This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSG.2017.2655014, IEEE Transactions on Smart Grid

Efficient Signature Scheme for Delivering Authentic Control Commands in the Smart Grid

Neetesh Saxena, *Member, IEEE,* and Santiago Grijalva, *Senior Member, IEEE*

***Abstract*—Enabling secure delivery of control commands and alert messages is one of the critical requirements of smart grid communications. If adversaries were to gain access, they could alter or send malicious commands from the control center, intended to the Remote Terminal Units (RTUs) and Intelligent Electronic Devices (IEDs). These malicious commands could disrupt power system operations and cause damage to power devices. In this paper, we propose an efficient signature scheme for ensuring the delivery of only legitimate commands over the communication network. We propose efficient signatures using hash that are sent by the control center along with commands for protecting command alteration over the network, which also provide sender authenticity by verifying the signatures at the RTU/IED. By using the proposed scheme, both random and malicious commands sent by an adversary are discarded. The security analysis shows that the scheme is secure against replay, repudiation, and impersonation attacks. The performance evaluation and prototype simulation show that our signature scheme using SHA1, SHA256, SHA384 is feasible, and can be used for communications from the control center to the field.**

***Index Terms*—Smart grid security, power system communications, malicious commands, short signature, security attacks.**

I. Introduction

THE Smart Grid (SG) is a modernized electricity delivery system, which relies heavily on two-way communications, information, and software to optimize control and to achieve advanced grid functionalities. The smart grid relies on the capability of transmitting accurate data and control commands in a timely and secure manner [1]. If an adversary gains unauthorized access to the grid control system, she can perform various types of security attacks. Some types of attacks, such as reply attacks [2], are difficult to detect. In this paper, we consider a traditional Supervisory Control and Data Acquisition (SCADA) control scenario. This model considers a central Supervisory Node (SN) (such as the Control Center (CC)) and a remote Control Node (CN), usually represented by Remote Terminal Units (RTUs) and Intelligent Electronic Devices (IEDs), which sense field quantities and actuate on the various power devices. The SN is connected to various CNs through different communication media and protocols. An adversary who has gained access to the network can send malicious commands or can modify transmitted commands and data that may result in disruption to the physical power system.

There are various communication technologies and protocols used in power system SCADA. An important standard for transmission of command messages is IEC 60870-5-101, which defines power system monitoring, control and associated communications for tele-control, tele-protection, and associated telecommunications. This standard is compatible with IEC 60870 part 5-1 to 5-5 and uses a standard asynchronous serial tele-control channel interface between the Data Terminal Equipment (DTE) and the Data Circuit-Terminating Equipment (DCE). IEC 60870 is based on Enhanced Performance Architecture (EPA) that considers only the physical, data-link, and application layers of the OSI model [3]. IEC 60870 was originally developed for serial communications, but with the subsequent release of the IEC 60870-5-104 standard in 2000, the same serial frames are allowed to be transmitted over TCP/IP [4]. This protocol specification is suitable for communications between different components within the substation network. At present, most of the power utilities use Distributed Network Protocol (DNP3) for real-time communications between the supervisory and control nodes. DNP3 is highly standardized, with relatively high compatibility and inter-operability between devices from different manufacturers. This protocol consists of three layers: data-link, application, and transport pseudo layers [5]. We use DNP3/C37.118 protocol packet format to send a control command and/or alert message.

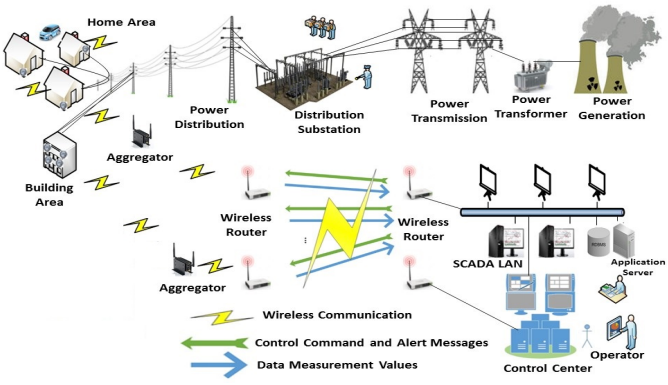


Fig. 1: A communication scenario between the supervisory and control nodes in the smart grid.

N. Saxena is with the Department of Computing and Informatics, Bournemouth University, UK.

S. Grijalva is with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, USA.

E-mail: nsaxena@ieee.org, sgrijalva@ece.gatech.edu

Manuscript received December XX, 20XX; revised XXX XX, 20XX.

