Integrated Distributed Authentication Protocol for Smart Grid Communications

Neetesh Saxena, *Member, IEEE*, and Bong Jun Choi, *Member, IEEE*

***Abstract*—In the smart grid, an integrated distributed authentication protocol is needed to not only securely manage the system but also efficiently authenticate many different entities for the communications. In addition, a lightweight authentication protocol is required to handle frequent authentications among billions of devices. Unfortunately, in the literature, there is no such integrated protocol that provides mutual authentication among the home environment, energy provider, gateways, and advanced metering infrastructure network. Therefore, in this paper, we propose a lightweight cloud-trusted authorities-based integrated (centrally controlled) distributed authentication protocol that provides mutual authentications among communicated entities in a distributed manner. Based on certificateless cryptosystem, our protocol is lightweight and efficient even when there are invalid requests in a batch. Security and performance analysis show that the protocol provides privacy preservation, forward secrecy, semantic security, perfect key ambiguous, and protection against identity thefts while generating lower overheads in comparison with the existing protocols. Also, the protocol is secure against man-in-the-middle attacks, redirection attacks, impersonation attacks, and denial-of-service attacks. Moreover, our protocol provides a complete resistance against flood-based denial-of-service attacks.**

***Index Terms*—Authentication, cloud computing, denial-of-service (*DoS*) attacks, redirection attacks, smart grid (*SG*).**

I. Introduction

THE smart grid (*SG*) is a critical infrastructure whose objective is to provide more efficient, secure, stable, and reliable power to the consumers, operators, and utilities. The *SG* system for home environment consists of various components, such as smart meters (*SM*), home appliances (*HA*), energy providers (*EP*), gateways (*GW*), and advanced metering infrastructure (*AMI*) network. It is generally assumed that the home area network (*HAN*) is wirelessly connected with the Zigbee [1], whereas the building area network (*BAN*)/neighborhood area network (*NAN*) is connected by wide area network and cellular technologies, such as global system for mobile communication (*GSM*) and long term evolution (*LTE*) [2]. *SM*s are equipped with two communication interfaces, where one interface works as a *SM* and other works as a *HAN-GW*. Therefore, the *SM* is a central home controller that communicates with all *HA* within a household. Further, the *BAN-GW*/*NAN-GW* acts as (or deploy) an aggregator (*AG*) that receives data from the *SM* and forwards it to the respective control center (*CC*) via relays and concentrators with wired/wireless connections. Fig. 1 shows the overall architecture of the *SG* system.

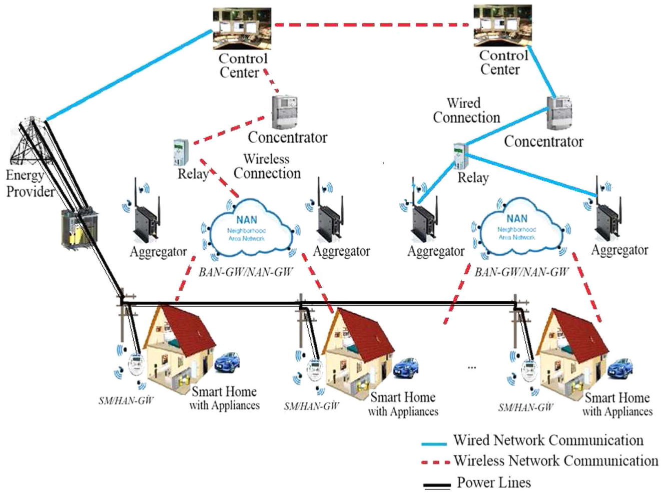


Fig. 1. Overall architecture of the *SG* system.

A. Motivation and Research Problem

Two-way communications in the *SG* enable instant interaction between different *SG* entities and help to improve the overall efficiency of the *SG* system. According to the *NIST* report [3], one of the main security issues in the *SG* system is that existing authentication mechanisms do not sufficiently authenticate devices or exposes authentication keys. Without proper authentications, the system resources and entities can be compromised that may result in financial losses and performance degradation [4]. Centrally control authentication in a decentralized environment is required for the centralized security management in terms of event logging/analysis and authentication [8], [9]. A fast and lightweight protocol is needed to support frequent authentications repeated many times among billions of devices. In sum, an integrated, distributed, fast, and lightweight authentication protocol will provide mutual authentication between the various entities of the *SG* system. An integrated distributed protocol can help to maximize the utilization of shared resources with low overhead. Furthermore, the security protocol of the *SG* system must defend against known security attacks,

Manuscript received January 19, 2016; revised April 30, 2016; accepted May 24, 2016. This work was supported by the Ministry of Science, ICT and Future Planning, Korea, under the "ICT Consilience Creative Program" IITP-2015-R0346-15-1007 supervised by the Institute for Information & Communications Technology Promotion and under the "Basic Science Research Program" 2013R1A1A1010489 and 2015R1C1A1A01053788 through the National Research Foundation.

N. Saxena is with the Department of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA (e-mail: mr.neetesh.saxena@ieee.org).

B. J. Choi is with the Department of Computer Science, State University of New York Korea, Incheon 21985, South Korea, and also with the Department of Computer Science, Stony Brook University, New York, NY 11794 USA (e-mail: bjchoi@sunykorea.ac.kr).

Digital Object Identifier 10.1109/JSYST.2016.2574699

including man-in-the-middle (*MITM*) and denial-of-service (*DoS*) attacks [5].

In the subsequent parts of this section, we first discuss about limitations and concerns of the existing protocols for command and control information delivery. We also highlight the standardized protocols supporting authentication process along with their limitations of not suitable for the *SG* system. Also, we raise a point of user data privacy, which is not covered and maintained by the existing protocols.

There are many different communication protocols used for delivering commands and control information. However, these protocols were not initially designed with security in mind. Today, when Internet is connected to the *SG* system, various organizations, such as *ETSI*, *IEEE*, and *NIST* are embedding security to the existing protocols as new standards in order to prevent the system against well-known security attacks. However, they need to modify many communication standards to make them security embedded. This creates additional overheads. Furthermore, researchers have not yet focused much on an integrated protocol, rather they have proposed separate protocols for individual connections between different entities in the *SG*. They have not discussed the integration of these protocols for compatible communication among them. This motivates us to propose such an efficient and secure authentication protocol for the *SG* system.

There are some standardized protocols available for the *SG* that support authentication process, such as open smart grid protocol (*OSGP*) for the *SM*s, distributed network protocol (*DNP3*) between the *CC* and the substations, device language message specification/companion specification for energy metering (*DLMS*/*COSEM*) for the *AMI* network, and *OpenADR* for the demand response program. In addition, other standardized authentication protocols also exist, such as remote authentication dial-in user service (*RADIUS*) and *Diameter* protocols for the *2G*, *3G*, and *4G* cellular networks [11].

The *OSGP* protocol was deployed for providing the authentication and confidential security to the *SG* applications. This protocol is expected to provide reliable and efficient delivery of command and control information between the *SM*s, direct load control modules, *GW*s, and other *SG* devices. However, recently, researchers from Germany recovered private encryption keys of the *SM*s in a system following *OSGP* without a significant computational effort [12]. Also, a number of attacks has been performed over the *OSGP* protocol [6], including one with just 13 queries to a homegrown message authentication code (*OMA* digest) oracle, and by which the protocol further failed to deliver authenticity guarantee and confidentiality (due to using a nonstandard composition of *RC4* as weak encryption algorithm) [12]. Similar security issues were found in the *DNP3* protocol, which does not provide authentication, message integrity, and confidentiality. In 2012, a new version of the *DNP3* protocol, named *DNP3* secure authentication version 5 was announced, which provides methods to remotely change user update keys using either symmetric or asymmetric cryptography [13]. However, *DNP3* secure authentication considers only spoofing, modification, and replay attacks over the network, and does not provide confidentiality of the message. Also, version 5 of the protocol is not backward compatible with previous versions, which may add a heavy protocol replacement cost.

Furthermore, the authentication provided by *DLMS*/*COSEM*, *OpenADR*, *RADIUS*, and *Diameter* are not sufficient, and also *OpenADR* is costly [11]. The *DLMS* (application layer communication protocol) and *COSEM* (data model) together provide an interface model for metering applications [14]. However, *DLMS*/*COSEM*'s security services are restricted to use symmetric key encryption. In practice, *SM*s need asymmetric key to be used in secure socket layer/transport layer security (*TLS*/*SSL*), but *DLMS*/*COSEM* does not support *TLS*/*SSL*. In demand response, *OpenADR*, which is a standard development effort, supports authentication based on public key cryptography with exchange of certificates [15]. This standard maintains a hierarchy of certified authorities and requires a *PKI* to use three-tier *PKI*, which ultimately results in high development cost.

RADIUS is commonly used protocol to provide centralized remote user authentication and accounting in cellular networks, and *WLAN* interworking and *Wi-Fi* offload situations [16]. However, the *SG* requires decentralized solutions, as a single-point-of-failure can massively affect the centralized system. *RADIUS* implementation supports peer authentication between communication endpoints using a pre-shared key, which brings key management issues and is not suitable for large systems, such as *SG*. Furthermore, *RADIUS* has poor scalability and uses the user datagram protocol (*UDP*), which does not provide reliable data transfer. Therefore, *RADIUS* is not suitable for the *SG* where the availability of information is extremely important. On the other hand, *Diameter* protocol is an authentication, authorization, and accounting protocol used in networking, which supports transmission control protocol (*TCP*) instead of *UDP*. However, its supported capabilities are sometimes more expansive when a large number of entities are involved. Furthermore, *RADIUS* and *Diameter* protocols do not directly protect against *DoS* attacks carried out by flooding the target equipment with bogus traffic.

