IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, VOL. 14, NO. 8, AUGUST 2019

Universally Composable Key Bootstrapping and Secure Communication Protocols for the Energy Internet

Abubakar Sadiq Sani, *Member, IEEE*, Dong Yuan, *Member, IEEE*, Wei Bao , *Member, IEEE*,

Zhao Yang Dong, *Fellow, IEEE*, Branka Vucetic, *Fellow, IEEE*, and Elisa Bertino , *Fellow, IEEE*

***Abstract*—The Energy Internet is an advanced smart grid solution to increase energy efficiency by jointly operating multiple energy resources via the Internet. However, such an increasing integration of energy resources requires secure and efficient communication in the Energy Internet. To address such a requirement, we propose a new secure key bootstrapping protocol to support the integration and operation of energy resources. By using a universal composability model that provides a strong security notion for designing and analyzing cryptographic protocols, we define an ideal functionality that supports several cryptographic primitives used in this paper. Furthermore, we provide an ideal functionality for key bootstrapping and secure communication, which allows exchanged session keys to be used for secure communication in an ideal manner. We propose the first secure key bootstrapping protocol that enables a user to verify the identities of other users before key bootstrapping. We also present a secure communication protocol for unicast and multicast communications. The ideal functionalities help in the design and analysis of the proposed protocols. We perform some experiments to validate the performance of our protocols, and the results show that our protocols are superior to the existing related protocols and are suitable for the Energy Internet. As a proof of concept, we apply our functionalities to a practical key bootstrapping protocol, namely generic bootstrapping architecture.**

***Index Terms*—Energy Internet, security, universal composability, key bootstrapping, communication.**

I. INTRODUCTION HE Energy Internet is a software-based smart grid that supports efficient management and sharing of energy resources through the Internet [1], [2]. Security attacks on the energy resources are relatively simple to carry out. Communication among energy resources can be used to steal data and disrupt energy operations. Without understanding the behavior

**T**

|  |
| --- |
| Manuscript received October 16, 2018; revised December 23, 2018; accepted December 29, 2018. Date of publication January 10, 2019; date of current version May 16, 2019. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Ivan Visconti. *(Corresponding author: Abubakar Sadiq Sani.)*  A. S. Sani, D. Yuan, W. Bao, and B. Vucetic are with the Faculty of Engineering and Information Technologies, The University of Sydney, Sydney, NSW 2006, Australia (e-mail: sadiq.sani@sydney.edu.au; dong.yuan@ sydney.edu.au; wei.bao@sydney.edu.au; branka.vucetic@sydney.edu.au).  Z. Y. Dong is with the School of Electrical Engineering and Telecommunications, University of New South Wales, Sydney, NSW 2052, Australia (e-mail: joe.dong@unsw.edu.au).  E. Bertino is with the Department of Computer Science, Purdue University, West Lafayette, IN 47907 USA (e-mail: bertino@purdue.edu).  Digital Object Identifier 10.1109/TIFS.2019.2892005 |

of adversaries that execute these attacks, the Energy Internet will not be able to provide appropriate security measures to mitigate the attacks. This paper makes a step in this direction by analyzing adversarial behavior during key bootstrapping and communication in the Energy Internet.

Security bootstrapping [3] is a widely adopted means to support secure communication in the smart grid. Shared secret session keys derived via key bootstrapping makes it possible for users or components to securely exchange data. However, the criticality of key bootstrapping and communication makes it a high potential target for security attacks that can cause outages. Session keys can be exploited by adversaries in a short amount of time [4], thereby introducing huge security concerns in the Energy Internet.

Limited progress has been made towards developing techniques for detecting security attacks and presence of adversaries during key bootstrapping and communication in the Energy Internet. Some studies on security in the Energy Internet have shown that security innovations are required [1], [5] to support and improve its growth and deployment. Additionally, most smart grid based secure communication solutions like the architecture proposed by Kounev *et al.* [6] lack adequate security and their limitations could be exploited to disrupt energy operations (see Section II). Thus, additional security measures are required to enhance secure communication in the Energy Internet as well as smart grid.

User Datagram Protocol (UDP) is one of the widelyrecognized transport layer protocols capable of performing several communication functions in power systems [7]. A simple communication network of the Energy Internet is presented in Figure 1. This figure shows the secure communication channels and control between the sensors and actuators attached to the physical devices of the smart grid and other energy networks (e.g. gas) in the Energy Internet.

|  |
| --- |
| 1556-6013 © 2019 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications\_standards/publications/rights/index.html for more information. |

We stress that lack of tools for detecting the presence of adversaries during key bootstrapping as well as lack of secure and efficient communication are major hindrances towards the broad integration of energy resources in the smart grids. Based on a universal composability model [9], [27], which allows modular design and analysis of cryptographic protocols, provides cryptographic soundness and security in arbitrary adversarial environments, and allows one to show that the protocols are secure and adversaries can be detected

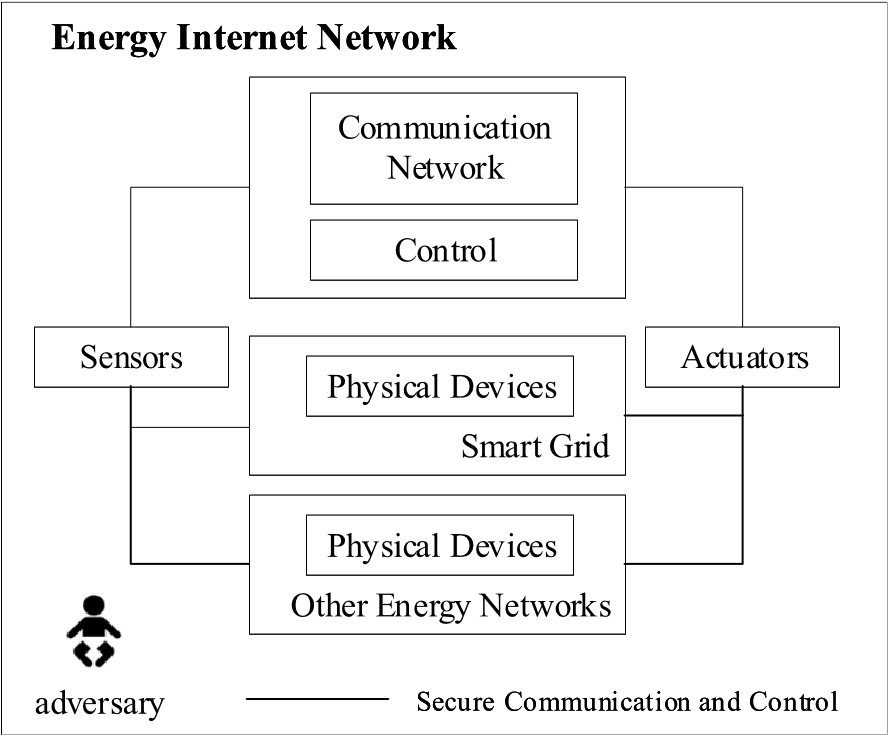


Fig. 1. A simple communication network of the energy internet.

in these environments (see, e.g., [8]–[11]), we design a Key Bootstrapping Protocol (KBP) and a Secure Communication Protocol (SCP) to enhance key bootstrapping and secure communication, respectively, in the Energy Internet. In the universal composability model, real protocols may use ideal functionalities as functions to perform their cryptographic operations. Composition theorems [9], [29] available for the universal composability model then allow replacements of the ideal functionalities by their realizations, which then implies that the actual protocols using the realizations realize other ideal functionalities. Furthermore, our protocols are supported by an enhanced form of an Elliptic Curve Cryptography (ECC) [12] algorithm, which we proposed and referred to as a twofactor mutually authenticated ECDH key exchange, 2MAECDH, which enforces user authentication and two-factor mutual authentication before generating public/private keys and deriving session keys, respectively, to prevent active manin-the-middle attacks during key exchange. More specifically, our contributions are as follows.

* We propose an ideal functionality *FCP* for cryptographic operations related to our KBP and SCP (see below). This functionality supports many cryptographic primitives such as authentication and our 2MA-ECDH, which is based on the ECDH key exchange that exhibits very low computational cost. We show that *FCP* can be realized under standard cryptographic assumptions and that it ensures that an adversary on the network cannot interfere with the key bootstrapping and secure communication protocols when these protocols are using *FCP* to perform local computations.
* We propose and prove a realization *PCP* of *FCP* based on standard cryptographic assumptions. The proof requires several hybrid arguments as *FCP* supports many cryptographic operations.
* We propose an ideal functionality *FCPK B* for key bootstrapping and secure communication with perfect forward secrecy and efficient transition from key bootstrapping to secure communication. This functionality allows our protocols to use exchanged keys in an ideal manner.
* We provide the above mentioned key bootstrapping and secure communication protocols which use *FCP* and realize *FCPK B*. Due to the use of *FCP*, the security analyses and proofs of these protocols do not require any hybrid arguments or probabilistic reasoning.
* We perform some experiments to analyze the performance of our protocols, and then show that our protocols are efficient for the Energy Internet, i.e., they meet the latency and reliability requirements of applications and technologies in the smart grid.
* We analyze the Generic Bootstrapping Architecture (GBA) [30] and find some weaknesses in its bootstrapping procedure. We then introduce, using our functionalities, an enhancement of the GBA protocol that is a secure universally composable key bootstrapping protocol.

The rest of the paper is organized as follows: Section II provides an overview of the related works. In Section III, we briefly recall the general notion of the universal composability and the universal composability model that we use in this work, and present a data and communication model. Section IV presents our ideal functionality *FCP* and its realization *PCP*. The proof of *PCP* is also presented in this section. Our ideal functionality *FCPK B* for key bootstrapping and secure communication is presented in Section V. The security analysis of our protocols is presented in Section VI. The performance analysis of our protocols and their comparison with the existing related protocols are presented in the experimental setup in Section VII. The case study is carried out in Section VIII. We draw conclusions and give suggestions for future work in Section IX.

