		C2		C3	
				min	max	min	max	min	max
DEWA	Y	Y	$\mathcal{E}$	17	22	19	32	20	23
			$\mathcal{D}$	16	23	21	30	18	27
Paillier (1024b)	Y	N	$\mathcal{E}$	43	49	39	52	44	53
			$\mathcal{D}$	75	88	80	91	78	90
Paillier (2048b)	Y	N	$\mathcal{E}$	342	368	331	352	342	378
			$\mathcal{D}$	653	686	640	678	662	692
RS/Li [14]	N	N	$\mathcal{E}$	12698	13362	-	-	-	-
			$\mathcal{D}$	19862	20294	-	-	-	-

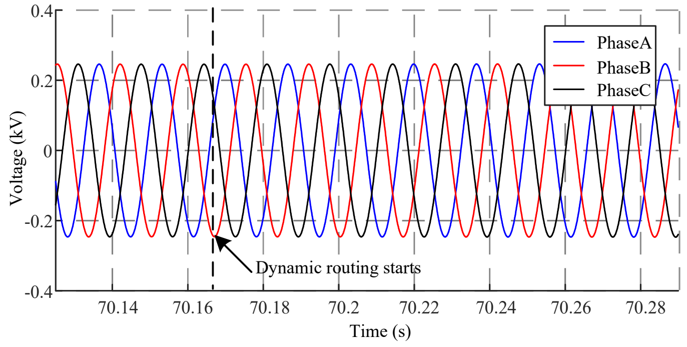

Fig. 14. The responses of V_1, V_2, and V_3 before and after a communication congestion occurs when the dynamic routing function is enabled.

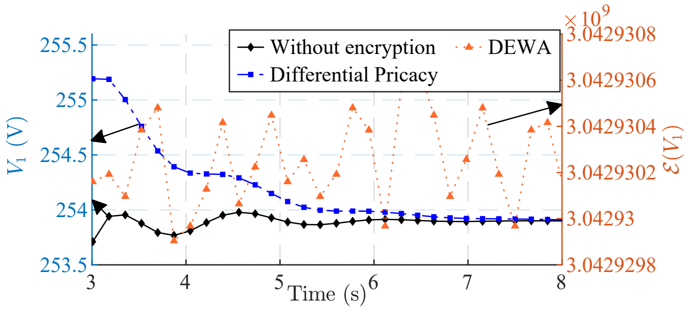

Fig. 15. The response of V_1 in three different scenarios, i.e., without encryption, with the differential-privacy method, and with DEWA.

cooperative control system, and the operation time is longer than DEWA. For the widely used Ring Signcryption scheme, the long operation time makes it not suitable for microgrid control.

VI. CONCLUSION

This paper presents a PCNMs architecture, where crypto-controllers are fully virtualized and SDN is utilized to manage the network. DEWA is developed to preserve each DER's privacy while real-time computation is ensured. The DEWA privacy-preserving property is further mathematically analyzed, and a real-time DEWA-based PCNMs testbed incorporating DEWA, real SDN switches, and IoT devices is established in an RTDS environment. Test results validate the effectiveness, benefits, and superiority of DEWA-based PCNMs. Some future work includes adapting DEWA to other communication-based controls and developing methods to further enhance the privacy-preserving in PCNMs.

REFERENCES

[1] Z. Tang, P. Zhang, and W. O. Krawec, "A quantum leap in microgrids security: The prospects of quantum-secure microgrids," *IEEE Electrification Magazine*, vol. 9, no. 1, pp. 66–73, 2021.

[2] P. Zhang, *Networked Microgrids*. Cambridge, U.K.: Cambridge University Press, 2021.

[3] Z. Tang, Y. Qin, Z. Jiang, W. O. Krawec, and P. Zhang, "Quantum-secure microgrid," *IEEE Transactions on Power Systems*, vol. 36, no. 2, pp. 1250–1263, 2020.

[4] P. Babahajiani, L. Wang, J. Liu, and P. Zhang, "Push-sum-enabled resilient microgrid control," *IEEE Transactions on Smart Grid*, 2021.

[5] L. Wang, Y. Qin, Z. Tang, and P. Zhang, "Software-defined microgrid control: The genesis of decoupled cyber-physical microgrids," *IEEE Open Access Journal of Power and Energy*, vol. 7, pp. 173–182, 2020.

[6] S. Han, U. Topcu, and G. J. Pappas, "Differentially private distributed constrained optimization," *IEEE Transactions on Automatic Control*, vol. 62, no. 1, pp. 50–64, 2016.

[7] J. Cortés, G. E. Dullerud, S. Han, J. Le Ny, S. Mitra, and G. J. Pappas, "Differential privacy in control and network systems," in *2016 IEEE 55th Conference on Decision and Control (CDC)*, pp. 4252–4272, IEEE, 2016.