There are several challenges with the current authentication protocols in terms of efficiency, overhead, cost, delay, and privacy. Also, many vulnerabilities do exist in the available authentication schemes of various communication protocols, such as weak encryption and message digest in the *OSGP* protocol [6], security issues in the *DNP3* protocol [7] (even in version 5 [13]), etc. There is not yet an integrated distributed authentication protocol that provides mutual authentication between the home environment (*HA*, *SM*, *HAN-GW*), *EP*, *GW*s (*BAN-GW*, *NAN-GW*), and the *AMI* network (*SM*, *AG*/collector, *CC*). An integrated protocol can provide a common platform for authenticating various devices while efficiently maximizing the utilization of shared resources with low overhead in the *SG* system. Also, the privacy protection in the *SG* system is an important requirement, so the protocol must not reveal the confidential and private information related to any entity involved in the authentication process. Therefore, an end user (consumer) should have a control over his/her own home environment, such as *HA*, since data generated and being sent belong to a particular user.

Furthermore, the protocol must be fast and efficient, and should be able to defeat known security attacks.

B. Protocol Design Challenges

There are several challenges in designing a centrally controlled integrated distributed authentication protocol as we identified below.

1) The protocol should not only be controlled by a central entity, but also by the subsystems of the *SG* network in a distributed manner.
2) Embedding security solutions in each communication protocol of the *SG* network is not only highly complex but also generates huge overhead and cost. Therefore, it would be more flexible and efficient to instead design a cyber-security layer over the communication network to maintain end-to-end security [10]. This simplifies the integration at the cyber-security layer by avoiding complex integration of different communication protocols in the *SG* network.
3) The protocol should be able to utilize the available system resources efficiently.
4) The generated overhead by the protocol should be as low as possible. The protocol should be fast and lightweight, as authentication process is frequently repeated many times among billions of devices, especially, when devices receive multiple messages at once, such as when *GW*s authenticate multiple *SMs* and gather data from them.
5) The protocol must utilize suitable cryptosystem (with symmetric and/or asymmetric keys) as recommended by standard organizations, such as *IEEE*, *ETSI*, and *NIST*. Particularly, *NIST* report [8] emphasizes the issues of key exchange in symmetric key cryptography and the public key infrastructure (*PKI*) in asymmetric key cryptography. Hence, key management issue must be considered in design.
6) The protocol should enable consumers to have security control over his/her home, i.e., control over all *HA*s with the *SM*.
7) The protocol must support secure communications over the network with strong encryption. Moreover, the identity of each device should be protected over the network to maintain identity anonymity and untraceability.
8) The protocol must be able to defeat various well-known security attacks, such as *MITM* attacks, redirection attacks, impersonation attacks, replay attacks, and flood-based *DoS* attacks.

C. Our Contribution

In this paper, we design an integrated distributed protocol for the *SG* network, which meets all the aforementioned challenges. Note that the proposed protocol may not be suitable for some parts of the *SG* system with very low communication latency requirements, such as for the generic object-oriented substation event (*GOOSE*) and sampled measured values (*SMV*) layer-2 messages within the substation. Here, messages are not encrypted due to the transmission requirements within 4 ms. In such scenarios, a virtual *LAN* with layer-2 capabilities can be used with signed authenticated values [17], or a simple lightweight protocol can be designed for the authentication with integrity. Our new *SG* authentication protocol has the following features.

1) Provides mutual authentication between the *EP* and the *SM*, between the *SM/HAN-GW* and the *BAN-GW/NAN-GW*, between the *SM* and the *HA*, and between the *NAN-GW* and the *CC*.
2) Provides a secure solution for the consumers to easily choose or change the *EP* of their own choice. The protocol also provides more satisfaction to the consumer as he/she will have the control over its *HA* (secured with a password shared between the *SM* and all *HA*, and only he/she can change it).
3) Defeats security attacks: defeats flood-based *DoS* attacks targeting transmitted messages between the *SM/HAN-GW* and the *BAN-GW/NAN-GW*; protects the *SM* and the *EP* from redirection attacks as Zip codes are verified at both ends; preserves the privacy of each message as it is encrypted before being transmitted over the network; provides resistances against *ID* thefts, *MITM* attacks, replay attacks, brute-force attacks, repudiation attacks, and impersonation attacks.
4) Lightweight in terms of communication (*CMO*) and computation overheads (*CPO*). The execution time of 3.96 s can be considered fast, as it is for all the involved entities in the *SG* network and is within the requirements (few minutes) set by the standards [8].
5) Uses cloud-based trusted authorities (*TA*) for key management, which does not have the key exchange or *PKI* issues. Instead, the *TA* generates partial public and private keys, and the legitimate device generates its actual public and private keys.

D. Organization of the Paper

The rest of the paper is organized as follows: Section II discusses related work, and Section III presents our *SG* system model. A new authentication protocol is proposed in Section IV. Security and performance analysis is presented in Section V, including a formal proof of the protocol. Section VI presents the conclusion of this paper.

Table I summarizes different symbols and abbreviations used in the paper along with their descriptions and sizes. Note that the sizes of public and private keys depend on the algorithm used in asymmetric encryption.

II. Related Work

We first discuss the existing authentication protocols that provide authentications between various entities with lower overhead, and then those that provide protection against security attacks and preserves the privacy over the *SG* network.

For providing low overheads, a lightweight authentication scheme based on the Diffie–Hellman key exchange protocol and a hash-based message authentication code (*HMAC*) was proposed in [1]. However, it provides mutual authentication

TABLE I
Symbols and Abbreviations

Symbol	Description	Size (bits)
H_1 ()/H_2 ()	Hash functions used in ciphering	—
H_3 ()	Hash function for *SK* key generation	—
$H_{3\,\text{change}}$ ()	Hash function for changing the password	—
h()	Hash function for computing *e*	—
ID	Identity of the entity	128
e	Hash value	128
MAC	Message authentication code	64
PUK	Public key	160
PRK	Private key	160
SK	Shared secret key	256
T	Timestamp	64
K	Random number	128
Zip	Postal code	128
S	Signature	128
pwd	Password shared between *SM* and *HA*s	128
Z	Sum of products of *K* and *ID*	128
P	Sum of products of *PRK* and *ID*	128
R	Sum of products of *S* and *ID*	128

only between the *HAN-GW* and the *BAN-GW*. Sule *et al.* [18] made a change in [1] by using an *MAC* between the *AMI* devices and the controller nodes instead of *HMAC*. Although this scheme reduces the verification time, it also reduces the protocol security provided by the function. As in [1], the scheme only involves the *HAN-GW* and the *BAN-GW* communication. Further, an authentication scheme using a batch signature verification was proposed in [19]. However, the scheme does not focus on authentication among *SM*, *HAN*, and *HA*, rather authenticating data aggregation. A key agreement protocol for the *SG* is proposed in [20], which reduces the number of hash functions used and the delay caused by the security process. Recently, an identity-based scheme is proposed to provide authentication between the *SM* and the *AS*, and reduces the total number of exchanged packets, but increases the *CPO* [21].

Many researchers have proposed solutions in order to resist against different attacks in the *SG* system, such as replay, *MITM*, impersonation, and *DoS*. However, in the absence of authentication, an attacker can easily tamper the message and/or can send a fabricated message. In this direction, a mutual authentication scheme between the *SM* and the data concentration unit (*DCU*) was proposed to prevent impersonation and *MITM* attacks [22]. However, this scheme neither discusses the generated overhead nor provides authentication in a home environment. Recently, an authentication scheme using a Merkle hash tree technique was proposed to prevent replay, injection, and message modification attacks [23]. However, communication only between the *HAN* and the *NAN* is considered. A Diffie–Hellman-based secure aggregation scheme for collecting data was presented in [24], which generates lower *CPO* and *CMO*, but the scheme does not consider *SM*'s authentication. Kursawe *et al.* [25] stated that a strong authentication technique is required for all users and devices within the *SG* network. It is expected that in the near future, due to the increase in the number of devices, the current protocols may not be scalable.

In addition, the privacy of the customers in terms of power usage, billing, and other information must be preserved during the authentication. In this direction, an identity-based authentication protocol is proposed to provide source authentication, data integrity, nonrepudiation services, and privacy preservation in *AMI* [26]. However, the protocol does not consider overhead and efficiency. Yan *et al.* [27] proposed an integrated authentication and confidentiality protocol that provides a mutual authentication between the *SM* and the *AMI* network, and enables data privacy, integrity, and confidentiality. However, the protocol generates a large overhead as it performs several encryption/decryption operations. Further, it does not consider *EP* and *HA* entities in the authentication system.

In summary, several standard, lightweight, and privacy-preserved protocols have been proposed by researchers. However, the existing standard protocols do not provide sufficient security and privacy preservation to the *SG* system. Also, many existing protocols (including privacy-preserved) are inefficient and generate large overheads. Furthermore, the existing lightweight and privacy-preserved protocols are with limited capability of authenticating only few entities (mostly two devices) in the *SG*. In other words, these protocols do not enable authentication among all entities with optimized resource utilization. Moreover, embedding security to the existing protocols generates large overheads and requires integration to authenticate all entities of the *SG* network, which results in inefficient and costly solutions. Therefore, there is a need of an integrated lightweight authentication protocol that provides mutual authentication from end-to-end, protects the *SG* system from known attacks, and keeps the privacy preserved. We tackle this problem in this paper.

III. System Model

In the *SG* system, security operations are usually assumed to be done independently by individual center. However, due to limited processing capability, these centers do not support online analysis and generate high maintenance cost [28]. Further, the *SG* requires a powerful platform with effective integration and ubiquitous seamless access to collect and analyze large data collected from a variety of sources, such as *AMI*, wide area measurement system (*WAMS*), and *HA*. Recent studies [29]–[32] show that cloud computing is very much compatible with the *SG* system because of its several advantages, including energy efficiency, flexibility, scalability, agility, and cost effectiveness. Various researchers have proposed their solutions by integrating cloud computing in the *SG* system. Baek *et al.* [30] designed a big data information management framework, called Smart-Frame, based on a cloud computing model. Also, Jiang *et al.* [33] proposed a scheme for searchable encryption on the cloud database in the *SG*, and Bitzer and Gebretsadik [34] presented a feasibility study of monitoring renewable energy in the *SG* based on a cloud computing framework retaining *SG* security. Developing a secure cloud network is not our goal in this paper. However, we consider that our scheme uses secure cloud servers as discussed in [30], [33], and [34]. We employ the cloud computing into our *SG* system, particularly [30], which builds a hierarchical structure of cloud computing centers. Employing cloud computing in the *SG* not only addresses the issue of large information management, but also provides a high energy and cost saving platform.