# II. RELATED WORKS

Many cryptographic protocols for securing communication have been proposed for the smart grid and each has tried to achieve specific security goals (see, e.g., [6] and [13]–[20]). Kounev *et al.* [6] proposed a secure real-time communication architecture to support microgrid operation and control. A microgrid is a standalone grid for integrating renewables such as solar and wind with the main smart grid. However, the architecture presents a single point of failure to the microgrid via the server during key bootstrapping. Additionally, encrypted unicast and multicast communications are available to all components that possess the microgrid’s shared confidentiality communication key. By contrast, our approach eliminates the single point of failure during key bootstrapping and enhances the secrecy of unicast and multicast communications.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| TABLE I  FUNCTIONALITIES AND LIMITATIONS OF RELATED APPROACHES   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | |  | |  |  |  |  |  |  | |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |

Nicanfar *et al.* [13] proposed an authentication and key agreement protocol for smart grid communication using symmetric and asymmetric encryption operations. However, the protocol has huge computational and communication costs. To achieve lightweight security for key agreement in the smart grid, ECC-based schemes have been proposed [21] which can also be utilized for the Energy Internet. We note that it is very important that communication schemes for smart grid as well as the Energy Internet meet the latency and reliability requirements of the smart grid applications and technologies which range from 1*ms* to 1*min* and from 99% to 99*.*99%, respectively [22]–[25].

Mohammadali *et al.* [14] proposed two ECC-based novel identity-based key establishment protocols, NIKE and NIKE+, for advanced metering infrastructure in the smart grid. There are a total of seven key exchange messages in NIKE, but the authors succeeded in reducing them to six using NIKE+. Both NIKE and NIKE+ use several ECC point multiplication operation, which increases the computational load. Ancillotti *et al.* [16] identified some fundamental security requirements, like Public Key Infrastructure (PKI) communication in smart grids. While PKI is an effective security fundamental solution for secure communication in the smart grid, its high computational cost and the heavy reliance on the availability of trusted certificate authorities are considered as limitations for efficient communication.

Although many of the proposed solutions succeeded in providing communication security for the smart grid, these solutions suffer from one or more of the following drawbacks: i) solution analysis based on symbolic methods and/or theoretical security analysis which do not offer cryptographic soundness and security in arbitrary adversarial environments (see, e.g., [6], [13], [14], and [17]–[19]); ii) high computational and communication costs for smart grid applications (see, Section VII); iii) the assumption that each component preshares a unique password or key with other components; however, this assumption is not realistic and requires another protocol, which makes it difficult to justify the implementation in a real environment; thus, extensive justifications for deriving and using pre-shared passwords (without relying on another protocol) are required (see, e.g., [6] and [14]); and iv) lack of evidence about meeting the latency and reliability requirements of smart grid applications and technologies (see e.g., [13] and [14]). A summary of functionalities and limitations of the related approaches is presented in Table I. In this paper, we propose integrated solutions which address all those drawbacks, and more importantly, provide secure and efficient key bootstrapping and communication in the Energy Internet. Furthermore, we present a case study of the key bootstrapping procedure of the GBA in which the use of our functionalities allow us to analyze and enhance the protocol.

# III. PRELIMINARIES

In this section, we briefly describe the general notion of universal composability, the universal composability model we use, and present a data and communication model for the

Energy Internet.

1. *General Notion of Universal Composability*

The general notion behind universal composability involves real and ideal protocols. The desired behavior of a protocol and its intended security properties are represented by an ideal protocol, also known as ideal functionality. The real protocol, which represents the protocol one would like to analyze after designing the ideal protocol should be at least as secure as the ideal protocol, i.e., it is supposed to realize the ideal protocol. An ideal adversary (or simulator) exists for the ideal protocol while a real adversary exists for the real protocol, such that the real and ideal settings cannot be distinguished by any environment. Hence, by definition of universal composability, since no successful attack exists on the ideal protocol, there is no attack on the real protocol that can be successful. We note that the universal composability model has been utilized for designing and analyzing cryptographic protocols for the smart grid (see, e.g., [26]).

1. *IITM Model*

The universal composability model we use in this work is the Inexhaustible Interactive Turing Machine (IITM) model [9], [27] with responsive environments [28]. It consists of a general computational model and general composition theorems. The general computational model of the IITM model represents systems of interactive Turing machines. A machine (or interactive Turing machine) is defined as a probabilistic polynomial-time Turing machine with named two-direction tapes. In a system *Sy* = *M*1|*...*|*Mk*|!*M*1|*...*|!*Mk* of IITM, where *Mj* and *Mj* are machines, and ! indicates that an unbounded number of (new) copies of machines may be generated in a run of *Sy*, the names are used to determine the connections of machines in *Sy*. In a run of *Sy*, the first machine to be triggered is called a master machine. A machine *M*1 can be triggered by another machine *M*1 if *M*1 sends a message to *M*1 on a tape that connects them. Two systems *X* and *Y* are indistinguishable *(i.e., X* ≡ *Y)* if the difference between the probability that *X* outputs 1 and the probability that *Y* outputs 1 is negligible. We now consider three different types of systems in the IITM model as follows: i) real and ideal protocols; ii) adversaries and simulators; and iii) environments. The systems with both network and I/O interfaces are the protocol systems and environment systems, while the systems

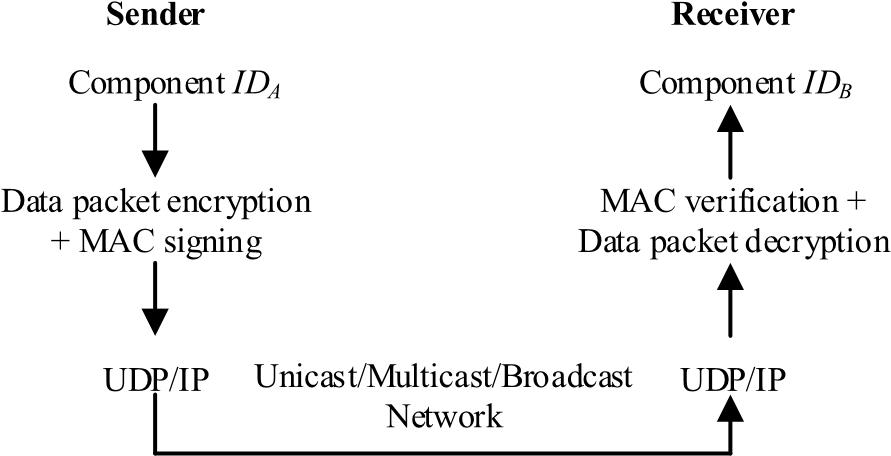


Fig. 2. Data and communication model.

with only a network interface are the adversarial systems. If the environment systems and adversarial systems respond immediately to so-called restricting messages on the network, we say that these systems are responsive, thus responses to restricting messages are immediately provided by the environments/adversaries. Restricting messages on the network in the IITM model can be represented in the form *(*Respond*,id,m)*, where *id* and *m* are random bit strings. These messages prevent the adversary from interfering with/disrupting a protocol run, thereby allowing us to enforce the natural execution of the protocol (see [28] for more details). We now present the definition of simulation-based security in universal composability which signifies strong simulatability.

*Definition 1 [39]: Let P and F be protocol systems, i.e., the real and ideal protocol, respectively, with similar I/O interfaces. Then, P realizes F (P* ≤ *F) if there exists an adversarial system S (an ideal adversary or a simulator) such that P and S*|*F have similar external interfaces and for all environmental systems E, connecting to the external interface of P (and thus, S*|*F), it holds true that E*|*P* ≡ *E*|*S*|*F, where the adversary in E*|*P is omitted as it is subsumed by E.*

In the IITM model, there are several general composition theorems that handle composition of protocol systems. Theorem 1 below handles composition of a fixed number of protocol systems simultaneously.

*Theorem 1 [39]: Let P*1*, P*2*, F*1*, F*2 *be protocol systems such that P*1 *and P*2 *as well as F*1 *and F*2 *only connect with each other via their I/O interfaces and Pi* ≤ *Fi, for i* ∈ 1*,*2*. Then, P*1|*P*2 ≤ *F*1|*F*2*.*

Other composition theorems which guarantee secure composition of an unbounded number of copies of a protocol system can be found in [9] and [29]. To construct more complex systems, composition theorems of the IIITM model can be combined and applied in an iterative manner.

*C. Data and Communication Model*

As a standard practice in real-time communication, transport layer protocols are utilized for end-to-end data exchange. A simple data and communication model for the Energy Internet is presented in Figure 2 where the User Datagram Protocol/Internet Protocol (UDP/IP) [40] is considered for unicast/multicast/broadcast communication due to its low latency requirement and protection against data integrity errors, to name a few [7]. In this work, we assume that the size of a data packet *ZP* (in bits) from a component can be calculated as *Mi* +*tag(s)*, where *Mi* is the size of the message payload in the smart grid and *tag(s)* represents the size of security feature(s) added to enhance the security of the data packet (say, 256-bit Adavanced Encryption Standard (AES) Cipher-based Message Authentication Code (CMAC) and/or 128-bit AES). Some smart grid applications can have a minimum *Mi* of 33 bytes for UDP [42]. These values of *Mi* are utilized in this work without loss of generality (see Section VII).

IV. IDEAL FUNCTIONALITY FOR CRYPTOGRAPHIC

# PRIMITIVES IN KEY BOOTSTRAPPING AND SECURE COMMUNICATION

In this section, we present our ideal functionality *FCP* that supports our 2MA-ECDH. The 2MA-ECDH is similar to the existing ECDH key exchange, except that 2MAECDH provides two-factor mutual authentication during key bootstrapping. In 2MA-ECDH, every key bootstrapping user verifies the identity and attribute of its key bootstrapping partner before deriving a shared key to prevent active manin-the-middle attacks. A key bootstrapping protocol (or secure communication protocol) *P* can use *FCP* for its key bootstrapping (or secure communication) cryptographic operations (see Section V). We argue that *P*|*FCP* ≤ *F* (i.e., a key bootstrapping protocol using *FCP* realizes some ideal key bootstrapping functionality *F*). Using the composition theorems of the IITM model (see Section II), we can replace *FCP* with its realization *PCP* after *P*|*FCP* ≤ *F* has been proven, where all ideal operations provided by *FCP* are replaced by their real counterpart (see below). The main guarantees provided by *FCP* are as follows: i) for key bootstrapping, *FCP* guarantees that only the authenticated owners of a 2MAECDH key can have access to secret session keys derived from the key; and ii) for secure communication, *FCP* guarantees that only the authenticated initiator and specified authenticated responder(s) can have access to specific session keys (see below for more details).