[8] V. Dvorkin, F. Fioretto, P. Van Hentenryck, P. Pinson, and J. Kazempour, "Differentially private optimal power flow for distribution grids," *IEEE Transactions on Power Systems*, vol. 36, no. 3, pp. 2186–2196, 2020.

[9] F. Fioretto, T. W. K. Mak, and P. Van Hentenryck, "Differential privacy for power grid obfuscation," *IEEE Transactions on Smart Grid*, vol. 11, no. 2, pp. 1356–1366, 2020.

[10] R. Pal, P. Hui, and V. Prasanna, "Privacy engineering for the smart micro-grid," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 5, pp. 965–980, 2018.

[11] B. Fan and X. Wang, "Distributed privacy-preserving active power sharing and frequency regulation in microgrids," *IEEE Transactions on Smart Grid*, vol. 12, no. 4, pp. 3665–3668, 2021.

[12] Z. Wang, K. Yang, and X. Wang, "Privacy-preserving energy scheduling in microgrid systems," *IEEE Transactions on Smart Grid*, vol. 4, no. 4, pp. 1810–1820, 2013.

[13] S. Cirani, G. Ferrari, and L. Veltri, "Enforcing security mechanisms in the ip-based internet of things: An algorithmic overview," *Algorithms*, vol. 6, no. 2, pp. 197–226, 2013.

[14] F. Li, Z. Zheng, and C. Jin, "Secure and efficient data transmission in the internet of things," *Telecommunication Systems*, vol. 62, no. 1, pp. 111–122, 2016.

[15] J. H. Ziegeldorf, O. G. Morchon, and K. Wehrle, "Privacy in the internet of things: threats and challenges," *Security and Communication Networks*, vol. 7, no. 12, pp. 2728–2742, 2014.

[16] T. Wu, C. Zhao, and Y.-J. A. Zhang, "Privacy-preserving distributed optimal power flow with partially homomorphic encryption," *IEEE Transactions on Smart Grid*, 2021.

[17] M. S. Darup, A. Redder, and D. E. Quevedo, "Encrypted cooperative control based on structured feedback," *IEEE control systems letters*, vol. 3, no. 1, pp. 37–42, 2018.

[18] A. B. Alexandru, M. S. Darup, and G. J. Pappas, "Encrypted cooperative control revisited," in *2019 IEEE 58th Conference on Decision and Control (CDC)*, pp. 7196–7202, IEEE, 2019.

[19] C. N. Hadjicostis and A. D. Domínguez-García, “Privacy-preserving distributed averaging via homomorphically encrypted ratio consensus,” *IEEE Transactions on Automatic Control*, vol. 65, no. 9, pp. 3887–3894, 2020.

[20] P. Gope and B. Sikdar, “Lightweight and privacy-preserving two-factor authentication scheme for iot devices,” *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 580–589, 2019.

[21] Z. Tang, P. Zhang, W. O. Krawec, and Z. Jiang, “Programmable quantum networked microgrids,” *IEEE Transactions on Quantum Engineering*, vol. 1, pp. 1–13, 2020.

[22] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, “Software-defined networking: A comprehensive survey,” *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2014.

[23] S. Yang, L. Cui, Z. Chen, and W. Xiao, “An efficient approach to robust sdn controller placement for security,” *IEEE Transactions on Network and Service Management*, vol. 17, no. 3, pp. 1669–1682, 2020.

[24] L. Wang and F. Xiao, “Finite-time consensus problems for networks of dynamic agents,” *IEEE Transactions on Automatic Control*, vol. 55, no. 4, pp. 950–955, 2010.

[25] P. Paillier, “Public-key cryptosystems based on composite degree residuosity classes,” in *International conference on the theory and applications of cryptographic techniques*, pp. 223–238, Springer, 1999.

[26] R. Cramer, I. B. Damgård, and J. B. Nielsen, *Secure Multiparty Computation and Secret Sharing*. Cambridge University Press, 2015.

[27] J. Katz and Y. Lindell, *Introduction to modern cryptography*. CRC press, 2020.

[28] K. Kogiso, “Attack detection and prevention for encrypted control systems by application of switching-key management,” in *2018 IEEE Conference on Decision and Control (CDC)*, pp. 5032–5037, 2018.

[29] S. Farah, Y. Javed, A. Shamim, and T. Nawaz, “An experimental study on performance evaluation of asymmetric encryption algorithms,” in *Recent Advances in Information Science, Proceeding of the 3rd European Conf. of Computer Science,(EECS-12)*, pp. 121–124, 2012.

[30] L. Wang, Y. Zhou, W. Wan, H. Ye, and P. Zhang, “Eigenanalysis of delayed networked microgrids,” *IEEE Transactions on Power Systems*, 2021.

[31] Z. Huang, S. Mitra, and N. Vaidya, “Differentially private distributed optimization,” in *Proceedings of the 2015 International Conference on Distributed Computing and Networking*, pp. 1–10, 2015.