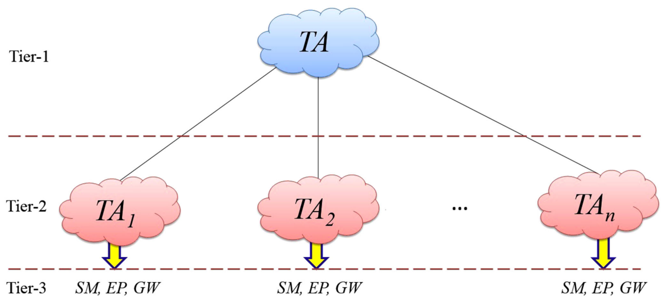


Fig. 2. Hierarchy of *TA*s.

A roadmap in [35] presents a realistic example of deploying cloud computing centers in the *SG* system.

We propose to have a cyber-security layer on the top of communication layer that takes care of the security issues existing in the communication between any two entities over the network. Our *SG* system is divided into several regions/areas, each of which is managed by either a public or private, but secure cloud computing center [30]. As shown in Fig. 2, we consider three different tiers in our *SG* system as follows:

1) Tier-1: Central cloud computing center;
2) Tier-2: Distributed cloud computing centers;
3) Tier-3: *SM*s, *GW*s, and *EP*s.

As shown in Fig. 2, there are *n* distributed cloud computing centers, also called trusted authorities (*TA*s). Each *TA* manages a region that includes various *SM*s, *GW*s, and *EP*s. The tier-1 *TA* provides inter-*TA* communication among different entities within the system, while the tier-2 *TA*s are responsible for managing the public key repository, and generating partial public and private keys of devices at their ends. The main purpose of enabling cloud environment in our *SG* system is to provide an easy and fast access to the public key repository and to efficiently generate public and private key pairs. In addition, the *SG* requires a powerful computing platform to handle a large-scale data analysis and to support complex real-time application services. In each *TA*, various cloud computing services can be deployed, such as infrastructure-as-a-service for *SG* information collection, processing and storage, platform-as-a-service for developing and integrating cloud computing specific security-based applications for the *SG* environment, and software-as-a-service for specific services, such as optimization of energy usage.

IV. Proposed Authentication Protocol

This section proposes an authentication protocol for the *SG* system. We first present an overview of our protocol, then present mutual authentication approaches between different *SG* entities. The authentication between *EP-SM*, *SM-GW*, and *SM-HA* are based on asymmetric key cryptography, asymmetric key cryptography in batch, and symmetric key cryptography, respectively.

A. Overview

Recently, identity-based cryptography (*IBC*) is considered suitable for securing grid and cloud computing environments [36], [37]. However, *IBC* suffers from the key escrow problem [38]. Our protocol is based on a certificateless cryptosystem, which is a combination of identity-based cryptography and traditional public key cryptography [39]. Our approach not only overcomes the key escrow problem in *IBC*, but also does not require traditional *PKI* that is costly due to the private key generation. We instead use a key generation center (*KGC*). The security of our scheme is based on the security of elliptic curve discrete logarithm problem (*ECDLP*) for the group of points over the finite field. Here, we let *E* be an elliptic curve defined over a finite field F_p as $E : y^2 = x^3 + Ax + B; A, B \in F_p$. Let E_1 and E_2 be points in $E(F_p)$ and integer x is found such that $E_1 = xE_2$. We do not design a pairing based scheme under *ECC*, but design a certificateless-based asymmetric encryption scheme. This is because a multiplication of points under *ECC* is more efficient than a pairing operation. For instance, it takes 0.6 ms for a point multiplication and 4.5 ms for a pairing operation under the same setting [40]. The identity (*ID*) of each device (*EP*, *SM*, *GW*, *HA*) in the *SG* network is taken from a random point on elliptic curve over $E(F_q)$.

Each *TA* generates its private and public key pair, known as a master private key and a master public key, and makes the public key available to its users. Our approach is simpler than the Diffie–Hellman protocol, as it uses one-way hash functions instead of exponential functions. The *KGC* (at each *TA*) supplies an entity with a partial private key (*PPR*) and a partial public key (*PPU*). We assume that *KGC* securely delivers the partial keys to the intended entities. Each entity then combines its partial public and private keys with secret information to generate its actual private and public keys. In this way, the entity's private key is not known to the *KGC* and the anonymity of the user's public key is also achieved. This anonymity is useful when we consider that in order to receive the public key of a device, the requested device must be verified authentic to the *TA* using its partial key credentials.

First, we present generic definitions of the algorithms used in our scheme, and then explain each of these algorithms in detail.

Definition 1: A generic certificateless public key encryption scheme consists of the following algorithms.

1) *Setup:* The *KGC* generates a common public parameter (*param*) and a master secret key (*masterKey*), and uses these keys to generate different keys.
2) *PartialKeyGeneration:* *TA* uses *param*, *masterKey*, and an identity *ID* (a point of elliptic curve group) received from a user to generate a *PPR* and a *PPU* as (*PPU*, *PPR*) = PartialKeyExtract (*param*, *masterKey*, *ID*).
3) *SecretValue:* Each user/device generates a unique secret value *SID* using a random number *rand* as *SID* = SecretValue(*rand*, *ID*).
4) *GenPrivateKey:* User/device uses *param*, *PPR*, and *SID* to generate private key *PRK* as *PRK* = GenPrivateKey(*param*, *PPR*, *SID*).
5) *GenPublicKey:* User/device uses *param*, *PPU*, *SID*, and *ID* to generate public key *PUK* as *PUK* = GenPublicKey(*param*, *PPU*, *SID*, *ID*).
6) *Encrypt:* The plaintext M is encrypted using *param* and *PUK* to generate a ciphertext C as C = Encrypt(*param*, *PUK*, M).

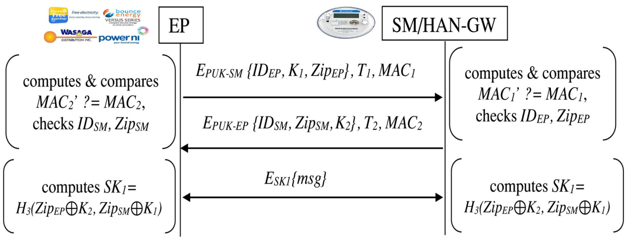

EP
SM/HAN-GW
$E_{PUK\text{-}SM}\{ID_{EP}, K_1, Zip_{EP}\}, T_1, MAC_1$
$E_{PUK\text{-}EP}\{ID_{SM}, Zip_{SM}, K_2\}, T_2, MAC_2$
$E_{SK1}\{msg\}$
computes & compares MAC_2' ?= MAC_2, checks ID_{SM}, Zip_{SM}
computes & compares MAC_1' ?= MAC_1, checks ID_{EP}, Zip_{EP}
computes $SK_1 = H_3(Zip_{EP} \oplus K_2, Zip_{SM} \oplus K_1)$
computes $SK_1 = H_3(Zip_{EP} \oplus K_2, Zip_{SM} \oplus K_1)$

Fig. 3. Authentication between the *EP* and the *SM*.

7) *Decrypt:* The ciphertext C is decrypted using *param* and *PRK* to retrieve the plaintext M as $M =$ Decrypt(*param*, *PRK*, C).

The public key of each entity is available in a public repository of the corresponding tier-2 cloud computing center (*TA*). The private keys are kept secret and stored on the *SM*s, the *GW*s, and the *EP*s. Since each entity is registered to a specific *TA*, it knows the identity and the public key of the *TA*. The details of generating different keys are as follows.

1) *Setup:* $t \xleftarrow{r} \mathbb{Z}_q^*$ is a random integer with large prime q, and P is a generator of a large cyclic group G over $E(F_q)$. Each *TA* generates its private and public key pair as (*PRK*$_{\rm TA} = t$, *PUK*$_{\rm TA} = tP$) . Let us define the hash functions used in this protocol as $H_1 : \mathbb{Z}_q^* \rightarrow \{0,1\}^*$, $H_2 : \{0,1\}^* \rightarrow \{0,1\}^*$ and $H_3 : \mathbb{Z}_q^* \times \mathbb{Z}_q^* \rightarrow \{0,1\}^*$. Returns *param* $= (q, P,$ *PUK*$_{\rm TA}, H_1, H_2, H_3)$ and *masterKey* $=$ $(q, P,$ *PRK*$_{\rm TA}, H_1, H_2, H_3)$.
2) *PartialKeyGeneration:* *TA* chooses a random $s \in \mathbb{Z}_q^*$, and computes $w = sP$ and $x = s +$ *PRK*$_{\rm TA}\, H_1(ID)$. Note that *ID* is first converted from an elliptic curve point to a bit string [41] in $H_1()$ and then is hashed. Returns (*PPU*, *PPR*) $= (w, x)$.
3) *SecretValue:* Each device generates a unique $z \in \mathbb{Z}_q^*$ using SecretValue() function. Returns *SID* $= z$.
4) *GenPrivateKey:* Each device computes its private key *PRK* $= (z, x)$. Returns *PRK*.
5) *GenPublicKey:* Each device computes its public key *PUK* $= (w, v)$, where $v = zP$. Returns *PUK*.
6) *Encrypt:* Sender device computes $r = H_2(M||\gamma)$, where $M \in \{0,1\}^*$ is a plaintext and $\gamma \in \{0,1\}^*$. Furthermore, it computes ciphertext $C = (c_1, c_2, c_3)$ such that $c_1 = rP$; $c_2 = rv + M||\gamma$; $c_3 = w + u$; where $u =$ *PUK*$_{\rm TA}\, H_1(ID)$. Returns C.
7) *Decrypt:* Receiver device first applies *PPR* by computing *Ver*$_1 = c_3 - xP$. If *Ver*$_1 = 0$, it proceeds further, otherwise terminates the connection. Thereafter, the device retrieves the message $M||\gamma$ as $c_2 - zc_1$ and verifies *Ver*$_2$ as $H_2(M||\gamma)P \overset{?}{=} c_1$. Returns M.