The ideal functionality *FCP* allows its users to perform the following key bootstrapping and secure communication cryptographic operations in an ideal manner: i) generate identity values for authenticating users; ii) generate pre-shared and confidential nonces; iii) generate authenticated pre-shared keys for authenticated encryption; iv) generate authenticated public/private keys; v) validate the status of public keys, identity values, and pre-shared nonces; vi) generate 2MAECDH keys; vii) perform key confirmation; viii) derive secret session keys from 2MA-ECDH keys and identity values; ix) derive multicast session keys from authenticated private keys and confidential nonces; x) encrypt and decrypt messages using secret session keys and multicast session keys; and xi) compute and verify MACs.

Formally, *FCP* is machine with *n* I/O tapes that represents different roles in a protocol, and a network tape for communicating with the adversary. There is always one instance of *FCP* which handles all requests in every run that uses it (i.e., *FCP*) for key bootstrapping and secure communication cryptographic operations. A user of *FCP* is recognized by a tuple *(I D,lsid,r)*, where *I D* is the user identity, *lsid* is a local session identifier, and *r* is the role/tape that connects the user to *FCP*, *I D*, and *lsid* (we assume that every *lsid* is always selected and managed by the protocol). To identify the user who sent/received messages in *FCP*, all messages (on the I/O tapes) are prefixed with *(I D,lsid)*. Furthermore, every user requires an identity value *(iv)* that shows that the user has been authenticated for key bootstrapping. In *FCP*, we use restricting messages (cf. Section III) to assure that the following properties hold: i) the adversary cannot interfere with any run of *FCP*; ii) the adversary always responds to *FCP*’s requests; and iii) all operations in *FCP* always succeed. The format of the restricting messages in the network tape is *(*Reply*,restricted,m)*, where m is random bit string and Reply is the content of the message.

To model secret keys and confidential nonces in *FCP*, users can only access pointers instead of the actual values of these keys and nonces. The users can use these pointers to perform several key bootstrapping and secure communication cryptographic operations (see below). *FCP* maintains several sets of these actual values stored in a database within the functionality. These sets include Private, CNonces, 2MA-ECDHKeys, PSKeys, SSKeys, and MSKeys, for storing the actual values of private keys, confidential nonces, 2MA-ECDH keys, preshared keys, secret session keys, and multicast session keys, respectively. Furthermore, *FCP* also maintains the following sets: i) a set BlockedPublic of blocked public keys (to private keys) that cannot be generated again; ii) a set EIUsers of users’ identities *I D(s)*; iii) a set of IDValues of identity values *iv(s)* issued to all users; and iv) a set PSKNonce of preshared nonces *nc(s)* for computing a pre-shared key. *FCP* uses these sets to prevent guessing and collisions of related keys, nonces, users’ identities, and identity values (see below). For example, if a new uncorrupted private key *d* is created, then *d* ∈*/* Private. Any key that cannot be verified by *FCP* is treated as an unverified key and as such the key or any derived key from such a key cannot be used in *FCP* (see below), thus, a set UnverifiedKeys is also maintained by *FCP* for storing all keys created from unverified keys (and values).

We parameterize *FCP* with a GenDP*(*1*η)* algorithm that is used to generate the ECDH domain parameters. The algorithms takes a security parameter *η* and returns *(p,a,b, G,n,h)*, where *p* is a prime modulus, *a* and *b* are curve parameters, *G* is a generator point, *n* is the order of *G*, and *h* is a cofactor. Upon the activation of *FCP* for the first time, we allow *FCP* to execute GenDP*(*1*η)* and store the generated domain parameters *(p,a,b, G,n,h)*. Additionally, we expect *FCP* to receive a list of users’ identities *(*EIUsers*, I Ds)* from a Distributed Identity Registration Center *(DI RC)*, where *I Ds* represent users’ identities in the Energy Internet. We note that the *DI RC* is a distributed entity for availability and accuracy but not security and privacy, and

can always update the *I Ds* in the set EIUsers. Then, *FCP* sends the algorithm, users’ identities list, and a cryptographic algorithms request to the adversary via a restricting message. Upon the completion of this activation, *FCP* returns control to the adversary if a response is received on a network tape.

Otherwise, if the response is received on the I/O tape, *FCP* continues to process the original message.

The commands that *FCP* provides to a user *(I D,lsid,r)* on the I/O interface are listed in Table II. Other commands such as Derive a new pre-shared key I (GenPSKeyI), Derive a new pre-shared key II (GenPSKeyII), Generate a new confidential nonce (GenCNonce), Multicast session key derivation (DeriveMulticastKey), Encryption using multicast session key (MEnc), and Decryption using multicast session key (MDec) returns *(PSKeyI Pointer, ptr*3*)*, *(PSKeyI I Pointer, ptr*4*)*, *(CNoncePointer, ptr*6*)*, *(MulticastKeyPointer, ptr*7*)*, resulting ciphertext, and plaintext, respectively to the user at the end of its execution. The description of the six aforementioned commands is omitted due to page limit.

We note that we restrict *FCP* to have feasible computation times. We plan to explore a formulation of additional commands such as adding and removing users in EIUsers, and revoking keys based on the above commands in future work.

We now construct a realization *PCP* of *FCP*. The realization

*PCP* implements all operations of *FCP* via well-known cryptographic schemes. Formally, *PCP* is a machine with similar I/O and network interfaces as *FCP*. We parameterize *PCP* with an algorithm *GenDP(*1*η)* that has the same properties as for *FCP*, a MAC scheme *MAC*, three encryption schemes

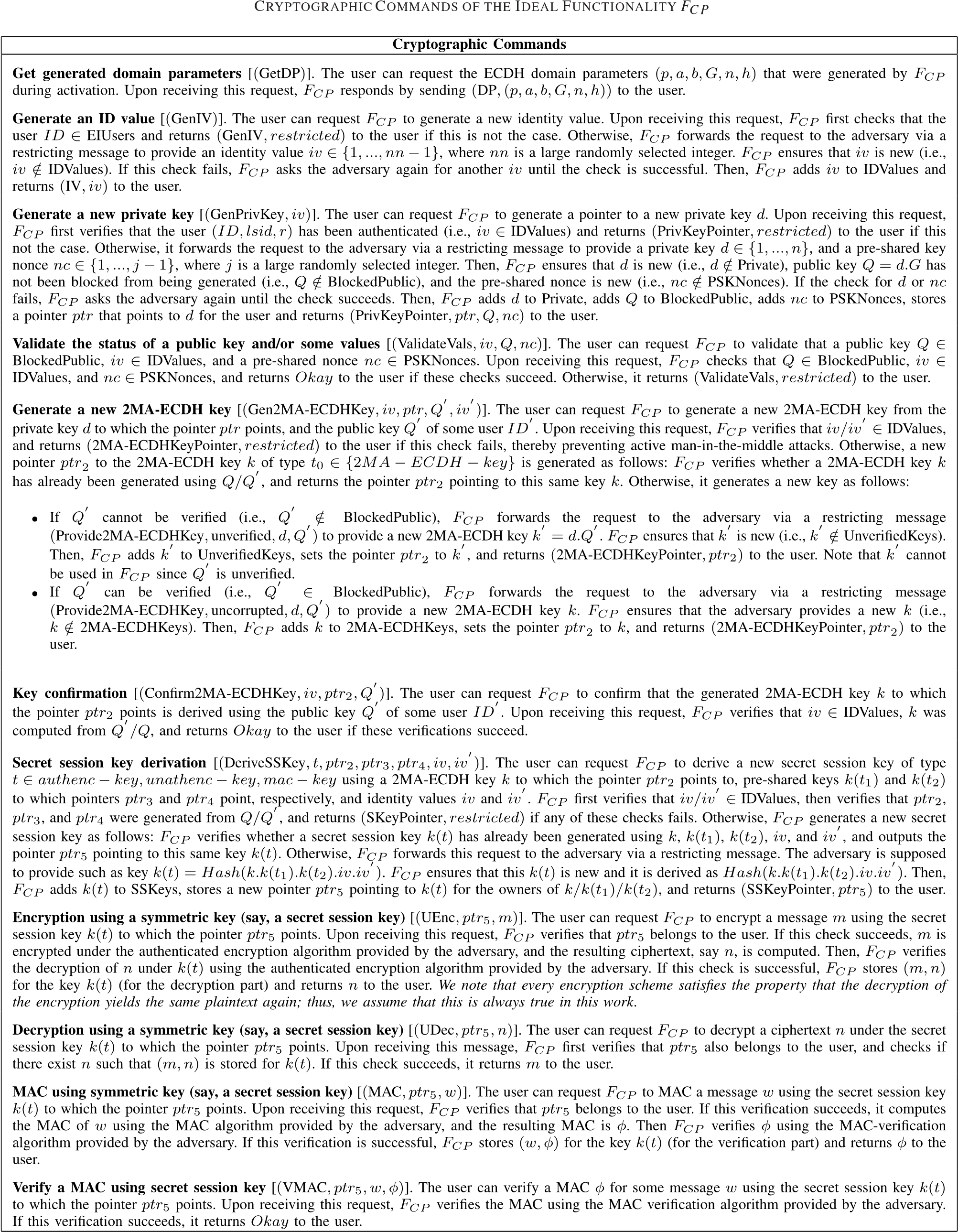
*authenc*, *unauthenc*, *pubenc* for authenticated symmetric key encryption, unauthenticated symmetric key encryption, and public key encryption, respectively, and three families of

PRF *F* *N*, *F* *N*, and *F*= *N* that take key(s) and salt as input, and output a key (see [31] for formal definitions of these cryptographic primitives). Upon the activation of *PCP* for the first time, *PCP* receives some message *m* and activates itself by executing the algorithm GenDP and storing results of the execution before processing the received message *m*. Similar to *FCP*, *PCP* maintains keys, identity values, nonces, and pointers to secret keys in a database within the functionality, and uses them to select which type of cryptographic primitive to execute. For example, *F* is used for deriving pre-shared keys from 2MAECDH keys, *F* is used for deriving secret session key from 2MA-ECDH keys and pre-shared keys, and *F* is used for deriving multicast session key from private keys. However, it does not maintain any set of *FCP*. The implementation of the above commands in *PCP* is given in Table III. Other commands such as GenPSKeyI, GenPSKeyII, GenCNonce, DeriveMulticastKey, MEnc, and MDec returns *ptr*3, *ptr*4, *ptr*6, *ptr*7, resulting ciphertext, and plaintext, respectively to the user at the end of its execution. The implementation of the six aforementioned commands is omitted due to page limit.