1949-3053 (c) 2016 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.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSG.2017.2655014, IEEE Transactions on Smart Grid

A. Research Problem and Challenges

Figure 1 illustrates the system model considered in this paper including supervisory and control nodes. Here, a supervisory node can correspond to the control center or a microgrid central controller. The control nodes on the other hand are closer to the electrical devices and can correspond to a substation Remote Terminal Unit (RTU), an Intelligent Electronic Device (IED), the smart meter, etc. The operator at the control center is responsible for making decisions based on the operating conditions of the power system and sends control commands to different power devices at the substation in order to operate the power system as planned. An adversary in the smart grid can alter the power system dynamics by malicious yet valid commands over the wireless network. It is therefore required to analyze resiliency of the system to malicious command attacks. A promising direction is to secure the power network against malicious commands trying to change the state of the power system by actions, such as opening circuit breakers. The smart grid network must have a mechanism that ensures that the control node receives control commands and alert messages over the communication network that were actually sent by the legitimate supervisory node. Since the latency requirements for control commands are also crucial, the mechanism must be fast and efficient. In some cases a control arming signal is sent to the RTU as alert message to make sure that the command can get to the actual address and be executed. Upon confirmation by the operator, the command is actually executed by the RTU/IED. This command arming can be represented as an alarm by the control node. The scenario considered has following requirements and challenges:

- Since the supervisory node communicates frequently with the control nodes in order to exchange control commands and alert messages, the generated communication overhead must be as low as possible.
- The supervisory and control nodes must not be able to perform repudiation attacks while transmitting and receiving control commands and alert messages. Also, the supervisory node (malicious one, such as insider attacker) must not be able to perform impersonation and replay attacks.

In order to protect the smart grid communications between the supervisory and control nodes, a scheme must be designed that it carries out semantic analysis as follows:

- A module at the supervisory node generates a fresh command and sends it to the respective control node with fresh information.
- A module at the control node is activated immediately after receiving a command from the supervisory node, which could verify whether the received command is legitimate or malicious.

There are two different kinds of malicious command behaviors: (i) the command has different behavior than a normal operation, such as not a regular command, suspicious command for a critical power component, uncommon command for opening transmission lines, etc., and (ii) the command was sent by a malicious user or adversary. We capture first command behavior by using an Intrusion Detection System (IDS) with defined rules and later command behavior is verified by the proposed scheme.

Let consider a real-world smart grid scenario. During polling request, the supervisory node (control center) sends a command to one or more control nodes (RTUs) over the communication network, and asks to send their data. As a polling response, the control nodes send their measurement data to the supervisory node. During polling request, the RTU at a substation checks whether the command was sent by a legitimate control center or not. The RTU verifies the signature of the sender with message integrity. Normally, sending such commands by the control center is a part of its daily routine activities, which does not involve confidentiality of the information. By the time of polling response, the RTUs send periodic measurement data of different power components deployed at the substations to the control center. We must also protect this transmitted data from the adversary over an insecure network. In order to secure measurement data sent from the substation RTU to the control center, we propose an encryption over the transmitted data. However, in this paper, we only focus on accurate and correct command delivery, not the encryption of transmitted data. In a real world scenario, the utilities either protect their communication networks using Virtual Private Network (VPN) or simply do not involve any protection due to a large deployment cost. Even in the presence of a VPN network, the adversary can modify the measurement data values or the commands just before the starting points of the VPN at the substation. The traditional techniques of encryption (*DES*, *AES*) and signing (*RSA*, *DSA*, *ECDSA*) are not suitable for encrypting the transmitted data and generating the signatures for commands delivery in the smart grid network due to generating large overheads. Such a scenario involves frequent communications between the supervisory and control nodes, which requires an efficient and lightweight scheme.

B. Our Contribution

We propose an efficient short signature scheme for secure delivery of commands and alert messages from the supervisory node to the control node. Our scheme is based on the security of one-way function and subset resilience of hash functions. We propose to generate the signatures at the supervisory node by using a hash function, and each time, send a new signature along with the message to the control node. Upon receiving the message and the signature, the control node verifies the received signature by using a one-way function and checks the message integrity by computing the hash of the message and comparing it with the received hash. The following are our contributions in this paper:

- Our scheme ensures the correct and accurate delivery of control commands and alert messages from the supervisory node to the control node by using efficient one-time signatures that provide authenticity and integrity of the messages. One-time signatures are digital signatures

1949-3053 (c) 2016 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.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSG.2017.2655014, IEEE Transactions on Smart Grid

that can be used to sign a single message for each pair of verification and signing keys. Our scheme generates efficient (speedy) and light-weighted (low overhead) one-time signatures that provide strong unlinkability and does not require public key infrastructure. One-time signatures enable quick system recovery even if the system is compromised (worst case).
- As these commands and alert messages are not confidential, avoiding encryption improves the overall efficiency of the system, and also reduces the communication latency of the messages.
- The proposed signature scheme generates a lower overhead, as only one hash and one concatenation operations are used along with $m-$hash operations to generate $m-$signatures. Signature verification depends upon the number of communication instances and the types of communication (unicast, multicast, and broadcast). The overall overhead is lower compared to the existing schemes, hence meeting the time requirements of the control application.