B. Authentication Between the EP and the SM

We assume that *EP* knows the identity of each *SM* that it supplies the electricity to. Similarly, each *SM* also knows the identity of its *EP*, as it has a contract with the *EP*. As shown in Fig. 3, the authentication between the *EP* and the *SM/HAN-GW* is carried out as follows:

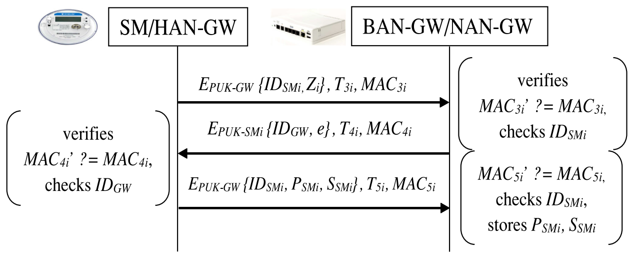

SM/HAN-GW
BAN-GW/NAN-GW
$E_{PUK\text{-}GW}\{ID_{SMi}, Z_i\}, T_{3i}, MAC_{3i}$
$E_{PUK\text{-}SMi}\{ID_{GW}, e\}, T_{4i}, MAC_{4i}$
$E_{PUK\text{-}GW}\{ID_{SMi}, P_{SMi}, S_{SMi}\}, T_{5i}, MAC_{5i}$
verifies MAC_{4i}' ?= MAC_{4i}, checks ID_{GW}
verifies MAC_{3i}' ?= MAC_{3i}, checks ID_{SMi}
MAC_{5i}' ?= MAC_{5i}, checks ID_{SMi}, stores P_{SMi}, S_{SMi}

Fig. 4. Authentication between the *SM* and the *BAN/NAN-GW*.

Step-1 EP→SM: $[E_{\rm PUK_{SM}}\{ID_{\rm EP}, K_1, Zip_{\rm EP}\}, T_1, MAC_1]$: First, the *EP* retrieves the public key of the *SM* from the repository stored at its tier-2, *i.e.*, *PUK*$_{\rm SM}$. Then, the *EP* encrypts its identity *ID*$_{\rm EP}$, a nonce K_1, and the location (Zip code) *Zip*$_{\rm EP}$ with the public key of the *SM* and sends it to the *SM* along with a current timestamp T_1 and an *MAC*$_1$ (message-1), where *MAC*$_1$ $= [E_{\rm PUK_{SM}}\{ID_{\rm EP}, K_1, Zip_{\rm EP}\}, T_1]$. We consider each *MAC* as a *HMAC* function, *i.e.*, *HMACSHA256*, that uses a pre-assigned key, say *K*.

Step-2 SM→EP: $[E_{\rm PUK_{EP}}\{ID_{\rm SM}, K_2, Zip_{\rm SM}\}, T_2,$ *MAC*$_2]$: On receiving message-1, the *SM* computes *MAC*$_1'$ and checks if *MAC*$_1 \overset{?}{=}$*MAC*$_1'$. If it is verified, the *SM* decrypts the message using its private key. Then, the *SM* retrieves the public key of the *EP* (*PUK*$_{\rm EP}$) and verifies the identity and the location of the *EP*. If it is verified, the *SM* sends (*ID*$_{\rm SM}$, K_2, *Zip*$_{\rm SM}$) encrypted with *PUK*$_{\rm EP}$ to the *EP* along with T_2 and *MAC*$_2$ (message-2), where *MAC*$_2$ $= [E_{\rm PUK_{EP}}\{ID_{\rm SM}, K_2, Zip_{\rm SM}\}, T_2]$.

Step-3: On receiving message-2, the *EP* computes *MAC*$_2'$ and checks if *MAC*$_2 \overset{?}{=}$*MAC*$_2'$. If it is verified, the *EP* decrypts the received message using its private key, and verifies the identity and the location of the *SM*. If both are correct, the *EP* computes a shared secret key as *SK*$_1$ $= H_3(Zip_{\rm EP} \oplus K_2, Zip_{\rm SM} \oplus K_1)$ and sends message to the *SM* encrypted with this shared key. Here, $H_3()$ is a one-way hash function. Similarly, the *SM* also computes the same secret *SK*$_1$ key.

C. Authentication Between the SM and the GW

We assume that a group of *SM*s sends its metering data to a specific *GW*. The *GW* keeps a record of the identity of each *SM* associated with it. A number of *SM*s communicates with a *GW* simultaneously, so the authentication process is executed in a batch. The authentication process and the communication scenario of the proposed authentication scheme between a group of *SM*s and the *GW* are shown in Figs. 4 and 5, respectively. As shown in Fig. 4, the authentication process is carried out as follows:

Step-1 SM$_i$→GW: $[E_{\rm PUK_{GW}}\{ID_{{\rm SM}_i}, Z_i\}, T_{3_i}, MAC_{3_i}]$: First, each *SM*$_i$ retrieves the identity and the public key of the *GW*. Then, each *SM*$_i$ sends its identity and Z_i encrypted with *PUK*$_{GW}$ along with its current timestamp T_{3_i} and *MAC*$_{3_i}$ to the *GW* (message-1), where *MAC*$_{3_i}$ $= [E_{\rm PUK_{GW}}\{ID_{{\rm SM}_i}, Z_i\}, T_{3_i}]$ and $Z_i =$ *ID*$_{{\rm SM}_i} K_i$. The $K_i \in [1, q$-$1]$ are the random secret values selected by each *SM*$_i$.

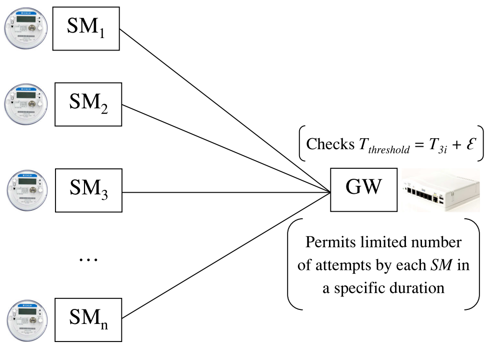

SM1
SM2
SM3
…
SMn
GW
Checks $T_{threshold} = T_{3i} + \mathcal{E}$
Permits limited number of attempts by each SM in a specific duration

Fig. 5. Communication scenario between a group of *SM*s and the *GW*.

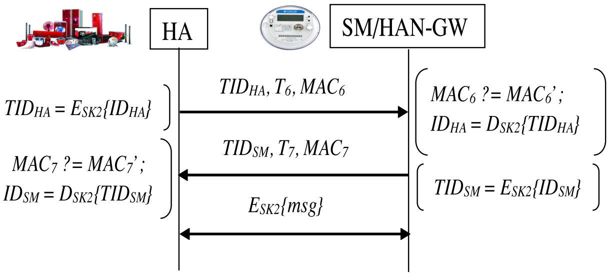

HA
SM/HAN-GW
$TID_{HA} = E_{SK2}\{ID_{HA}\}$
TID_{HA}, T_6, MAC_6
MAC_6 ?= MAC_6';
$ID_{HA} = D_{SK2}\{TID_{HA}\}$
MAC_7 ?= MAC_7';
$ID_{SM} = D_{SK2}\{TID_{SM}\}$
TID_{SM}, T_7, MAC_7
$TID_{SM} = E_{SK2}\{ID_{SM}\}$
$E_{SK2}\{msg\}$

Fig. 6. Authentication between the *HA* and the *SM*.

Step-2 GW→SM$_i$: $[E_{\mathrm{PUK}_{\mathrm{SM_i}}}\{ID_{GW}, e\}, T_{4_i}, MAC_{4_i}]$: On receiving message-1, the *GW* computes *MAC*$'_{3_i}$ and checks message integrity. If it is verified, the *GW* compares its current timestamp t_m with $T_{\text{threshold}} = T_{3_i} + \epsilon$, where ϵ is the maximum allowed delay to transmit the message to the *GW*. If $t_m > T_{\text{threshold}}$, the request is discarded and the connection is terminated. Otherwise, the *GW* decrypts the message using its private key, and verifies the identity of the *SM$_i$*. If it is verified, the *GW* checks the number of attempts by the *SM$_i$* within a specified interval. If it is more than the assigned limit, the connection is terminated. Otherwise, the *GW* sends its identity and a value e encrypted using the public key of the corresponding *SM$_i$* along with T_{4_i} and *MAC*$_{4_i}$ (message-2) to the *SM$_i$*. Here, $e = h(Z)$, h is a one-way hash function, $Z = \sum_{i=1}^{n} Z_i$, and n is the number of *SM$_i$* communicating with the *GW*.