We now state and prove that *PCP* realizes *FCP* using standard cryptographic assumptions. We use the Decisional Diffie-Hellman (DDH) assumption [43], [44] to prove that

*PCP* ≤ *FCP* such that the simulator provides *k* for uncorrupted 2MA-ECDH key and *k* for unverified 2MA-ECDH key. To capture the expected use of *PCP/FCP* as a function of our key bootstrapping protocol and secure communication protocol (see Section V), we slightly restrict the environment not to

# TABLE II



cause commitment problem [32] and key cycles [33], [34], are satisfied by the environment in the theorem below, and say that the environment is used-order respecting and we introduce a machine *M*∗ that is situated between the well-behaved. To ensure that these restrictions and conditions environment and I/O interface of *PCP/FCP*. If any condition is

TABLE III



violated at some point, the protocol run is stopped and blocked by *M*∗ and no further communication can take place. We now obtain the following theorem:

*Theorem 2: Let MAC be a MAC scheme, authenc,*

*unauthenc, pub be encryption schemes, GenDP be an algorithm as above. Let F* *and F* *be families of pseudo random functions for GenDP, and F* *be a family of pseudo random functions. Let PCP be parameterized with these algorithms. Then,*

## M∗|PCP ≤ M∗|FCP (1)

if *MAC* is UF-CMA secure, *authenc* is IND-CPA and INT-CTXT secure, *unauthenc* and *pubenc* are IND-CCA2, GenDP always output random form primes for field order *p* and (ECDH) groups with *n* ≥ 2, and DDH assumption holds true for GenDP.

*Proof Sketch:* This proof involves several hybrid systems as follows: In step one, we define a hybrid system *PCP*1 where all asymmetric operations and generation of nonce/confidential nonce are replaced with their ideal versions. In step two, we define a hybrid system *PCP*2 where real identity value and private key handlings are replaced with their ideal versions. In step three, we define a hybrid system *PCP*3 where real 2MA-ECDH key generation is replaced with its ideal version, nevertheless, guessing or collisions of keys are not prevented. In step four, we define a hybrid system *PCP*4 where real symmetric operations and key derivations are replaced with their ideal versions, and guessing and collisions of keys are prevented. In step five, we replace real MACs operations with their ideal versions. We conclude the proof by combining all the five steps and present that the simulator is responsive.

### V. IDEAL FUNCTIONALITY FOR KEY BOOTSTRAPPING AND SECURE COMMUNICATION

In this section, we present our ideal functionality for Key Bootstrapping (KB) and Secure Communication (SC) denoted by *FSCK B*. The functionality *FSCK B* is used for the KBP and SCP (see below). The main idea of *FSCK B* is that users can send requests to *FSCK B* to start a key bootstrapping by exchanging session keys and further use the exchanged keys for secure communication. This functionality can be used for both key bootstrapping and secure communication, and it allows us to model the validity and expiration of exchanged sessions which are important to the Energy Internet. Formally, *FSCK B* is a machine with two I/O tapes, one network tape, and additional two I/O tapes that connects to *FCP*, which is used as a function

for cryptographic operations by *FSCK B*. We parameterize *FSCK B* with different types of keys *t* ∈ {*pre* − *key,authenc* − *key,unathenc* − *key,mac* − *key*} that help to determine the key generated after a successful key bootstrapping.

In *FSCK B*, a key bootstrapping user can also be identified by a tuple *(I D,lsid,r)*, where *I D* is a user identifier, *lsid* is a local session *id* selected and managed by the protocol and *r* specifies the role of the user, where *r* can either be an initiator *A* or responder *B* of a key bootstrapping and/or secure communication. We note that, for any key bootstrapping protocol with a server role, *FSCK B* can be extended with a role of the server *C* in the protocol (see Section VIII).

Like *FCP*, every message from/to any I/O tapes are prefixed with *(I D,lsid)* in *FSCK B*. Furthermore, *FSCK B* maintains several states for key bootstrapping and secure communication. These states include *restricted*, *startedK B*, *startedSC*, *closedSC*, and *corrupted*. The initial state of every user is stored as *restricted*. The operations provided by *FSCK B* to a key bootstrapping protocol (see Section V.A) as well as a secure communication protocol (see Section V.B) are as follows:

* A user *(I D,lsid,r)* with state *restricted* can start a key bootstrapping by sending a message *m* =

*(*InitializeKB*, I D,m)*, where *I D* is the user identifier of the intended responder, and *m* is a random bit string that might be used by the key bootstrapping protocol. Upon receiving this message, *FSCK B* verifies the users’ identities *I D* and *I D*. We note that as the initialization message *m* is not digitally signed since the user *(I D,lsid,r)* in this state *restricted* does not have access to any cryptographic operation, prefixing every message with *(I D,lsid)* plays an important role here to prevent any other user from claiming to be *I D*, or *I D* claiming to be another user, say *I D*. If these verifications succeed, *FSCK B* sets *state(I D,lsid,r)* = *startedK B*, sets *Responder(I D,lsid,r)* = *I D*, and forwards the message *(m,(I D,lsid,r))* to the adversary. In this case, the states of the initiator and responder become *startedK B*.

* A user *(I D,lsid,r)* with state *startedK B* can use *FSCK B* to access the following commands of *FCP*: i) GetDP; ii) GenIV; iii) Gen2MA-ECDHKey; iv) Confirm2 MA-ECDHKey; v) GenPSKeyI; vi) GenPSKeyII; vii) GenCNonce; viii) DeriveMulticastKey; ix) DeriveSSKey; x) UEnc; xii) UDec; xii) MAC; and xiii) VMAC. In this case, *FSCK B* forwards all responses received from *FCP* to the user, and keeps track of keys and pointers available to the user. Once secret session keys have been derived by the user, the state of the user becomes *startedSC*.
* A user*FSCK B* to access the DeriveMulticastKey command of*(I D,lsid,r)* with state *startedSC* can still use*FCP*.

Again, *FSCK B* forwards *FCP*’s response to the user and keeps track of the multicast session key and its pointer.

* A user*FCP (I DK B,lsid*for the following operations: i) UEnc;*,r)* with state *startedSC* can access via *FSC*

ii) UDec; iii) MEnc; iv) MDec; v) MAC; and vi) VMAC. Since the user is in possession of symmetric keys, this makes it easier for her to use the keys for exchange of messages.

• A user*FSCK B* to close her session. Upon receiving this request,*(I D,lsid,r)* with state *startedSC* can request

*FSCK B* forwards the request to the adversary via a restricting message, and returns the adversary’s response *Okay* to the user. Now, the state of the user becomes *closedSC* and thus the user losses access to all keys, pointers, and cryptographic operations.

We now model corruption in *FSCK B*. Note that the adversary can only corrupt the user in the states *restricted* and *closedSC* since the user has no access to any keys, pointers, or cryptographic operations. Since the adversary does not have access to any keys after KB/SC, this shows that *FSCK B* models perfect forward secrecy. The adversary may decide to corrupt the user with state *closedSC*. In this case, *FSCK B* sets the state of the user as *corrupted*. Thus, we restrict

|  |
| --- |
|  |
|  |

Fig. 3. Key bootstrapping protocol.

this corruption model to make *FSCK B* easier to use by the key bootstrapping and secure communication protocols.

1. *Key Bootstrapping Protocol*

We propose our KBP with mutual authentication, data confidentiality, and data integrity as depicted in Figure 3. We show that this protocol uses *FCP* to perform all cryptographic operations and it realizes *FSCK B* and it is based on the 2MA-ECDH key exchange. To model the initiator *A* and responder *B* of our protocol, we use two machines

*MA* and *MB* for initiator and responder, respectively. Each of these machines has a network tape and they provide the same I/O interface as *FSCK B*. Furthermore, they use *FCP* as a function to perform all cryptographic operations. In the run of KBP, there is only one instance of *MA/MB* for every user *(I D,lsid)* which executes the protocol from Figure 3. At the end of the protocol, the instances compute a 2MAECDH key *k* from *QA* and *QB*, and use *k* to derive preshared keys *k(t*1*)* and *k(t*2*)*, multicast session key *kSEU(t)*, and secret session key *k(t)*. Pointers to these keys are delivered by these instances, which further allow a user to use *FCP* to perform cryptographic operations (with the keys) in an ideal manner.

1. *Secure Communication Protocol*

The SCP with data confidentiality, data integrity, and data initiator authentication is depicted in Figure 4. It uses the derived session keys that have been exchanged by KBP to establish secure communication between users. More precisely, we have unicast and multicast communications which use secret session keys and multicast session keys, respectively, for secure communication in the Energy Internet. A session key can be used for both encryption and MAC. A session key in SCP can be deployed for encrypt-then-MAC to provide confidentiality, integrity, and authentication during communication in the Energy

Internet.

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |

Fig. 4. Secure communication protocol.