The proposed scheme can also be utilized in the transmission of messages from the control node to the supervisory node. Some of these messages could be a specific alert message, suspect activity notification, etc. The proposed scheme is specific to the wireless communication, as the secret key is computed at the sender and the receiver based on wireless transmission error. The proposed scheme supports required features and does not require any modifications to the current communication infrastructure in smart grids.

The rest of the paper is organized as follows. Section 2 presents related work in the context of signature schemes. Section 3 presents the adversary model. Section 4 describes our signature scheme. Section 5 presents security analysis. Section 6 presents performance evaluation and experimental analysis. Finally, Section 7 presents the conclusion of this work. Table I describes various abbreviations and symbols used in this paper.

TABLE I: Symbols And Abbreviations

Symbol	Description	Size (bits)
SN	Supervisory node	–
CN	Control node	–
id_{sn}	Identity of the supervisory node	128
id_{cn}	Identity of the control node	128
$H()$	Cryptographic hash function	–
$\mathcal{H}$	A family of hash functions	–
h	hash code from $H()$	160-384
$f()$	One-way function	–
pk	A set of public keys of the SN	160-384
pk_i	A public key	160-384
sk	A set of private keys of the SN	160-384
sk_i	A private key	160-384
SK	A shared secret key	128
T	Timestamp	64
s_i	Signature	160-384
msg_i	Message	Variable
$negl$	Negligible	–
$Prob/Pr$	Probability	–
$Sign()$	Signature generation function	–
KeyGen()	Key generation function	–
Ver()	Verification function	–
MTU	Maximum transmission unit	Variable
$\|\|$	Concatenation	–

II. Related Work

In this section, we present work related to the signature schemes. Several researchers have developed attacks scenarios and their detection mechanism for smart grid applications, such as IEC 60870-5-101 [3], Maynard *et al.* [4], and Temple *et al.* [6]. Lin *et al.* [7] performed a security analysis on malicious control commands detection. However, a solution that could prevent malicious commands to be accepted by the control node has not been proposed. A feasible solution is to impose signatures over commands whenever they are sent as a part of polling request or control actions.

In general, few lightweight signature schemes are available in literature. In this direction, Perrig [8] proposed a one-time signature scheme, called *BiBa*. The scheme provides short signature and fast verification, but the signature generation is time consuming. Thereafter, Reyzin *et al.* [9] proposed a scheme, named *Better than BiBa* or *HORS* that generates signatures efficiently. However, this method has a limitation that if an adversary is able to swap the order of the signatures in a set, an attack can be performed [10]. The Powerball scheme [11] proposed by Mitzenmacher generates small size signatures, however, the cost of online signing is large. Another scheme, named *TV-HORS* [12] was proposed by Wang *et al.*, which provides fast signing/verification and buffering-free data processing. However, the scheme uses a relatively large public key of size 8-10 KB. *HORS++* scheme [13] proposed by Pieprzyk *et al.* is a variant of *HORS* that also uses a large key size. Thereafter, a one-time signature scheme was proposed by Zaverucha *et al.* [14] that supports aggregation and batch verification. Other signature schemes have also been proposed for the smart grid network [15], [16], [17]. Li *et al.* [15] presented an approach to authenticate data aggregation in smart grid via deploying signature aggregation. Vaidya *et al.* [16] proposed elliptic curve cryptography-based digital signature scheme for remote access to home area networks. Fu *et al.* [17] proposed a multi-dimensional aggregation signature scheme for smart grid communications. However, none of these schemes fulfills our communication and system requirements for secure command delivery. Yavuz *et al.* [18] presented a real-time broadcast authentication scheme based on *RSA* and *condensed-RSA*. However, the scheme generates a large storage and communication overheads.

Our work represents a departure from the existing works in the following way:

- Our scheme uses a set of hash functions, instead of a single hash function to generate signatures, and also maintains strong message integrity. A one-way function used in our scheme is announced later at the time of revealing the signatures. This provides stronger security and leaves less time to the adversary to make a forgery attempts on signatures.
- Our signature scheme utilizes the advantages of methods *BiBa* [8], *Better than Biba* [9] and the scheme in [12], and

1949-3053 (c) 2016 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.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSG.2017.2655014, IEEE Transactions on Smart Grid

addresses their disadvantages. *BiBa* has a large signing time, and *Better than Biba* does not have randomness to the input of hash function. The scheme in [12] has a quite large public key size. Our scheme has faster signing time by using only one hash operation. The scheme also provides randomness to hash, and uses a fixed size (160-384 bits) public and private keys by computing hash over keys to generate one-time signatures.