Step-3 SM$_i$→GW: $[E_{\mathrm{PUK}_{\mathrm{GW}}}\{ID_{\mathrm{SM}_i}, P_{\mathrm{SM}_i}, S_{\mathrm{SM}_i}\}, T_{5_i}, MAC_{5_i}]$: On receiving message-2, each *SM$_i$* computes *MAC*$'_{4_i}$ and verifies the integrity of each message. If it is verified, *SM$_i$* decrypts the messages using private keys *PRK*$_{\mathrm{SM}_i}$, and verifies the received identity of the *GW*. If it is verified, each *SM$_i$* stores e, and generates a variable $P_{\mathrm{SM}_i} =$ *PRK*$_{\mathrm{SM}_i}$*ID*$_{\mathrm{SM}_i}$ and a signature $S_{\mathrm{SM}_i} = (K_i + e\mathit{PRK}_{\mathrm{SM}_i}) \bmod n$. Note that the first 128 bits of $P_{\mathrm{PRK}_{\mathrm{SM}_i}}$ are used in P_{SM_i} and S_{SM_i} for operations' compatibility. Then, each *SM$_i$* sends *ID*$_{\mathrm{SM}_i}$, P_{SM_i}, and S_{SM_i} encrypted using public key of the *GW* along with T_{5_i} and *MAC*$_{5_i} = [E_{\mathrm{PUK}_{\mathrm{GW}}}\{ID_{\mathrm{SM}_i}, P_{\mathrm{SM}_i}, S_{\mathrm{SM}_i}\}, T_{5_i}]$ (message-3) to the *GW*. On receiving message-3, the *GW* computes *MAC*$'_{5_i}$ and checks message integrity. If it is verified, the *GW* decrypts the messages, and verifies the identity of each *SM$_i$*. In a scenario where a group of *SM$_i$* communicates with a *GW*, adversary may possibly compromise some of the *SM$_i$* to perform flood-based *DoS* attacks. The compromised *SM$_i$* can flood the victim *GW* with fake message-3 by spoofing meters' identities. Adversary can even send an empty or a random message to the *GW*. This leads to half-open authentication requests at the *GW*. Step-2 of this protocol addresses such issues. In order to prevent these attacks, the identity and signature of each *SM$_i$* is verified. For each unresponsive *SM$_i$*, the *GW* removes the corresponding Z_i and re-computes Z. Then, the *GW* computes $P = \sum_{i=1}^{n} P_{\mathrm{SM}_i}$ and $R = \sum_{i=1}^{n} S_{\mathrm{SM}_i} ID_{\mathrm{SM}_i}$, and verifies ($R - eP \overset{?}{=} Z$). Therefore, our scheme is efficient even with the presence of invalid requests in a batch since the *GW* only needs to recompute Z, which is simply a summation of all Z_i.

D. Authentication Between the HA and the SM

Since data generated and sent by all *HA*s belong to a particular user, we involve the end user (owner) for authenticating the *HA*s (at the initial setup) [42]. The energy consumption information can reveal personal details of the consumers, such as their daily routines (including times when they are at home or asleep), what electronic equipment they own and are being used, etc. Consumers expect that the privacy of this information is maintained. We assume that the *SM* and all *HA*s share a password selected by the user. A secret key $SK_2 = H_3(pwd, T)$ is generated each time a *HA* and the *SM* communicates, where pwd is the shared password, T is a timestamp, and H_3 is a one-way hash function. As shown in Fig. 6, the authentication process between the *HA* and the *SM* is carried out as follows:

Step-1 HA→SM: $[TID_{\mathrm{HA}}, T_6, MAC_6]$: First, each *HA* generates SK_2 from a shared password and uses it to encrypt the original identity of the *HA*. Then, it sends a temporary identity *TID*$_{\mathrm{HA}}$, a timestamp T_6, and *MAC*$_6$ to the *SM* (message-1), where *MAC*$_6 = [TID_{\mathrm{HA}}, T_6]$ and $SK_2 = H_3(pwd, T_6)$. The encryption can be performed by any standard symmetric key algorithm, such as *AES-CTR* or *MAES-CTR* [43].

Step-2 SM→HA: $[TID_{\mathrm{SM}}, T_7, MAC_7]$: On receiving message-1, the *SM* verifies *MAC*$'_6$ with the received *MAC*$_6$. If it is verified, the *SM* decrypts and recovers the actual identity of the *HA*. If the identity belongs to one of its *HA*, it generates a temporary identity *TID*$_{\mathrm{SM}}$ and sends its identity to the *HA* along with T_7 and *MAC*$_7$ (message-2), where *MAC*$_7 = [TID_{\mathrm{SM}}, T_7]$.

On receiving message-2, the *HA* computes *MAC*$'_7$ and compares it with *MAC*$_7$, and further decrypts and recovers the actual identity of the *SM*. If it is correct, the *HA* and the *SM* can start communicating using messages encrypted by SK_2. Moreover, the password can be automatically changed at a regular interval by calculating $pwd_{i+1} = N \times H_{3_{\text{change}}}(d \times pwd_i)$, where N is the number of days, d is a random secret, and $H_{3_{\text{change}}}()$ is a hash function. For the password change, the user needs to provide N to the *SM*. When, a new password is generated at *SM*, the *SM* encrypts the password using last session key and sends it to all the *HAS* before discarding the previous key.

E. Authentication Between the NAN-GW and the CC

We assume that the *NAN-GW* aggregates the received data from different *SM*s. The *CC* is assumed to be connected to the *NAN-GW* using wired network and is authenticated. In case, if it is wireless connected, the scenario similar to *EP-SM* provides mutual authentication.

V. Security and Performance Analysis

This section presents the verification proofs, defenses against security attacks, and security and performance analysis of our protocol in comparison with the existing lightweight protocols.

A. Verification Proof

We present the verification proofs for public decryption of our public encryption scheme, and the correctness of the protocol between *SM*s and their corresponding *GW*.

1) Verification of Decryption in Our Encryption Scheme:

$$
\begin{aligned}
Ver_1 &= c_3 - xP \\
&= w + u - xP \\
&= sP + PUK_{\mathrm{TA}} H_1(ID) - [s + PRK_{\mathrm{TA}} H_1(ID)]P \\
&= sP + PRK_{\mathrm{TA}} H_1(ID)P - sP - PRK_{\mathrm{TA}} H_1(ID)P \\
&= 0. \\
Ver_2 &= H_2(M||\gamma)P \overset{?}{=} c_1 \\
&= H_2(M||\gamma)P \overset{?}{=} rP \\
&= H_2(M||\gamma)P \overset{?}{=} H_2(M||\gamma)P.
\end{aligned}
$$

2) Correctness of the Protocol Between SM$_i$-GW:

$$
\begin{aligned}
\text{L.H.S.} &= Z = \sum_{i=1}^{n} Z_i \\
&= ID_{\mathrm{SM}_1} K_1 + ID_{\mathrm{SM}_2} K_2 + \cdots + ID_{\mathrm{SM}_n} K_n. \\
\text{R.H.S.} &= R - eP \\
&= (S_{\mathrm{SM}_1} ID_{\mathrm{SM}_1} + S_{\mathrm{SM}_2} ID_{\mathrm{SM}_2} + \cdots + S_{\mathrm{SM}_n} \\
&\quad ID_{\mathrm{SM}_n}) - e(P_{\mathrm{SM}_1} + P_{\mathrm{SM}_2} + \cdots + P_{\mathrm{SM}_n}) \\
&= ((K_1 + e(PRK_{\mathrm{SM}_1}))ID_{\mathrm{SM}_1} + (K_2 + e \\
&\quad (PRK_{\mathrm{SM}_2}))ID_{\mathrm{SM}_2} + \cdots + (K_n + e(PRK_{\mathrm{SM}_n})) \\
&\quad ID_{\mathrm{SM}_n}) - e(P_{\mathrm{SM}_1} + P_{\mathrm{SM}_2} + \cdots + P_{\mathrm{SM}_n}) \\
&= (ID_{\mathrm{SM}_1} K_1 + ID_{\mathrm{SM}_2} K_2 + \cdots + ID_{\mathrm{SM}_n} K_n) + e \\
&\quad ((PRK_{\mathrm{SM}_1})ID_{\mathrm{SM}_1} + (PRK_{\mathrm{SM}_2})ID_{\mathrm{SM}_2} \\
&\quad + \cdots + (PRK_{\mathrm{SM}_n})ID_{\mathrm{SM}_n}) - e(P_{\mathrm{SM}_1} \\
&\quad + P_{\mathrm{SM}_2} + \cdots + P_{\mathrm{SM}_n}) \\
&= (ID_{\mathrm{SM}_1} K_1 + ID_{\mathrm{SM}_2} K_2 + \cdots + ID_{\mathrm{SM}_n} K_n) + e \\
&\quad (P_{\mathrm{SM}_1} + P_{\mathrm{SM}_2} + \cdots + P_{\mathrm{SM}_n}) - e(P_{\mathrm{SM}_1} + \\
&\quad P_{\mathrm{SM}_2} + \cdots + P_{\mathrm{SM}_n}) \\
&= ID_{\mathrm{SM}_1} K_1 + ID_{\mathrm{SM}_2} K_2 + \cdots + ID_{\mathrm{SM}_n} K_n = Z.
\end{aligned}
$$

TABLE II
Comparison of Security Capabilities

Vulnerabilities	[20]	[1]	[21]	Proposed
MITM attacks	Yes	Yes	Yes	Yes
Replay attacks	Yes	Yes	Yes	Yes
Impersonation attacks	Yes	Yes	Yes	Yes
Brute-force attacks	Yes	Yes	Yes	Yes
Redirection attacks	No	No	No	Yes
Flood-based *DoS* attacks	Partial	No	Partial	Yes

TABLE III
Comparison of Security Requirements Fulfilled

Requirements	[20]	[1]	[21]	Proposed
Mutual authentication	Yes	Yes	Yes	Yes
Forward secrecy	Yes	Yes	Yes	Yes
Privacy preservation	No	Yes	No	Yes
Prevents *ID* thefts	No	Yes	No	Yes

B. Defenses Against Security Attacks

We assume that an adversary $\mathcal{A}$ has a complete knowledge about the system topology, as well as the identities and public keys of the entities. $\mathcal{A}$ may be an internal entity or an external entity. $\mathcal{A}$ may attempt to launch *MITM* attacks on the active connections between any two entities of the *SG* network. Since all messages over the network are encrypted, inherently, *MITM* attacks will not be successful to modify the transmitted information. Replay attacks are also prevented as each message over the network contains a unique timestamp value. As discussed in Section IV-C, the proposed protocol also defeats flood-based *DoS* attacks. In addition, impersonation attacks are prevented, since the fake request is discarded and the connection is terminated. $\mathcal{A}$ does not have the actual private key/shared secret key of the valid entity and therefore cannot decrypt the transmitted message. The key size of each shared secret key and public key/private key is chosen to be longer than 128 bits to resist against brute-force attacks. Furthermore, the Zip codes sent by the devices are used to overcome redirection attacks. Table II shows a comparison of the security capabilities of the proposed protocol with the existing protocols. Note that [20] and [21] partially protect *DoS* attacks by simply limiting the key agreement sessions.