### VI. SECURITY ANALYSIS

In this section, we first analyze the key bootstrapping and secure communication protocols using our developed ideal functionalities *FCP* and *FSCK B* in the previous sections. Then, we prove that KBP and SCP are secure universally composable protocols. Due to the use of *FCP/PCP*, the proofs of these protocols require only information theoretic arguments (i.e., without reduction proof or probabilistic reasoning).

*A. Security Analysis of Key Bootstrapping Protocol*

We provide a theorem that states that the KBP is a secure universally composable key bootstrapping protocol. Session keys returned by the KBP can be used in an ideal manner.

*Theorem 3: Let MA and MB be machines that model the*

*Key Bootstrapping Protocol (KBP) as described above. Let FCP and FCP* *be two versions of the ideal functionality for cryptographic operations with the same parameters, and let FSCK B be the ideal functionality for key bootstrapping with parameters t* = *authenc* − *key and t*= *M AC* − *key. Then the following holds true:*

## MA|MB|FCP ≤ FSCK B|FCP (2)

*Proof Sketch:* We say that a user *I D* is corrupted if its encryption key and/or MAC key is corrupted. Also, we say that an instance is corrupted if it returns *Yes* when requested for its corruption status by the environment. We define a responsive simulator *S* and show that *E*|*MA*|*MB*|*FCP* ≡ *E*|*S*|*FSCK B*|*FCP* for all responsive environments *E* ∈ *Env(MA*|*MB*|*FCP)*. We note that *S* fulfils the runtime conditions and responds to restricting messages instantly as long as *E* also does the same with overwhelming probability. The protocol *MA*|*MB*|*FCP* is simulated by *S* and *S* keeps track of corruption statuses of instances belonging to the users in *FSCK B*, and synchronizes the simulated instances of *MA/MB*. To initialize *FCP*, *S* sends a message to *FCP* and receives the domain parameters *(p,a,b, G,n,h)* which is used for simulating *FCP*. Then, *S* requests for the cryptographic algorithms from the environment and once received, it forwards the algorithms to *FCP* . If *FSCK B* specifies that a user *(I D,lsid,r)* has started a key bootstrapping, *S* updates its internal simulation as a result of this. If an identity value *ivB*, nonce *ncB*, and public key *QB* are accepted by an uncorrupted instance *(I DA,lsidA, A)*, then *FSCK B* is instructed by *S* to create a key bootstrapping session from *(I DA,lsidA, A)* and instance *(I DB,lsidB, B)* that computed the encryption and MAC in the second message of the protocol. Then, the function *FCP* of *FSCK B* will request *S* to provide the value of the session key. Now, *S* instructs *FSCK B* to output the pointer of the secret session key for *(I DA,lsidA, A)*. *FCP* may request *S* to provide another session key when an initial session key is in used. Then, *S* uses *FCP* to simulate the operation and forwards the keys to *FCP* . We note that sets of keys in *FCP* and *FCP* are synchronized. While *FCP* can be used for simulating operations and storing keys, *FCP* can only be used for storing keys, thus, all keys accepted by *FCP* will also be accepted by *FCP* . If an instance notifies *S* that she has closed her session, *S* updates its internal simulation as a result of this, and returns *Okay* to the instance. Due to the restricting messages as presented in Section IV, all operations performed by *FCP* are always successful and have no side effects on *MA* or *MB*.

We argue that the simulation is perfect using a case of uncorrupted initiator instances during key bootstrapping. Let *(I DA,lsidA, A)* be an uncorrupted instance of *MA* that wants to perform key bootstrapping with a user *I D*. This instance will use *FCP* for cryptographic operations (as provided in Section IV). We have to show that *S* finds a responder instance that can be paired with this instance. If *(I DA,lsidA, A)* outputs a pointer of a secret session key, this means that it has already accepted the second message of the KBP and the encryption key and MAC key of the user *I D* is still uncorrupted. This shows that a message *m* = *UM AC(k(t*2*),(ivA,ivB,ncB, QA, QB)k(t*1*)))* is encrypted and MACed by some instance belonging to *I D*, say *(I D,lsid,r)*, where *QA* = *dA.G* and *QB* = *dB.G*, thus this instance is uncorrupted. Furthermore, since *(I D,lsid,r)* considers *I DA* to be her key bootstrapping partner as recognized in the encryption and MAC, this shows that the instance *(I D,lsid,r)* is not corrupted and thus the instance *(I D,lsid,r)* is a responder instance. We now argue that *r*= *B*: Suppose by contradiction *r*= *A*, this means that the second message of the protocol was accepted by this instance. However, as *dA/QA* is ideally computed, this shows that there is only one instance that would encrypt and MAC such a message, namely *(I DA,lsidA, A)*, which does not output any encryptions or MACs before accepting the second message of the protocol. Thus, this indicates that *r*= *B*.

We note that, other cases such as uncorrupted responder instances during key bootstrapping, uncorrupted instances after key bootstrapping, and corrupted instances are omitted due to page limit.

By Theorem 2, we can now replace *FCP* by its realization *PCP* which yields that the KBP is a secure universally composable key bootstrapping protocol.

*Proposition 1: Let MA, MB as defined above. Let FCP,*

*PCP, and M*∗ *be as in Theorem 2. Then the following holds true:*

## M∗|MA|MB|PCP ≤ M∗|FSCK B|FCP (3)

*Proof:* It can be seen easily from Theorem 1, Theorem 2, and Theorem 3 that *M*∗|*MA*|*MB* creates a well-behaved environment when combined with *E*.

*B. Security Analysis of Secure Communication Protocol*

We model the initiator and responder of the SCP as well as the responsive simulator in a similar way to the KBP, except that in this case, the machines are used for secure communication. At the end of this protocol, the instances exchange messages in an ideal manner. The following theorem states that the SCP is a secure universally composable secure communication protocol.

*Theorem 4: Let MA and MB be machines that model the Secure Communication Protocol (SCP) as described above. Let FCP and FCP* *be two versions of the ideal functionality for cryptographic operations with the same parameters, and let FSCK B be the ideal functionality for secure communication with parameters t* =*secret session key and t*=*multicast session key. Then the following holds true:*

## MA|MB|FCP ≤ FSCK B|FCP (4)

*Proof Sketch:* The proof of this theorem does not require any reduction proof or probabilistic reasoning. Just as the KBP, by Theorem 2, we can replace *FCP* by its realization

*PCP* which yields that SCP is a secure universally composable secure communication protocol.

*Proposition 2: Let MA, MB as defined above. Let FCP, PCP, and M*∗ *as in Theorem 2. Then the following holds true:*

## M∗|MA|MB|PCP ≤ M∗|FSCK B|FCP (5)

*Proof:* It can be seen easily from Theorem 1, Theorem 2, and Theorem 4 that *M*∗|*MA*|*MB* creates a well-behaved environment when combined with *E*.

### VII. EXPERIMENTAL SETUP

In this section, we first analyze the performance of the proposed key bootstrapping protocol, and then compare the protocol with existing related protocols in terms of computational and communication costs. Lastly, we evaluate the End-to-End Delay (EED) and reliability of our protocols to show their impacts on the smart grid. We note that, we utilize Figure 2 in Section II to show that both unicast and multicast communications meet the latency and reliability requirements of the smart grid. The cryptographic algorithms employed in this analysis are hash function SHA-256, digital signature ECDSA-160, symmetric encryption AES-128, and public key encryption (ECC) ElGamal-160.

To estimate the performance of various protocols, we use wireless sensor modules, i.e., Tmote Sky motes [35], to implement the protocols through their cryptographic operations. The representation and computation times of the cryptographic operations are presented in Table IV. These cryptographic operations are compiled under an operating system, TinyOS [36], conducted on Intel(R) Core(TM) i5-6500 CPU@3.20GHz with 16 GB RAM Windows 7 System, and executed on the Tmote Sky motes with their programs

# TABLE IV

REPRESENTATION AND COMPUTATION TIMES

OF CRYPTOGRAPHIC OPERATIONS

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# TABLE V

COMPARISON OF COMPUTATIONAL COSTS AND TIMES

OF VARIOUS PROTOCOLS

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

written in nesC language [37]. We note that, the implementation of our protocols on the Tmote Sky motes provides insight about the actual deployment of the protocols. Depending on the smart grids requirements such as computation, communication, and sensing, Tmote Sky motes provide increased automation in smart grid applications such as metering application (see, e.g., [41]).

*1) Computational Cost:* This cost is defined as the number of cryptographic operations and time required for the execution of key bootstrapping. We compare the computational cost for the key bootstrapping protocol with that of Nicanfar *et al.* [13], Mohammadali *et al.* [14], Wu and Zhou [17], and Nicanfar *et al.* [19]. The computation time of selecting random numbers *(Trn)* and modular multiplication operation *(Tmo)* is negligible when compared to the computation time of performing other cryptographic operations [14]. In our protocol, the computational cost required for each initiator (component *I DA*) and responder (component

*I DB*) is 4*Trn* + 2*Tem* + 4*Th* + *Tsd* + *Thm* ≈ 2*.*1366*s* and 4*Trn* + 2*Tem* + 4*Th* + *Tse* + *Thm* ≈ 2*.*1366*s*, respectively, and the *DI RC* (or say, a Trusted Authority *(T A)*) is not involved. As shown in Table V, it can be observed that our protocol reduces the amount of point multiplication operation and