- In a real scenario, a command can be sent using radio and/or carrier communications. Our scheme uses DNP3/C37.118 format for commands delivery in the simulated experiment.

III. Adversary Model

We assume that an adversary $\mathcal{A}$ tries to perform *impersonation* attacks on behalf of the supervisory node in order to send malicious commands to the control nodes. $\mathcal{A}$ may also perform *replay* attacks by re-transmitting the commands sent during previous sessions. Furthermore, there can be an attempt of *insider* attack by the operator (as *repudiation attack*) at the supervisory node, where he/she sends a command to the control node and then later denies of sending the command. We also assume that $\mathcal{A}$ knows the possible selection of hash functions used in our scheme and tries to learn some information from the generated hash.

IV. Proposed Solution

In this section, we propose a signature scheme for the correct and accurate delivery of control commands and alert messages. Our scheme proposes lightweight signatures with message integrity for critical as well as non-critical commands and/or alert messages, such as sending an “open breaker" command for a generator. A communication scenario with our proposed scheme is illustrated in Figure 2, where we only focus on commands and/or alerts with digital signatures part in this paper. In our signature scheme, the supervisory node sends either a single or a set of generated public keys pk in cipher form to the respective control node depending upon the types of communication (unicast, multicast, and broadcast). These pk keys can also be sent to the control nodes in advance.

A. Efficient Signatures for Control Commands

We propose a short and efficient signature scheme to maintain authentication and integrity of commands and alert

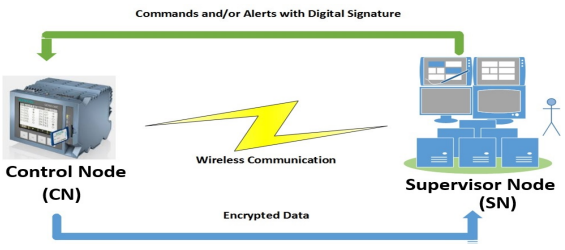


Fig. 2: Proposed scheme confirms the correct delivery of the commands over the network.

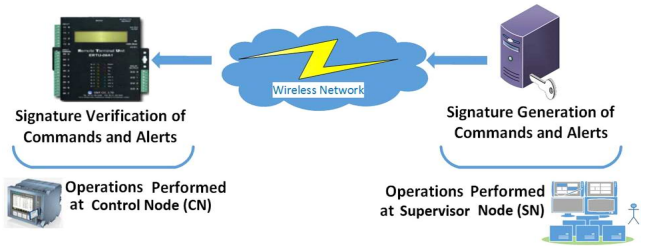


Fig. 3: Signature generation at the supervisory node and its verification at the control node.

messages, as illustrated in Figure 3. This scheme is suitable to be used by the supervisory node for unicast, multicast, and broadcast communications of commands and alert messages. A broadcast/multicast scenario is presented in Figure 4, in which the supervisory node (control center) sends a command to various control nodes (RTUs) simultaneously over the communication network. In order to secure measurement data sent from the substation RTU to the control center, we propose the encryption of transmitted data. In this paper, we only focus on accurate and correct command delivery, not the encryption of transmitted data. We assume that a secret key is shared between the control center and the substation RTU. For ensuring correct and accurate delivery of control commands from the control center to the substation RTU, our scheme delivers a one-time signature along with the command. The reason of choosing one-time signatures is quick system recovery. Our scheme does not provide encryption to the transmitted commands and alert messages due to the fact that the confidentiality of information is not our concern, rather the focus is to verify the supervisory node’s authenticity and the message integrity. Another advantage of disabling encryption is that it restricts the generated overhead to be remained low. Furthermore, received signature at the control node must be verified and matched with the supervisory node’s signature.

Our scheme is based on a set of cryptographic hash functions $H()$, such as *SHA1*, *SHA256*, and *SHA384* that maintains collision resistance. The SN can use any one of these hash functions (*SHA1*, *SHA256*, or *SHA384*) as $H()$, which generates a hash code h with specific length depending upon the hash function used (160, 256, or 384 bits, respectively). Various steps of the scheme are described in Algorithm 1. The method is based on [9] that splits hash code h into l

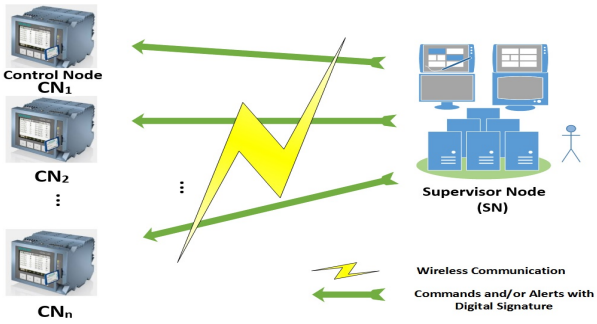


Fig. 4: A broadcast (or multicast) scenario between the supervisory node and all control nodes at the substation(s).