C. Security Analysis

The proposed protocol provides *mutual authentication* between the *EP* and the *SM*, between the *SM* and the *GW*, and between the *SM* and the *HA*. Our protocol also provides a *perfect forward secrecy*, since the adversary $\mathcal{A}$ can neither retrieve the actual key nor predict any of the future keys using a shared secret key. Furthermore, our protocol *preserves the privacy* of communicated entities over the network and *overcomes ID thefts*, as the transmitted messages are always encrypted. Table III shows a comparison of security requirements. Note that we have a system with $|K| = |C| = |P|$, each of 128 bits (with *AES-CTR*) or 256 bits (with *MAES-CTR*) for symmetric encryption and $|K| \geq |C| = |P|$ for asymmetric encryption. Therefore, our system has perfect secrecy as each key is used with equal probability 1/$|K|$, and for each plaintext P and ciphertext C, there is a unique key K such that $E_K(P) = C$. As well, our system with at least equal size spaces $|P|=|C|=|K|$ is *perfectly key ambiguous* as the keys are picked uniformly, and for all $x \in P$, $y \in C$, there is a unique key K such that $y = E_K(x)$.

Furthermore, $\mathcal{A}$ cannot retrieve the partial and actual private keys of any device. Even in other scenarios where $\mathcal{A}$ extracts any one of these parameters 1) *PPR*, 2) *PPU*, and 3) public key, or replaces the public key of the device, our public encryption scheme is able to defend such attacks as $\mathcal{A}$ cannot retrieve the actual private key and cannot decrypt the message. Let us consider two scenarios in which $\mathcal{A}$ tries to extract some information.

Scenario-1: $\mathcal{A}$ does not have access to the *masterKey*, but may replace public keys (*PUK*) of the devices with any value, and also requests the public key of victim device, extracts the *PPR*, and makes decryption queries. Under this scenario, $\mathcal{A}$ has following restrictions:

1) $\mathcal{A}$ cannot extract the *PPR* of the challenge device *ID* at any point, as the fake *ID* will be discarded by the *TA*;
2) $\mathcal{A}$ cannot request the private key (*PRK*) of any identity, if the respective public key (*PUK*) has been replaced;
3) $\mathcal{A}$ cannot make a decryption query on the challenge ciphertext C that was generated by a combination of (*ID*, *PUK*).

Scenario-2: $\mathcal{A}$ does have access to the *masterKey*, but may not replace public keys (*PUK*) of the devices. $\mathcal{A}$ can compute *PPR* of any device, and also can request public key and make private key extraction and decryption queries. Under this scenario, $\mathcal{A}$ has following restrictions:

1) $\mathcal{A}$ cannot replace the public key (*PUK*) of any device at any time, as the identity and public key repositories are stored at various *TA*;
2) $\mathcal{A}$ cannot extract the private key (*PRK*) of the challenge device at any time, as it is randomly selected by each device;
3) $\mathcal{A}$ cannot successfully decrypt the challenge ciphertext C on behalf of the victim device, as it may generate *PPR* of the device, but does not have the actual private key (*PRK*) of the device.

Definition 2: A protocol is secure against adaptive chosen plaintext attack (*IND-CPA*) and chosen ciphertext attack (*IND-CCA*) for symmetric and asymmetric key cryptosystems, respectively, if no polynomial bounded adversary has a nonnegligible advantage. Therefore, our protocol is secure against *IND-CPA* and *IND-CCA*.

Our system is secure in terms of indistinguishability as $\mathcal{A}$ cannot identify the message choice because of a unique combination of P and K for each transmitted message C. Here, *Indistinguishability under chosen plaintext attack* (*IND-CPA*) is equivalent to the property of *semantic security*. In our protocol, symmetric encryption is performed by *AES-CTR*, which is *IND-CPA* secure. Also, the asymmetric encryption, performed by the proposed scheme, is based on *ECC* and is *indistinguishable under chosen ciphertext attack* (*IND-CCA*) considering hardness of the *ECDLP* [44].

TABLE IV
PERFORMANCE EVALUATION FOR A SINGLE AUTHENTICATION TOKEN

Performance Parameter	[20]	[1]	[21]	Proposed
Computation overhead	8*E*, 3*XOR*, 8*D*, 27*H*, 19*MUL*	3*E*, 3*D*, 2*H*, 2*HMAC*, 4*EXP*	13*H*, 3*MUL*, 2*XOR*, 1*ADD*, 1*SUB*, 4*EXP*	7*E*, 4*EMUL*, 7*D*, 1*ESUB*, 5*H*, 4*XOR*, 14*HMAC*, 1*MUL*, 1*ADD*
Communication overhead (bits)	3712	1152	1152	2752
Entities involved in authentication	*SM*, *HAN-GW*, *HA*, *BAN-GW*, *NAN-GW*	*HAN-GW*, *BAN-GW*	*SM*, *AS* of *DCU-GW*	*EP*, *SM*, *HA*, *HAN-GW*, *BAN-GW*, *NAN-GW*

D. Performance Analysis

A mutual authentication between the *HAN-GW* and the *BAN-GW* is proposed in [1], and a mutual authentication between the *SM* and the *AS* of the *DCU-GW* is proposed in [21]. A number of authentication scenarios between *SM*, *HAN-GW*, *BAN-GW*, *NAN-GW*, and *HA* are presented in [20], whereas our protocol proposes mutual authentication between *EP*, *SM*, *HAN-GW*, *BAN-GW*, *NAN-GW*, and *HA*. This section computes and compares *CMO* and *CPO* among these four protocols, and evaluates total execution time of the proposed protocol.

The total *CMO* and the total *CPO* of the protocol for a single authentication token are calculated, respectively, as $CMO_{\text{total}} = CMO_{\text{EP-SM}} + CMO_{\text{SM-GW}} + CMO_{\text{SM-HA}}$ and $CPO_{\text{total}} = CPO_{\text{EP-SM}} + CPO_{\text{SM-GW}} + CPO_{\text{SM-HA}} + CPO_{\text{key-gen}}$. Table IV shows a comparison of the *CMO* and *CPO* of our protocol with the existing protocols [1], [20], [21]. Out of these three existing protocols, it is fair to compare our protocol with only the protocol in [20], as only this protocol includes most of the involved entities in the *SG*, while only two entities are involved in [1] and [21]. Although, the protocol in [20] and our protocol cover a similar range of entities, our protocol achieves much lower overhead. In detail, authentication scenario between the *EP-SM* generates *CMO* of 1024 bits and prevents *MITM*, replay, impersonation, and redirection attacks. The scenario between the *SM-GW* generates 1216 bits of *CMO* and prevents *MITM*, replay, impersonation, repudiation, and flood-based *DoS* attacks. In comparison with the protocol in [1], our protocol is also resistant against flood-based *DoS* attacks while adding just 24 bits of *CMO*. Furthermore, in the authentication scenario between the *SM-HA*, our protocol prevents *MITM*, replay, impersonation, and brute-force attacks while generating 512 bits of *CMO*.

We also evaluate the performance of our protocol when there are multiple authentication tokens. We assume that there are m users executing the protocol simultaneously and each user has n *HA*s. The *CMO* generated by the proposed protocol is calculated as $CMO(m, n) = CMO(EP\text{-}SM)m + CMO(SM\text{-}GW)m + CMO(SM\text{-}HA)n = 1024m + 1216m + 512n = 2240m + 512n$. The *CPO* generated by the proposed protocol is calculated as $CPO(m, n) = (5\ m + 2n)E + (5m + 2n)D + (3m + n + 1)H + (10m + 4n)MAC + 1ESUB + 4mEMUL + 1MUL + mADD + (2m\text{-}2)EADD + 4mXOR$. Here, *E* and *D* represent encryption and decryption, respectively, *XOR* is bit-wise exclusive-OR, *MUL* and *ADD* are scalar multiplication and addition over integers/binaries, respectively, *EMUL*, *EADD*, and *ESUB* are elliptic curve multiplication, addition and subtraction

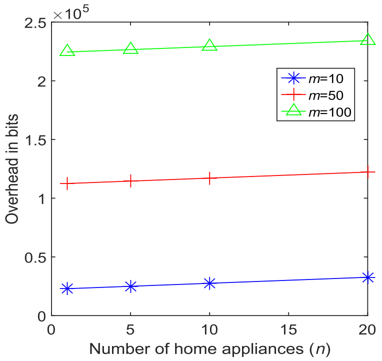


Fig. 7. Communication overhead.

(all three are computed as additions), respectively, and *H* and *MAC* are hash and authentication code functions, respectively. Furthermore, we assume that there are *r* malicious users in a batch. The protocol first removes invalid requests of the malicious users and then computes other parameters before further executing the protocol. In such case, the total recalculated *CPO* is as $CPO(m, n, r) = CPO(m, n) - r$*MUL* $- 2r$*EMUL* $-$ r*ESUB* $- 2(r-1)$*EADD*. Since *XOR* operations are negligible in comparison with other operations, they are not included in calculation of *CPO*.

Figs. 7 and 8, respectively, show the *CMO* and *CPO* generated by the proposed protocol for different number of users (m = 10, 50, 100) and *HA*s (n = 1, 5, 10, 20), considering unit value for each operation. In Fig. 7, $CMO(10, 1) =$ 2864 bytes, $CPO(10, 1) = 38.75$ bytes, $CMO(100, 20) =$ 29280 bytes, and $CPO(100, 20) = 397.625$ bytes. In Fig. 8, $CPO(100, 5, 1) = 380$ bytes, $CPO(100, 10, 50) = 348.875$ bytes, and $CPO(100, 20, 99) = 323.375$ bytes (worst case). Hence, even if there are some invalid requests *r* (Fig. 8(b): 1, *m*/2, and *m*−1) in a batch, the protocol efficiently handles them.