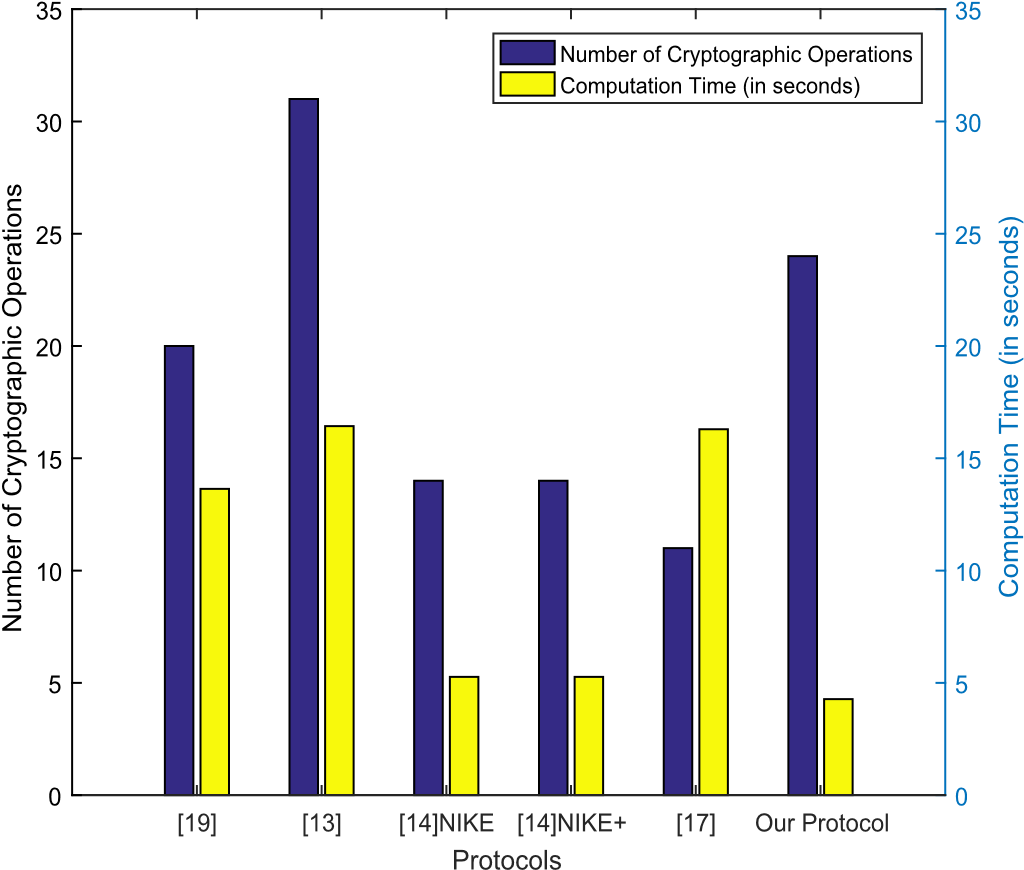


Fig. 5. Computational cost of various protocols.

hash function operation compared to Nicanfar et al.’s [13] protocol and Mohammadali et al.’s [14] protocol, respectively. Furthermore, our protocol requires less *Tse/Tsd*, *Tem*, and

*Th/Tse/Tsd* compared to that of Nicanfar et al. [19], Mohammadali et al. [14], and Nicanfar et al. [13], respectively. Hence, the computation time, which represents the time for calculating cryptographic operations, of our protocol is minimal compared to the other related protocols. We note that the CMAC computation time is equivalent to symmetric encryption for the same algorithm [38]. Furthermore, the computation time at *I DA* is equivalent to the computation time at *I DB* in our protocol thereby providing users with equal amount of computation time during key bootstrapping.

As shown in Figure 5, the overall computational cost of our scheme is more efficient than the protocols in [13], [14], [17], and [19], as the total number of cryptographic operations and computation time of our protocol is 24*/*4*.*2732*s*, while it is 20*/*13*.*6353*s*, 31*/*16*.*4327*s*, 14*/*5*.*2641*s*, 14*/*5*.*2641*s*, and 11*/*16*.*2954*s* for the protocols in [13], [14], [19] (NIKE), [14] (NIKE+), and [17], respectively. We note that our protocol requires the least computational cost due to the use of many low computation time cryptographic operations such as *Trn* and *Th*.

*2) Communication Cost:* This cost is defined as the number and size of messages that need to be transmitted during key bootstrapping. The proposed protocol requires a total of seven messages. Table VI provides the communication costs of all the related protocols. Mohammadali et al.’s [14] NIKE protocol requires seven messages, while their NIKE+ protocol require six messages. Our protocol has the lowest communication cost in terms of the amount of transmitted information. To estimate the number and size of transmitted messages required in our protocol and other related protocols, we provide the length of various binary sequences as follows: i) random number/nonce32 bits; ii) identity value-32 bits; iii) hash function/MAC256 bits; iv) symmetric encryption-128 bits; v) public key

# TABLE VI

COMPARISON OF COMMUNICATION COSTS OF VARIOUS PROTOCOLS

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

TABLE VII SIMULATION PARAMETERS

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

encryption-160 bits; and vi) user identity-160 bits. In our protocol, *mii* = *ivA,ivB,ncB, QA, QB* =416 bits, *mi* = *U Enck(t*1*)(mii)* =544 bits, and *m* =

*UM ACk(t*2*)(mi)* =256 bits. The communication cost (in terms of the number of transmitted bits) required in our protocol is given as follows:

i) *Message*1 = *User A* → *User B* =224 bits; and ii) *Message*2 = *User B* → *User A* =990 bits. Thus, the total number of required binary symbols to be transmitted is 1,214 bits. Nicanfar *et al.* [19], Nicanfar *et al.* [13], Mohammadali *et al.* [14] (NIKE), Mohammadali *et al.* [14] (NIKE+), and Wu and Zhou [17] require approximately 3,936 bits, 2,816 bits, 2,526 bits, 2,400 bits, and 2,880 bits, respectively. Table VI also shows the required number of transmitted bits for all protocols. Thus, we can observe that our protocol provides the least communication bits when compared to the other related protocols.

1. *End-to-End Delay (EED):* We assume the benchmark range for the latency requirements of smart grid applications and technologies to be from 1*ms* to 1*min* as presented in Section II. The Energy Internet network was simulated by using NS-3 and modeled as UDP/IP network for unicast/multicast communication. The EED is given by

*ni*=1 *TZi* +*mj*=1 *TPj* +*kn*=1 *T*prepare the packet including*Zk* , where *TZi* is the packetization delay (i.e., time taken to addition of security tag(s), *TPj* is the propagation delay (i.e. time taken to deliver the message by the network), and *TZk* is de-packetization delay. Details of the simulation parameters used in NS-3 are provided in Table VII where the selection of the number of users/components were carried out without loss of generality. Additionally, we consider other standard parameters from the NS-3. According to the Table VII, we have scenarios 1, 2, and 3 for the KBP, SCP (unicast), and SCP (multicast - with 100 components) which illustrates our protocols. Figure 6 shows the EED (in seconds) of our

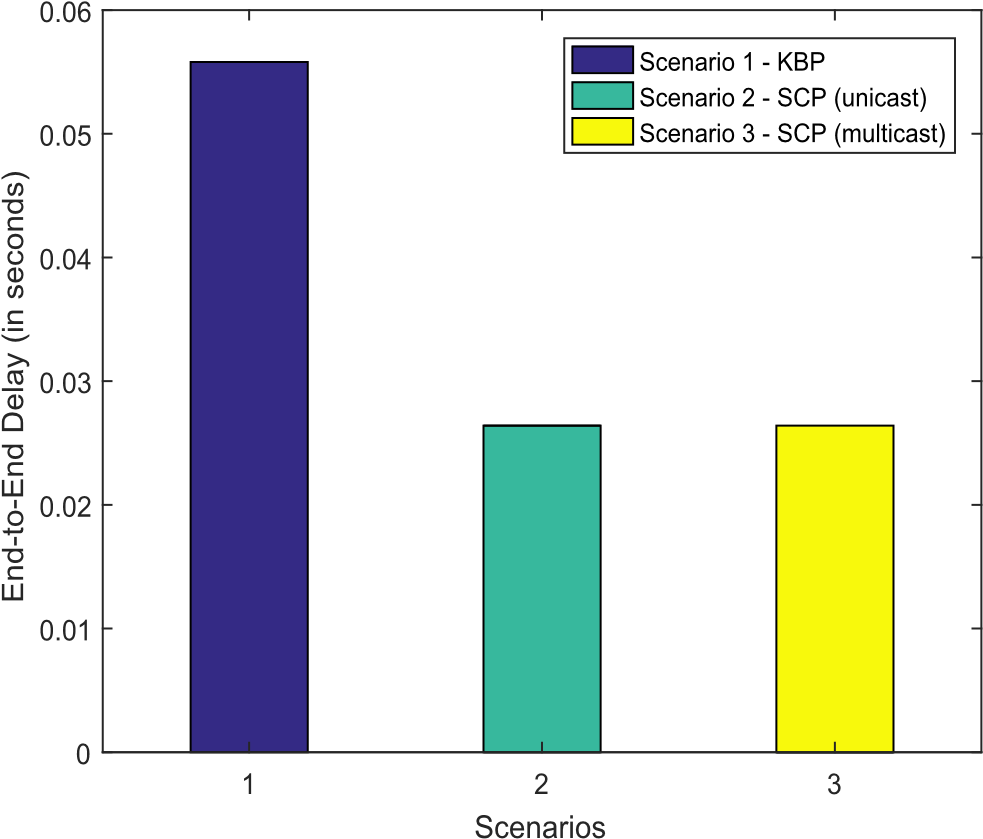


Fig. 6. End-to-end delay of our protocols. protocols under the three scenarios. The EED values are given as 0*.*0558*s*, 0*.*0264*s*, and 0*.*0264*s* for scenarios 1, 2, and 3, respectively, where the number of transmitted/received packets are 66*/*66, 16*/*16, and 16*/*16 for scenarios 1, 2, and 3, respectively. We note that as the number of transmitted/received packets decreases across the above scenarios, the EED value decreases as depicted in Figure 6. We can see that the EED values for scenarios 1, 2, and 3 are below our maximum 1*min* target of some of the latency requirements of smart grid applications and technologies such as advanced metering infrastructure, electric transportation, demand response, and distribution automation with latencies of 2*s* to 15*s* [23], 2*s* [23], *<* 1*min* [24], and *<* 5*s* [24], respectively, and are consistent with the theoretical performance evaluation. Furthermore, the EED value for each comoponent in different scenarios 2 and 3 is the same. This is because: 1) the scenarios use the same *ZP*, *TZi* , and *TZk* ; and 2) all the components are in the same group or network that requires no receiver to act as a route to other receivers.