1949-3053 (c) 2016 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.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSG.2017.2655014, IEEE Transactions on Smart Grid

substrings of length $log_2\ n$ each, where n is the number of strings, each with length m. Each hash substring will generate a different binary code, interpretation of which will be a unique and random integer number. This integer value will decide the sequence of next signature value from the set and will provide randomness to the process of sending a command with one-time signature. Let us interpret each substring as an integer and form a subset N of size at most l. We assume a one-way function $f()$ that works on m-bit string. The integer value equivalent to the hash of a substring provides randomness to the order of public keys and acts as an index to match the $f(signature)$ with the correct public key. Note that in order to build a signature scheme that can sign l-bit messages, we need to choose n and l such that $\binom{n}{l} \geq 2^l$. The public key and signature sizes are linear in n and l, respectively. We assume a set {1, 2, ..., n} as N, and a hash function $H()$ that maps input message msg ($0 \leq msg \leq \binom{n}{l}$) to msg's l-element subset of N. Note that the scheme in [8] uses a bijective function in order to achieve subset of N. Our selection of using $H()$ over bijective function improves the efficiency and also makes it infeasible to find two messages msg_1 and msg_2 such that $H(msg_2) \subseteq H(msg_1)$.

Because utilizing shared symmetric key does not provide non-repudiation, we construct signatures based on asymmetric keys, in which each party has a private key and a public key. In our scheme, only the supervisory node has a set of private and public keys as the signatures can only be created by the supervisory node, and are only verified by the control node. Our scheme generates n random fixed-length private keys $sk = (H(sk_1), H(sk_2), ..., H(sk_n))$ from m-bit strings, and computes respective public keys as $pk = (pk_1, pk_2, ..., pk_n)$ at the supervisory node. Here, $pk_1 = f(H(sk_1)),\ ..., pk_n = f(H(sk_n))$. Furthermore, we interpret m-bit message msg_i as an integer value between 0 and 2^m-1. Consider $\{i_1, ..., i_l\}$ be the msg_i's l-element subset of N. These l-element subset can be used to generate signatures for unicast, multicast, and broadcast communications. In order to make available the public key pk_i to the control node, each public key pk_i can be sent (in cipher form) to a control node when the supervisory node sends the acknowledgment ack of a received frame/packet to the respective control node (which is not a part of this scheme). In fact, the supervisory node can provide a complete set pk to each control node in advance that can be used over a period of time for multicast and broadcast communications for non-critical commands with no restriction of a shared secret key expiration. However, the pk_i key is used only once per signature.

There is no need of distributing the key as the same key is generated at both ends independently. We consider that a same secret key is generated at the control node as SK_{cn_i} and at the supervisory node as SK_{sn_i}. The key expiration time depends upon the frequency of the wireless transmission error in the wireless communications. We consider the key expiration time as 5 minutes. For wired communications, we still believe that there is a secret key, depending upon wireless transmission error at both ends, which is used between the supervisory node and the control node. The purpose of this secret key is to encrypt the transmitted measurement values from the control node to the supervisory node using a dynamic secret generation scheme [19]. In case of unicast communication, the generated signatures can only be used within the expiration of secret key SK_{sn_i}. Once the key is expired, the supervisory node provides a new set of the pk_i keys, *i.e.*, pk to the respective control node. The supervisory node discloses private keys $\{H(sk_1), H(sk_2), ..., H(sk_l)\}$ as signatures $\{s_{i_1}, s_{i_2}, ..., s_{i_l}\}$ (based on h_j order of sequence), where i_j are integer values of h_j for $1 \leq j \leq l$. Multicast/broadcast communications are preferred over unicast communication for non-critical command/alert messages to various control nodes. It reduces the generated overhead per message per control node as well as the overall overhead of the system. On receiving the signature along with message msg_i, the control node generates h_j and interprets them as integer values $\{i_1, ..., i_l\}$ between 0 and 2^m-1 as message l-element subset of N. Each control node