E. Simulation Result

We simulated the protocol in Java environment with JDK1.7, Intel Core i3-4500U CPU 1.7 GHz, 2GB RAM, and Windows7 OS. For a single authentication token, the scalar addition and multiplication operations over integer/binaries took 0.000933 and 0.00918 ms, single addition and doubling over elliptic curve took 0.6031 and 0.6047 ms, hash function *SHA256* took 0.9 ms, *HMAC* function *HMACSHA256* took 271.60 ms, and encryption and decryption times of symmetric *MAES-CTR* mode with 256 bits key between *EP-SM* and *SM-HA* took 0.97 and 0.78 ms, respectively. Moreover, the asymmetric encryption 1) using *RSA* with 2048 bits key and 2) using certificateless public encryption scheme took (30, 16) ms and (12, 7.6) ms, respectively. The total computation time by our protocol using *RSA* and using proposed scheme is 4041.91 and 3962.71 ms, respectively. This computation time can be further reduced by using the fast multiplication, where a single addition and doubling take approximately half of the ordinary *ECC* multiplication, *i.e.*, 0.303 ms [45]. The total messages (2752 bits) transmission times on *3G* and *4G* networks [46] by our protocol are 0.000451 and 0.000182 ms, respectively. Hence, the total execution time by our protocol (with certificateless cipher scheme) on *3G* and *4G* networks of approximately 3.96 s is quite reasonable, considering that it is the total time for completing authentication for all involved entities in the *SG* network. Here, we presented just one case for the overall protocol execution time. However, if we encrypt the message with *AES-CTR/MAES-CTR* for symmetric encryption, and the symmetric key is encrypted by an asymmetric algorithm, the overall time can be further reduced.

Keys generation of different entities are considered as a pre-execution phase, as all keys are generated before the protocol run starts. The key generation time varies with the generated random numbers and elliptic curve addition and doubling operations in our scheme. Let a represent the number of operations for elliptic curve addition and doubling points, and let b represent the number of devices deployed in the network. A random number generation takes 0.69 ms. Then, the generation time for the private and public master keys, *i.e.*, *PRK*$_{\mathrm{TA}}$ and *PUK*$_{\mathrm{TA}}$ are 0.69 and 0.60a ms, respectively. The total generation times for private keys (*z, x*) and public keys (*w, v*) are (0.69, 0.01)b ms and (0.60a, 0.60a)b ms, respectively. Therefore, total key generation time of our scheme is 0.69 + 0.60 $a + b$(0.70 + 1.20a) ms.

F. Formal Proof of the Properties of the Protocol

In order to justify our analysis, we use the *BAN-Logic* to provide a formal proof of our scheme. The notations used in *BAN-Logic* can be referred from [47].

1) Message Meaning Rule:

1) Rule shown at the bottom of the page.

2) Rule shown at the bottom of the page.

2) Timestamp Verification Rule:

1) $$\frac{SM_i| \equiv \#(T_i), SM_i| \equiv GW| \sim msg_1 \wedge msg_3}{SM_i| \equiv GW| \equiv msg_1 \wedge msg_3}$$

2) $$\frac{GW| \equiv \#(T_j), GW| \equiv SM_i| \sim msg_2}{SM_i| \equiv GW| \equiv msg_2}.$$

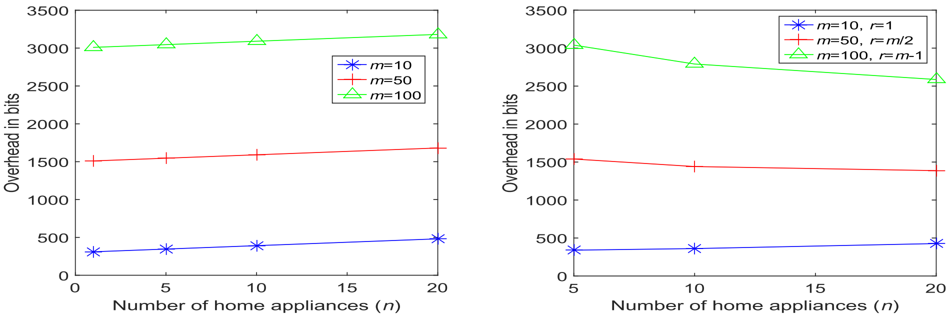
3) Jurisdiction Rule:

1) $$\frac{HA| \equiv SM \Rightarrow TID_{\mathrm{HA}}, HA \triangleleft HA| \sim TID_{\mathrm{HA}}}{HA| \equiv SM}$$

2) $$\frac{SM| \equiv HA \Rightarrow TID_{\mathrm{SM}}, SM \triangleleft SM| \sim TID_{\mathrm{SM}}}{SM| \equiv HA}.$$

$$\frac{EP| \equiv (EP \overset{SK_1}{\leftrightarrow} SM), EP \triangleleft E\{ID_{MP}, K_1, Zip_{\mathrm{EP}}\}_{\mathrm{PUK_{SM}}}}{EP| \equiv SM| \sim E\{ID_{\mathrm{EP}}, K_1, Zip_{\mathrm{EP}}\}_{\mathrm{PUK_{SM}}}}$$

$$\frac{SM| \equiv (SM \overset{SK_1}{\leftrightarrow} EP), SM \triangleleft E\{ID_{\mathrm{SM}}, K_2, Zip_{\mathrm{SM}}\}_{\mathrm{PUK_{EP}}}}{SM| \equiv EP| \sim E\{ID_{\mathrm{SM}}, K_2, Zip_{\mathrm{SM}}\}_{\mathrm{PUK_{EP}}}}.$$



(a) (b)

Fig. 8. Computation overhead. (a) $m = 10, 50, 100$; $n = 1, 5, 10, 20$. (b) $m = 10, 50, 100$; $n = 5, 10, 20$; $r = 1, m/2, m-1$.

4) Protocol Goals:

a) Mutual Authentication: $EP \mid \equiv SM \wedge EP \rightarrow SM \mid \equiv EP \wedge SM$. Thus, mutual authentication holds.

b) Session Key Agreement: Each key SK_1 between each *EP* and the *SM* provides session key agreement.

c) Freshness of messages: $SM \mid \equiv \#(T_j) \wedge EP \mid \equiv \#(T_i)$. Hence, freshness of messages between the *EP* and the *SM* holds.

d) Integrity and Privacy between the EP and the SM:

$$1)\ \frac{EP| \equiv (EP \overset{SK_1}{\leftrightarrow} SM), EP \triangleleft HMAC\{msg\}}{EP| \equiv SM| \sim msg}$$

$$2)\ \frac{EP| \equiv (EP \overset{SK_1}{\leftrightarrow} SM), EP \triangleleft E\{ID\}_{\mathrm{SK}_1}}{EP| \equiv SM| \sim ID}.$$

VI. Conclusion

The proposed protocol, based on hierarchical cloud *TA*s, provides mutual authentication between the *EP* and the *SM*, between the *SM/HAN-GW* and the *BAN-GW/NAN-GW*, between the *SM* and the *HA*, and between the *NAN-GW* and the *CC*. Particularly, the authentications between *EP-SM* and *GW-CC*, *SM-GW*, and *SM-HA* are, respectively, based on asymmetric key cryptography, asymmetric key cryptography in batch, and symmetric key cryptography. Processing requests in a batch improves the efficiency of the system, as a large number of *SMs* communicate with the *GW* simultaneously for mutual authentication. The certificateless scheme in the proposed protocol maintains privacy preservation as the transmitted message is always encrypted over the network. Simulation results show that the authentication scenarios between the *EP-SM*, the *SM-GW*, and the *SM-HA* generate lower *CMO* and *CPO* in comparison with the existing protocols. Also, the overhead generated by our protocol are manageable, even when invalid requests exist in a batch. Through security analysis, we show that our protocol is secure against existing attacks, such as *MITM* attacks, replay attacks, impersonation attacks, redirection attacks, and flood-based *DoS* attacks. In sum, our protocol is lightweight with low execution time and efficiently provides a centrally integrated control in a decentralized environment. Furthermore, our protocol can be readily integrated with the cloud computing-based trusted entities to utilize powerful computing services of the cloud for efficiently managing the *SG* system.

References

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

[2] M. Balakrishnan, "Smart energy solutions for home area networks and grid-end applications," in *Proc. Smart Energy*, 2012, pp. 67–73.

[3] W. Wang and Z. Lu, "Cyber security in the smart grid: Survey and challenges," *Comput. Netw.*, vol. 57, pp. 1344–1371, 2013.

[4] A. Bari, J. Jiang, W. Saad, and A. Jaekel, "Challenges in the smart grid applications: An overview," *Int. J. Distrib. Sens. Netw.*, vol. 2014, pp. 1–11, 2014.

[5] Smart grid cyber security, potential threats, vulnerabilities and risks, California State Univ., Long Beach, CA, USA, May. 2012. [Online]. Available: www.energy.ca.gov/2012publications/CEC-500-2012-047/CEC-500-2012-047.pdf. Accessed on: May. 22, 2015.

[6] K. Kursawe and C. Peters, "Structural weaknesses in the open smart grid protocol," *IACR*, Jun. 2015. [Online]. Available: https://eprint.iacr.org/2015/088.pdf. Accessed on: Jun. 26, 2015.

[7] Application Note: CyberFence Protection for DNP3, 3eTI, Ultra Electronics, Aug. 2015. [Online]. Available: http://www.ultra-3eti.com/assets/1/7/CyberFence_DNP3_Appliction_Note.pdf. Accessed on: Mar. 9, 2016.

[8] Guidelines for Smart Grid Cyber Security, NISTIR7628, Aug. 2010. [Online]. Available: csrc.nist.gov/publications/nistir/ir7628/nistir-7628_vol3.pdf. Accessed on: May. 24, 2015.

[9] Report to NIST on the smart grid interoperability standards roadmap, Electric Power Research Institute, Aug. 2009. [Online]. Available: www.nist.gov/smartgrid/upload/Report_to_NIST_August10_2.pdf. Accessed on: Jun. 6, 2015.

[10] A. Al-Majali, A. Vishwanathan, and C. Neuman, "Analyzing resiliency of the smart grid communication architectures under cyber attack," in *Proc. Cyber Security Exp. Test*, 2012, pp. 1–8.