1. *Reliability:* In this paper, the target reliability for the smart grid applications and technologies is 99*.*99% with latency of *<* 0*.*06*s*. The reliability, which represents the probability that the *ZP* of each of the scenarios 1, 2, and 3 are successfully transferred within *<* 0*.*06*s* stipulates that the packets are successfully delivered and the latency is satisfied in our simulation. In scenarios 1, 2, and 3, we note that: i) the total number of bits and packets sent is equal to the total number of bits and packets received, respectively; and ii) no packet was lost during transmission. Thus, the reliability of each of the scenarios 1, 2, and 3 is given as ≥ 99*.*99%, which meets/exceeds the above target. We can see that the reliability of each of the scenarios 1, 2, and 3 meets the 99*.*99% reliability of some of the smart grid applications and technologies such as advanced metering infrastructure, electric transportation, demand response, and distribution automation [23], [24].

VIII. CASE STUDY

In this section, we carry out a case study to demonstrate the usefulness of our functionalities. We analyze one of

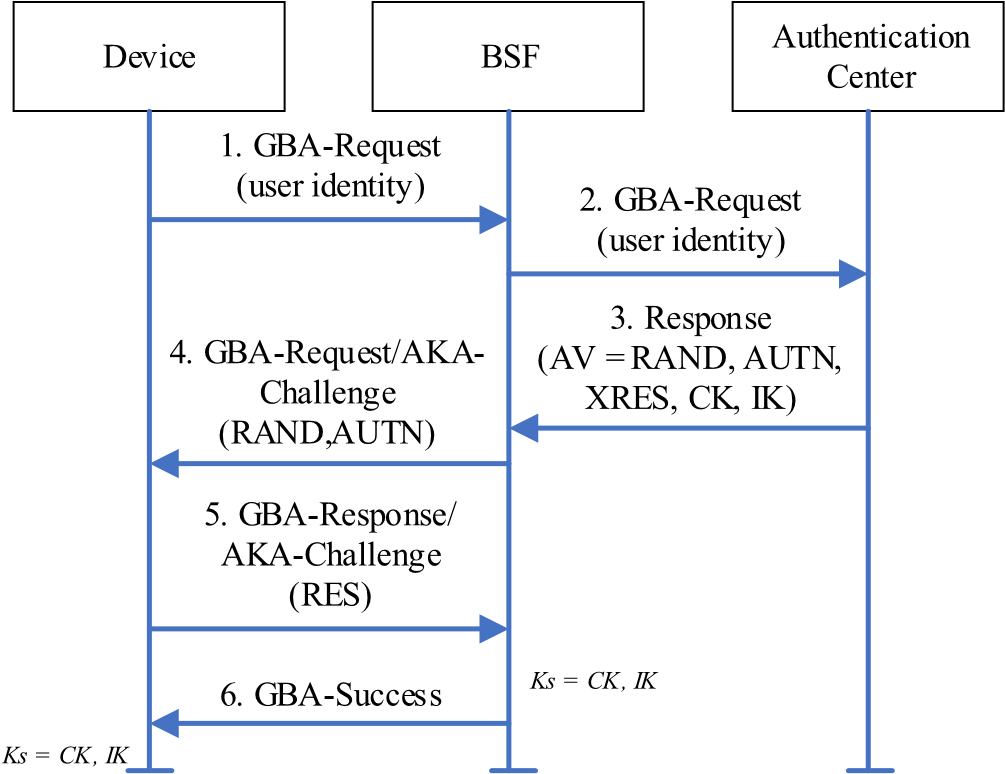


Fig. 7. The bootstrapping procedure of the GBA.

real-world key bootstrapping protocols, namely the GBA based on Authenticated and Key Agreement (AKA) protocol with mutual authentication and data integrity [30], and point out some weaknesses in its key bootstrapping procedure. The GBA consists of two main components, namely: the Bootstrapping Server Function (BSF) and the Network Application Function (NAF). In Figure 7, we show the bootstrapping procedure of the GBA, where a device is authenticated by the BSF using the AKA protocol which leads to the generation of a bootstrapping session key. The device uses this key as a root key or preshared key to generate application-specific session keys for secure communications and accessing available services at the NAF. We note that the random challenge *RAND*, authentication token *AUT N*, expected response *X RES*, encryption key *CK*, and integrity key *I K* in Figure 7 can be regarded as a nonce, confidential nonce, message, pre-shared key I, and preshared key II, respectively, in *FCP* without loss of generality. We leave a formulation of *FCP* based on the authentication vector *AV* for future work.

The bootstrapping protocol of the GBA does not realize *FSCK B*. To show this, we consider the following setting: We use three machines *MA*, *MB* and *MC* to model the initiator, responder, and authentication center roles, respectively, to bootstrap a key. An honest initiator outputs a bootstrapping session key, which was computed from its calculated *(CK, I K)*. The responder instance that sent *(RAND, AUT N)* might have received a different *AV*, say *(AV* = *RAND, AUT N, X RES,CK, I K)*, in the third message of the protocol since the authentication center is not authenticated by the responder. If *AV*  was not honestly generated by *FCP*, then *(CK, I K)* will be marked as unverified after the computation of *Ks* because there is no guarantee from the DDH assumption that an attacker does not learn anything from *(CK, I K)*. As *(CK, I K)* are marked as unverified, the key *(Ks* = *CK, I K)* and all session keys derived from *Ks* will also be marked as unverified. Thus, there is no security guarantee for the *(RAND, AUT N)* and an attacker can easily let the responder and initiator instances to accept the *(RAND, AUT N)*.

Furthermore, we argue that the bootstrapping procedure of the GBA is vulnerable to active man-in-the-middle attacks and does not support/offer the following features: i) multicast key bootstrapping; and ii) data integrity and data confidentiality. To support our arguments, we consider the setting where an honest authentication center outputs *(AV* = *RAND, AUT N, X RES,CK, I K)*, which was returned after verifying the initiator’s identity. The responder instance that sent part of the *AV*, i.e., *(RAND, AUT N)* in the fourth message of the protocol received the *AV* without the support of any security features such as MAC and/or encryption, and a bootstrapping session key *Ks* is derived by the originator and responder instances using *(CK, I K)*, which are also part of the *AV*. Thus, we have no security guarantee for the messages and key since MAC/encryption and hash function, respectively, are not used and an attacker can easily allow the initiator and responder instances accept the messages and key.

To fix these problems in our setting, we model GBA in the same way as the KBP, except that we have three machines *MA*, *MB*, and *MC* executing the GBA from Figure 7. The fixes for the problems are given as follows: i) establish mutual authentication between the BSF and the authentication center; ii) enhance the third, fourth, and fifth messages of the protocol with the MAC/UEnc, MAC, and MAC/UEnc commands, respectively to mitigate active man-in-the-middle attacks, and provide data integrity and data confidentiality; iii) equip the GBA with the DeriveMulticastKey to support multicast key bootstrapping; and iv) equip the bootstrapping key with GenPSKeyI/GenPSKeyII commands to provide universally composable security guarantees for the key. Thus, using *FSCK B* provides cryptographic soundness and key bootstrapping enhancement to the GBA. The following theorem states that this enhancement is a secure universally composable key bootstrapping protocol.

*Theorem 5: Let MA, MB, and MC be machines that model the Generic Bootstrapping Architecture (GBA) enhancement as described above. Let FCP and FCP* *be two versions of the ideal functionality for cryptographic operations with the same parameters, and let FSCK B be the ideal functionality for key bootstrapping with parameters t* = *authenc* − *key and t*= *M AC* − *key. Then the following holds true:*

## MA|MB|MC|FCP ≤ FSCK B|FCP (6)

*Proof Sketch:* The proof of this theorem require only information theoretic arguments. Just as the KBP, by Theorem 2, we can replace *FCP* by *PCP* which yields that the GBA enhancement is a secure universally composable key bootstrapping protocol.

*Proposition 3: Let MA, MB, MC as defined above. Let FCP, PCP, and M*∗ *as in Theorem 2. Then the following holds true:*

## M∗|MA|MB|MC|PCP ≤ M∗|FSCK B|FCP (7)

### IX. CONCLUSION

In this paper, we have proposed key bootstrapping and secure communication protocols for the Energy Internet. The key bootstrapping protocol provides session keys that support the secure communication protocol, which handles unicast and multicast communication. We proposed an ideal functionality *FCP* that models various key bootstrapping and secure communication cryptographic operations which can be used in the Energy Internet in an ideal manner. We also provided an ideal functionality *FSCK B* for key bootstrapping and secure communication which supports the properties of *FCP*. Specifically, these functionalities allow the designing and analyzing of the key bootstrapping and secure communication protocols. Compared with the existing related protocols, our protocols provide security in arbitrary environments and offer strong universal composable security guarantees. At the same time, our key bootstrapping protocol incurred lower costs for computation and communication which improves key bootstrapping in the smart grid, and both the key bootstrapping and secure communication protocols meet the latency and reliability requirements of the smart grid. Furthermore, our protocols can be used and implemented in any application and environment outside of the Energy Internet. We have illustrated the usefulness of our functionalities in a case study, where we uncovered some weaknesses in the key bootstrapping procedure of the Generic Bootstrapping Architecture and provided an enhancement using our functionalities. In future work, we will extend our functionalities to include cryptographic operations for access control and accountability to enhance practical key bootstrapping and secure communication protocols.