Algorithm 1 *Proposed Signature Algorithm*

Key Generation at the supervisory node

Input: Parameters n: number of strings, m: length of each string, and l: number of substrings;
Generate n random m-bit strings $sk_1, sk_2, ..., sk_n$;
Generate hash of strings as $H(sk_1), H(sk_2), ..., H(sk_n)$;
Compute $pk_i = f(H(sk_i))$ for $1 \leq i \leq n$;
Output: $pk = (l, pk_1, pk_2, ..., pk_n)$ and $sk = (l, H(sk_1), H(sk_2), ..., H(sk_n))$;

Signature Generation at the supervisory node

Input: Interpret message msg_i as integer value, shared secret key SK_{sn_i}, and a set of private keys $sk = (sk_1, sk_2, ..., sk_n)$;
1. Let h = H($id_{sn}||msg_i||SK_{sn_i}$) for unicast communication used for critical commands, otherwise h = H($id_{sn}||msg_i$) for multicast and broadcast communications used for non-critical commands and alert messages;
2. Split h into l substrings $h_1, h_2, ..., h_l$ of length $log_2\ n$ bits each;
3. Interpret each h_j as an integer i_j for $1 \leq j \leq l$;
Output: A set of signatures = $(s_{i_1}, s_{i_2}, ..., s_{i_l})$, message msg_i, and timestamp value T_i, where $s_i = H(sk_i)$;

Signature Verification at the control node

Input: Message msg_i, a set of signatures = $(s_1, s_2, ..., s_l)$, received timestamp value T_i, and public key $pk = (pk_1, pk_2, ..., pk_n)$;
1. Check whether $T_{curr} \leq T_i + T_{th}$, where T_{curr} is current timestamp value and T_{th} is threshold timestamp value. If yes, proceed further otherwise, discard the command.
2. Compute h = Hash($id_{sn}||msg_i||SK_{cn_i}$) for unicast communication used for critical commands, otherwise compute h = H($id_{sn}||msg_i$) for multicast and broadcast communications used for non-critical commands and alert messages;
3. Split h into l substrings $h_1, h_2, ..., h_l$ of length $log_2\ n$ bits each;
4. Interpret each h_j as an integer i_j for $1 \leq j \leq l$;
Output: "Accept" if for each j, $1 \leq j \leq l$, $f(s_j) = pk_{i_j}$; "Reject" otherwise;

1949-3053 (c) 2016 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.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSG.2017.2655014, IEEE Transactions on Smart Grid

just follows simple commitment, computes $f(s_j)$, and verifies whether it matches with pk_{i_j}. In practice, instead of searching from i = 0 to l, the control node starts searching with i = l/2 and proceed in both directions. We perform binary search to find equivalent computed value $f(s_j)$ faster in a set of pk_{i_j}.

The msg_i in $id_sn||msg_i||SK_{sn_i}$ changes for each unique command targeted from supervisory node to control node. Also, the secret key SK_{sn_i} changes frequently in every few minutes. These both parameters change the value of $id_sn||msg_i||SK_{sn_i}$ within few minutes. The old values of SK_{sn} and SK_{cn} are discarded after their expiration times whereas the sk_i and pk_i are discarded after its one-time usage. There is no need of synchronizing the SK_{sn_i} and SK_{cn_i} list values as both keys are generated independently. Even, the occasional loss of packet will not affect our scheme as the scheme itself is based on the wireless transmission error.

B. Examples on Proposed Scheme

In this section, we provide two examples in order to illustrate our scheme. The first example considers a multicast communication scenario. The second example illustrates signature generation and verification.

Example 1: Consider a scenario in multicast communications, in which the supervisory node provides a set of four public keys $\{pk_1, pk_2, pk_3, pk_4\}$ to two control nodes, say CN_1 and CN_2. After signing the message, the supervisory node provides s_1 signature to the CN_1 and s_2 signature to the CN_2, and the respective control node verifies the signatures. The signatures s_1 and s_2 will never be re-used by the supervisory node to any control node, even though each control node has some unused signatures. Once the supervisory node utilizes all the generated signatures, a new set of public keys will be provided to each control node.