[11] N. Saxena and B. J. Choi, "State of the art authentication, access control, and secure integration in smart grid," *Energies*, vol. 8, no. 10, pp. 11883–11915, Oct. 2015.

[12] P. Jovanovic and S. Neves, "Dumb crypto in smart grids: Practical cryptanalysis of the open smart grid protocol," *IACR*, Apr. 2015. [Online]. Available: https://eprint.iacr.org/2015/428.pdf. Accessed on: Jun. 14, 2015.

[13] DNP3 Secure Authentication Version 5, Apr. 2012. [Online]. Available: https://www.dnp.org/Lists/Announcements/Attachments/7/Secure Authenticationv5 2011-11-08.pdf. Accessed on: Jul. 4, 2015.

[14] IEC 62056-6-2:2013, Electricity Metering Data Exchange—The DLMS/COSEM Suite, Part 6-2: COSEM Interface Classes, 2006. [Online]. Available: https://webstore.iec.ch/publication/6410. Accessed on: Jul. 15, 2015.

[15] OpenADR and Cyber Security. [Online]. Available: http://www.openadr.org/cyber-security. Accessed on: Jul. 20, 2015.

[16] Remote Authentication Dial in User Service—RADIUS, Developing Solutions. [Online]. Available: https://www.developingsolutions.com/products/radius. Accessed on: Jul. 22, 2015.

[17] IEC TS 62351-6:2007, Power Systems Management and Associated Information Exchange—Data and Communications Security, Part 6: Security for IEC 61850. [Online]. Available: https:// webstore.iec.ch/publication/6909. Accessed on: Jul. 26, 2015.

[18] R. Sule, R. S. Katti, and R. G. Kavasseri, "A variable length fast message authentication code for secure communication in smart grids," in *Proc. IEEE Power Energy Soc. Gen. Meeting*, 2012, pp. 1–6.

[19] D. Li, Z. Aung, J. Williams, and A. Sanchez, "Efficient authentication scheme for data aggregation in smart grid with fault tolerance and fault diagnosis," in *Proc. IEEE PES Innovative Smart Grid Technol.*, 2012, pp. 1–8.

[20] H. Nicanfar and V. Leung, "Multilayer consensus ECC-based password authentication key-exchange protocol for smart grid system," *IEEE Trans. Smart Grid*, vol. 4, no. 1, pp. 253–264, Mar. 2013.

[21] H. Nicanfar, P. Jokar, and V. Leung, "Efficient authentication and key management mechanisms for smart grid communications," *IEEE Syst. J.*, vol. 8, no. 2, pp. 629–640, Jun. 2014.

[22] S. Oh and J. Kwak, "Mutual authentication and key establishment mechanism using DCU certificate in smart grid," *Appl. Math. Inform. Sci.*, vol. 6, no. 1, pp. 257S–264S, 2012.

[23] H. Li, R. Lu, L. Zhou, B. Yang, and X. Shen, "An efficient Merkle-tree-based authentication scheme for smart grid," *IEEE Syst. J.*, vol. 8, no. 2, pp. 655–662, Jun. 2014.

[24] A. R. Metke and R. L. Ekl, "Smart grid security technology," in *Proc. Innovative Smart Grid Technol.*, 2010, pp. 1–7.

[25] K. Kursawe, G. Danezis, and M. Kohlweiss, "Privacy-friendly aggregation for the smart grid," in *Proc. Privacy Enhancing Technol.*, 2011, pp. 175–191.

[26] C. Bekara, T. Lucken, and K. Bekara, "A privacy preserving & secure authentication protocol for advanced metering infrastructure with non-repudiation service," in *Proc. ENERGY*, 2012, pp. 60–68.

[27] Y. Yan, R. Hu, and S. Das, "An efficient security protocol for advanced metering infrastructure in smart grid," *IEEE Network*, 2013, pp. 64–71.

[28] F. Luo, Z. Y. Dong, Y. Chen, Y. Xu, and K. P. Wong, "Hybrid cloud computing platform: The next generation IT backbone for smart grid," in *Proc. PES Gen. Meeting*, 2012, pp. 1–7.

[29] S. Rusitschka, K. Eger, and C. Gerdes, "Smart grid data cloud: A model for utilizing cloud computing in the smart grid domain," in *Proc. Int. Conf. Smart Grid Commun.*, 2010, pp. 483–488.

[30] J. Baek, Q. H. Vu, J. K. Liu, X. Huang, and Y. Xiang, "A secure cloud computing based framework for big data information management of smart grid," *IEEE Trans. Cloud Comput.*, vol. 3, no. 2, pp. 233–244, Jun. 2015.

[31] S. Bera, S. Misra, and J. J. P. C. Rodrigues, "Cloud computing applications for smart grid: A survey," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 5, pp. 1477–1494, May 2015.

[32] A. H. Mohsenian-Rad and A. Leon-Garcia, "Coordination of cloud computing and smart power grids," in *Proc. IEEE Int. Conf. Smart Grid Commun.*, 2010, pp. 368–372.

[33] Y. Jiang, X. Guo, C. Li,, H. Wen, C. Lei, and Z. Rui, "An efficient and secure search database scheme for cloud computing in smart grid," in *Proc. Conf. Commun. Netw. Security*, 2013, pp. 413–414.

[34] B. Bitzer and E. S. Gebretsadik, "Cloud computing framework for smart grid applications," in *Proc. 48th Int. Univ. Power Eng. Conf.*, 2013, pp. 1–5.

[35] Cyber security challenges in using cloud computing in the electric utility industry, Pacific Northwest Nat. Lab., Prepared for the U.S. Department of Energy under Contract DE-AC05-76RL01830, PNNL-21724, Sep. 2012. [Online]. Available: http://www.pnnl.gov/main/publications/external/technical_reports/pnnl-21724.pdf. Accessed on: Aug. 2, 2015.

[36] H. Li, Y. Dai, and H. Yang, "Identity-based authentication for cloud computing," in *Proc. CloudCom*, 2009, pp. 157–166.

[37] H. Lim and K. Paterson, "Identity-based cryptography for grid security," *Int. J. Inform. Security*, vol. 10, pp. 15–32, 2011.

[38] J. H. Oh, K. K. Lee, and S. Moon, "How to solve key escrow and identity revocation in identity-based encryption schemes," in *Proc. Inform. Syst. Security*, 2005, pp. 290–303.

[39] J. Baek, R. Naini, and W. Susilo, "Certificateless public key encryption without pairing," in *Proc. Info. Security*, 2005, pp. 134–148.

[40] M. Scott, "On the efficient implementation of pairing-based protocols," in *Proc. IACR Cryptol.*, 2011, pp. 334–346.

[41] D. R. L. Brown, SEC 1: Elliptic curve cryptography, Standards for Efficient Cryptography. [Online]. Available: http://www.secg.org/sec1-v2.pdf. Accessed on: Aug. 5, 2015.

[42] Smart Grid System Report, U.S. Dept. Energy, Jul. 2009. [Online]. Available: smartgrid.gov/sites/default/files/resources/ systems_report.pdf. Accessed on: Aug. 12, 2015.

[43] N. Saxena and N. S. Chaudhari, "EasySMS: A protocol for end-to-end secure transmission of SMS," *IEEE Trans. Inf. Forensics Security*, vol. 9, no. 7, pp. 1157–1168, Jul. 2014.

[44] Security Guidelines for Cryptographic Algorithms in the W3C Web Cryptography API, W3C/MIT Cryptosense/INRIA, Nov. 2015. [Online]. Available: https://www.w3.org/2012/webcrypto/draft-irtf-cfrg-webcrypto-algorithms-00#ECDSA. Accessed on: Aug. 20, 2015.

[45] R. P. Gallant, R. J. Lambert, and S. A. Vanstone, "Faster point multiplication on elliptic curves with efficient endomorphisms," *IACR*, 2001. [Online]. Available: https://www.iacr.org/ archive/crypto2001/21390189.pdf. Accessed on: Aug. 22, 2015.

[46] Measuring mobile broadband performance in the UK, Nov. 13, 2014. [Online]. Available: http://stakeholders.ofcom.org.uk/binaries/research/broadband-research/mbb-nov14.pdf. Accessed on: Aug. 23, 2015.

[47] J. Wessels, Applications of BAN-logic, CMG Finance, Apr. 2001. [Online]. Available: win.tue.nl/ipa/archive/springdays2001/banwessels.pdf. Accessed on: Aug. 23, 2015.

Neetesh Saxena (S'10–M'14) received the Ph.D. degree in computer science and engineering from IIT Indore, India.

He is currently working as a Postdoctoral Researcher at the Georgia Institute of Technology, Atlanta, GA, USA. Prior to this, he was with the State University of New York Korea as a Postdoctoral Researcher and a Visiting Scholar at the Stony Brook University, USA. From 2013 to 2014, he was a Visiting Research Student and a DAAD Scholar with the B-IT, Rheinische-Friedrich-Wilhelms Universitt, Bonn, Germany. He was also a TCS Research Scholar from January 2012 to April 2014. He has published several papers in international peer-reviewed journals and conferences. His current research interests include smart grid security, vehicle-to-grid security and privacy, cryptography, security, and privacy in the cellular networks, and secure mobile applications. He is a member of ACM.

Bong Jun Choi (S'09–M'11) received the B.Sc. and M.Sc. degrees from Yonsei University, Seoul, South Korea, both in electrical and electronics engineering, and the Ph.D. degree from the University of Waterloo, Waterloo, ON, Canada, in electrical and computer engineering.

He is currently an Assistant Professor at the Department of Computer Science, State University of New York Korea, Incheon, South Korea, and jointly a Research Assistant Professor at the Department of Computer Science, Stony Brook University, New York, NY, USA. His current research interests include energy efficient networks, distributed mobile wireless networks, smart grid communications, and network security. He is an Editor of KSII *Transactions on Internet and Information Systems* and a member of the Smart Grid Core Security Technology Development Steering Committee, South Korea. He also serves on the technical program committees for many international conferences such as IEEE PECON, IFIP NTMS, and IEEE CMC. He is a member of the ACM.