# REFERENCES

1. A. Q. Huang, M. L. Crow, G. T. Heydt, J. P. Zheng, and S. J. Dale, “The future renewable electric energy delivery and management (FREEDM) system: The energy Internet,” *Proc. IEEE*, vol. 99, no. 1, pp. 133–148, Jan. 2011.
2. P. Yi, T. Zhu, B. Jiang, R. Jin, and B. Wang, “Deploying energy routers in an energy Internet based on electric vehicles,” *IEEE Trans. Veh. Technol.*, vol. 65, no. 6, pp. 4714–4725, Jun. 2016.
3. C.-S. Park and J.-H. Lee, “Security bootstrapping for secure join and binding on the IEEE 802.15.4-based LoWPAN,” *IEEE Internet Things J.*, vol. 4, no. 4, pp. 996–1005, Aug. 2017.
4. Z. Shi, B. Zhang, and D. Feng, “Practical-time related-key attack on Hummingbird-2,” *IET Inf. Secur.*, vol. 9, no. 6, pp. 321–327, 2015.
5. Y. Xu, J. Zhang, W. Wang, A. Juneja, and S. Bhattacharya, “Energy router: Architectures and functionalities toward energy Internet,” in *Proc. IEEE Int. Conf. Smart Grid Commun. (SmartGridComm)*, Brussels, Belgium, Oct. 2011, pp. 31–36.
6. V. Kounev, D. Tipper, A. A. Yavuz, B. M. Grainger, and G. F. Reed, “A secure communication architecture for distributed microgrid control,” *IEEE Trans. Smart Grid*, vol. 6, no. 5, pp. 2484–2492, Sep. 2015.
7. Y. Wang, T. T. Gamage, and C. H. Hauser, “Security implications of transport layer protocols in power grid synchrophasor data communication,” *IEEE Trans. Smart Grid*, vol. 7, no. 2, pp. 807–816, Mar. 2016.
8. R. Canetti, “Universally composable security: A new paradigm for cryptographic protocols,” in *Proc. 42nd IEEE Symp. Found. Comput. Sci. (FOCS)*, Oct. 2001, pp. 136–145.
9. R. Kusters, “Simulation-based security with inexhaustible interactive turing machines,” in *Proc. 19th IEEE Workshop Comput. Secur. Found.*, Jul. 2006, pp. 308–320.
10. U. Maurer, “Constructive cryptography—A new paradigm for security definitions and proofs,” in *Theory of Security and Applications*. Berlin, Germany, Springer, 2012, pp. 33–56.
11. D. Hofheinz and V. Shoup, “GNUC: A new universal composability framework,” *J. Cryptol.*, vol. 28, no. 3, pp. 423–508, 2015.
12. V. S. Miller, “Use of elliptic curves in cryptography,” in *Proc. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 1985, pp. 417–426.
13. H. Nicanfar, P. Jokar, K. Beznosov, and V. C. M. Leung, “Efficient authentication and key management mechanisms for smart grid communications,” *IEEE Syst. J.*, vol. 8, no. 2, pp. 629–640, Jun. 2014.
14. A. Mohammadali, M. S. Haghighi, M. H. Tadayon, and

A. Mohammadi-Nodooshan, “A novel identity-based key establishment method for advanced metering infrastructure in smart grid,” *IEEE Trans. Smart Grid*, vol. 9, no. 4, pp. 2834–2842, Jul. 2018.

1. Y. J. Kim, J. Lee, G. Atkinson, H. Kim, and M. Thottan, “SeDAX: A scalable, resilient, and secure platform for smart grid communications,” *IEEE J. Sel. Areas Commun.*, vol. 30, no. 6, pp. 1119–1136, Jul. 2012.
2. E. Ancillotti, R. Bruno, and M. Conti, “The role of communication systems in smart grids: Architectures, technical solutions and research challenges,” *Comput. Commun.*, vol. 36, nos. 17–18, pp. 1665–1697, Nov./Dec. 2013.
3. D. Wu and C. Zhou, “Fault-tolerant and scalable key management for smart grid,” *IEEE Trans. Smart Grid*, vol. 2, no. 2, pp. 375–381, Jun. 2011.
4. Z. Wan, G. Wang, Y. Yang, and S. Shi, “SKM: Scalable key management for advanced metering infrastructure in smart grids,” *IEEE Trans. Ind. Electron.*, vol. 61, no. 12, pp. 7055–7066, Dec. 2014.
5. H. Nicanfar, P. Jokar, and V. C. Leung, “Smart grid authentication and key management for unicast and multicast communications,” in *Proc.*

*IEEE PES Innov. Smart Grid Technol. Asia (ISGT)*, Nov. 2011, pp. 1–8.

1. V. Odelu, A. K. Das, M. Wazid, and M. Conti, “Provably secure authenticated key agreement scheme for smart grid,” *IEEE Trans. Smart Grid*, vol. 9, no. 3, pp. 1900–1910, May 2018.
2. G. Indra and R. Taneja, “A time stamp-based elliptic curve cryptosystem for wireless ad-hoc sensor networks,” *Int. J. Space-Based Situated Comput.*, vol. 4, no. 1, pp. 39–54, 2014.
3. P. Kansal and A. Bose, “Bandwidth and latency requirements for smart transmission grid applications,” *IEEE Trans. Smart Grid*, vol. 3, no. 3, pp. 1344–1352, Sep. 2012.
4. *Communications Requirements of Smart Grid technologies*, U.S. Dept. Energy, Washington, DC, USA, 2010, pp. 1–69.
5. K. C. Budka, J. G. Deshpande, T. L. Doumi, M. Madden, and T. Mew, “Communication network architecture and design principles for smart grids,” *Bell Labs Tech. J.*, vol. 15, no. 2, pp. 205–227, Sep. 2010.
6. S. Fuloria, R. Anderson, K. McGrath, K. Hansen, and F. Alvarez, “The protection of substation communications,” in *Proc. SCADA Secur. Sci. Symp.*, 2010, pp. 1–13.
7. M. R. Clark and K. M. Hopkinson, “Transferable multiparty computation with applications to the smart grid,” *IEEE Trans. Inf. Forensics Security*, vol. 9, no. 9, pp. 1356–1366, Sep. 2014.
8. R. Küsters and M. Tuengerthal, “The IITM model: A simple and expressive model for universal composability,” Cryptol. ePrint Archive, Tech. Rep. 2013/025, 2013. [Online]. Available: http://eprint.iacr.org/2013/025
9. J. Camenisch, R. R. Enderlein, S. Krenn, R. Küsters, and D. Rausch, “Universal composition with responsive environments,” in *Proc. 22nd Int. Conf. Theory Appl. Cryptol. Inf. Secur. Adv. Cryptol. (ASIACRYPT)*, vol. 10032. Springer, 2016, pp. 807–840.
10. R. Küsters and M. Tuengerthal, “Composition theorems without preestablished session identifiers,” in *Proc. 18th ACM Conf. Comput. Commun. Secur.*, Chicago, IL, USA, 2011, pp. 41–50.
11. *Generic Authentication Architecture (GAA); Generic Bootstrapping Architecture (GBA)*, document 3GPP TS 33.220, 3GPP, Accessed: Sep. 2018. [Online]. Available: http://www.3gpp.org/ DynaReport/33220.htm
12. R. Küsters and D. Rausch, “A framework for universally composable Diffie–Hellman key exchange,” Cryptol. ePrint Archive, Tech. Rep. 2017/256, 2017. [Online]. Available: http://eprint.iacr.org/2017/256
13. R. Canetti and M. Fischlin, “Universally composable commitments,” in *Advances in Cryptology—CRYPTO*. Berlin, Germany, Springer, 2001, pp. 19–40.
14. J. Black, P. Rogaway, and T. Shrimpton, “Encryption-scheme security in the presence of key-dependent messages,” in *Proc. 9th Annu. Int. Workshop Sel. Areas Cryptogr.*, 2003, pp. 62–75.
15. M. Backes, B. Pfitzmann, and A. Scedrov, “Key-dependent message security under active attacks-BRSIM/UC-soundness of Dolev–Yaostyle encryption with key cycles,” *J. Comput. Secur.*, vol. 16, no. 5, pp. 497–530, 2008.
16. T. Sky, “Ultra low power IEEE 802.15. 4 compliant wireless sensor module,” Moteiv Corp., Berkeley, CA, USA, 2006.
17. J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, “System architecture directions for networked sensors,” in *Proc. 9th*

*Int. Conf. Architectural Support Program. Lang. Oper. Syst.*, Cambridge, MA, USA, 2000, pp. 93–104.

1. D. Gay, P. Levis, R. von Behren, M. Welsh, E. Brewer, and D. Culler, “The nesC language: A holistic approach to networked embedded systems,” in *Proc. ACM SIGPLAN Conf. Program. Lang. Design Implement.*, San Diego, CA, USA, 2003, pp. 41–51.
2. OpenWrt. (2014). *The OpenWrt Developer Team*. [Online]. Available: https://openwrt.org/
3. R. Küsters and M. Tuengerthal, “Ideal key derivation and encryption in simulation-based security,” in *Proc. 11th Int. Conf. Topics Cryptol. CT-RSA*, San Francisco, CA, USA, 2011, pp. 1–36.
4. S. Raza, S. Duquennoy, J. Höglund, U. Roedig, and T. Voigt, “Secure communication for the Internet of Things—A comparison of link-layer security and IPsec for 6LoWPAN,” *Secur. Commun. Netw.*, vol. 7, no. 12, pp. 2654–2668, 2014.
5. V. C. Gungor, B. Lu, and G. P. Hancke, “Opportunities and challenges of wireless sensor networks in smart grid,” *IEEE Trans. Ind. Electron.*, vol. 57, no. 10, pp. 3557–3564, Oct. 2010.
6. A. R. Al-Ali and R. Aburukba, “Role of Internet of Things in the smart grid technology,” *J. Comput. Commun.*, vol. 3, pp. 229–233, May 2015. [43] D. Boneh, “The decision Diffie–Hellman problem,” in *Proc. Int. Algorithmic Number Theory Symp.* Berlin, Germany: Springer, 1998, pp. 48–63.

[44] M. Abdalla, M. Bellare, and P. Rogaway, “The oracle Diffie–Hellman assumptions and an analysis of DHIES,” in *Proc. Cryptogr. Track RSA Conf.* Berlin, Germany: Springer, 2001, pp. 143–158.