Example 2: Let us consider a simple example to understand the signature generation and verification. Assume that m = 160, l = 3 and n = 16 under multicast/broadcast communications. Initially, the supervisory node generates sk = ($H(sk_1)$,...,$H(sk_{16})$) and pk = ($pk_1, ..., pk_{16}$), and sends a set of public keys pk to the control node. Thereafter, the supervisory node computes $h = H(id_{sn}||msg)$, splits it into three substrings h_i ($1 \leq i \leq 3$) as h_1, h_2, h_3, and interprets each h_i as integer i_j ($1 \leq j \leq 3$). Here, hash function provides a security level as ($l \times log\ n$), *i.e.*, 12 bits. It produces three hash substrings, each with 4 bits in length. Consider i_1, i_2, and i_3 are 2, 9, and 15, respectively. Thus, the supervisory node reveals $H(sk_2)$, $H(sk_9)$, $H(sk_{15})$ as signatures s_1, s_2, s_3, and sends them to any three control nodes along with message msg, which is actually a command/alert message. On receiving the signature and the message, each control node directly computes $f(s_j)$ and verifies whether one of the public keys pk_{i_j} ($1 \leq j \leq 3$) in a set matches with $f(s_j)$. In a real scenario, in order to provide sufficient security level, we consider n as a large number where $log_2\ n$ is at least 160 bits. However, in case of unicast communication, on receiving a signature and the command message from the supervisory node, the respective control node computes h, generates substrings h_j, interpret each h_j as i_j, and then verifies whether $f(s_j) = pk_{i_j}$. For each subsequent signature within the expiry of secret key SK_{cn_i}, the control node can directly compute $f(s_j)$ and compare $f(s_j) = pk_{i_j}$, as the control node can use previously generated h_j and i_j because of having same $H(id_{sn}||msg_i||SK_{cn_i})$ for exactly same command. For unicast communication, the security level ($l \times log\ n$) of hash string should be at least 160 bits. One can consider l = 16, n = 2^{10} = 1024, or l = 20, n = 2^8 = 256 to maintain a standard security level of 160 bits hash code.

V. Security of the Proposed Scheme

This section presents the security analysis of the proposed scheme with its formal and provable security model.

A. Security Analysis

In this section, we present the security analysis of our short signature scheme. Given a message msg, the probability of finding a signature is equal to the probability of finding at least one two-way collision, *i.e.*, at least two m-bit strings, say sk_a and sk_b, have the same $f(H(sk_a)) \stackrel{?}{=} f(H(sk_b))$ when uniformly randomly chosen from n such strings. The security of the signature depends on the fact that the adversary has few forge signatures and therefore has a small probability to find a collision when uniformly randomly chosen t signatures from n such strings as $Prob = 1 - \prod_{i=1}^{t-1}(\frac{n-i}{n}) \approx 1 - e^{\frac{t(t+1)}{2n}}$ [8]. However, we increased the security by using a multi-round scheme, instead of two-way collisions similar to [8] (refer proof), where only the strings that have a k_1-way collision in the first round proceed in the next round. We prove the security strength of our scheme by the following properties:

Property 1: *Adversary $\mathcal{A}$ cannot perform replay and impersonation attacks over the communication network between the supervisory node and the control node.*

In our scheme, each command message is sent with a timestamp (or nonce) value from the supervisory node to the control node. This timestamp value is valid for a short period of time based on the propagation time between the supervisory node and the respective control node. Once the threshold time for accepting the message is over, the control node discards the message. If $\mathcal{A}$ sends a previously captured message to the control node, the control node discards the message due to invalid timestamp value. If $\mathcal{A}$ sends any old signature to the control node, the control node simply discards the message as the signature is invalid. Furthermore, if $\mathcal{A}$ acts as a supervisory node for the control node, the control node verifies the signature received with the message and finds it different than the earlier received as public key pk_i. This was also proved in [8].

Property 2: *Adversary $\mathcal{A}$ is not able to forge the strong one-way function used in our scheme.*

1949-3053 (c) 2016 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.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSG.2017.2655014, IEEE Transactions on Smart Grid

We can define a strong one-way function as follow:

Definition 1: A function $f : \{0,1\}^* \rightarrow \{0,1\}^*$ is said to be strongly one-way, if the following conditions hold:

- *Easy to compute:* There exists a polynomial-time algorithm *Algo* on input x, such that $f(x) = Algo(x)$.
- *Hard to invert:* For each probabilistic polynomial-time algorithm $Algo^{'}$ with a polynomial p and sufficiently large n: $\Pr(Algo^{'}(f(x)) \in f^{-1}f(x)) < 1/p(n)$.

We use secure hash function, such as *SHA1*, *SHA256* or *SHA384* as a strong one-way function [20]. It is easy to compute its output value (hash), but is not possible (impractically difficult) to reverse the operation. These functions are secure till date against different attacks, including collision attacks. Therefore, our selection of *SHA1*, *SHA256* or *SHA384* as strong one-way function is completely safe.

Property 3: *The subset of hash functions used in the proposed scheme is a subset-resilience.*

Let $\mathcal{H} = \{H_{i,n,l}\}$ be a family of functions, where $H_{i,n,l}$ maps an input of arbitrary length to a subset of size at most l of the set {0, 1, ..., n-1}. Note that for each n and l, H contains a number of functions, which are indexed by i. Moreover, assume that there is a polynomial-time algorithm that given i, 1^n, 1^l and *msg*, computes $H_{i,n,l}(msg)$ [9].

Definition 2: $\mathcal{H}$ is said to be r-subset-resilient, if for every probabilistic polynomial-time adversary $\mathcal{A}$,

$$\Pr_{i}[(msg_1, msg_2, ..., msg_{r+1}) \leftarrow \mathcal{A}(i, 1^n, 1^l)$$
$$s.\ t.: H_{i,n,l}(msg_{r+1}) \subseteq \cup_{j=1}^{r} H_{i,n,l}(msg_j)] < negl(n,l).$$

For r = 1, the definition can be realized using only collision-resilient hash families. Namely, if $\mathcal{H}$ is a collision-resilient hash function family, and S is a subset-selection algorithm, then $H(S)_{H \in \mathcal{H}}$ is a 1-subset-resilient family. For $r > 1$, the definition of using only common complexity-theoretic assumptions is an open problem [21]. However, it is reasonable to assume that selecting subset via cryptographic hash functions, such as *SHA1*, *SHA256*, and *SHA384* satisfies the definition for small values of r.

Property 4: *Adversary $\mathcal{A}$ cannot create or forge the signature of the supervisory node in our scheme.*

The security proof for our scheme follows directly from subset-resilience of hash functions, and the one-way property of function $f()$. The generated signatures in our scheme are unforgeable against r-time chosen plaintext attack as we consider hash functions, such as *SHA1*, *SHA256*, and *SHA384*, of r-subset-resilient and a strong one-way function $f()$, which is also one of these hash functions. We use *SHA1*, *SHA256*, and *SHA384* hash functions with 160, 256, and 384 bits of hash code, respectively, which are considered a secure level as per the current security standard. However, companies, such as Microsoft and Google, have decided to discontinue *SHA1* after 2016 [22]. Security systems will therefore be migrating *SHA1* to other forms of security, accordingly. A combination of l and n, such as l = 16, n = 2^{10} = 1024, or l = 40, n = 2^4 = 16 can be used depending upon the applications and requirements. If $\mathcal{A}$ tries to forge a signature depending upon whether $\mathcal{A}$ uses a new element of N or re-uses the previous revealed signatures, an attempt to do will be failed. The reason is that the supervisory node does not re-use previously revealed signatures. Also, $\mathcal{A}$ cannot choose the same subset as the supervisory node selects because the supervisory node modifies msg_i using the current secret key before generating a hash code that $\mathcal{A}$ does not know. Therefore, $\mathcal{A}$ cannot forge the signatures of the supervisory node. Furthermore, in our scheme unicast communications are used for the delivery of critical commands. Since $\mathcal{A}$ does not know the secret key SK_{cn_i}, it cannot compute a correct h. For non-critical commands and alert messages, our scheme uses multicast and broadcast communications. Even if $\mathcal{A}$ attempts to replace the signature with a forge signature, the control node will not be able to match the signature value with the public key value (earlier received from the supervisory node).

B. Provable Security of the Scheme

This section discusses the provable security of the proposed scheme, particularly, the practice-oriented provable security. A system is practice-oriented provable secure if its security requirements of the concrete objects, such as hash functions and block ciphers with fix key sizes, and assumptions about the hardness of certain computations are stated. We assume that the adversary has access to the system and has enough computational resources for efficient computations. Traditionally, the provable security is asymptotic that classifies the hardness of computational problems using polynomial-time reducibility. It also emphasizes the use of the Random Oracle (RO) model that are assessed with regards to their usefulness, such as hash, encryption, and decryption random oracles. Basically, a random oracle is a theoretical black box that responds to each unique query with a random response chosen uniformly from its output domain. When a query is repeated, a random oracle responds in the same way each time a query is submitted. Almost all modern security proofs take the reduction approach and rely on the hardness of some mathematical problem in order to prove their security.

1) Formal Security Model with Game-based Approach: We present a formal security model that consists of two elements:

1) Let us consider that an arbitrary, probabilistic, and polynomial-time adversary can interact with legitimate users of the smart grid system. The adversary can retrieve any message over an insecure network and also an output of the massage of its choice from a hypothetical probabilistic algorithm called a challenger.
2) In order to compromise the system and to forge a message, an adversary must be able to generate the correct signature of the actual sender. In fact, the adversary must generate or capture the private key of the sender and create a forged signature.

Let us follow a game-based approach to evaluate our scheme. In our security model, the adversary interacts with a hypothetical probabilistic challenger algorithm. This chal-

1949-3053 (c) 2016 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.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSG.2017.2655014, IEEE Transactions on Smart Grid

lenger responds to all the queries made by the adversary. The game terminates when an adversary announces its decision. The adversary wins the game if it can correctly guess the security parameters and break the system. A system is proved secure if the probability that an arbitrary adversary breaks the system is small.

The challenger generates an asymmetric key pair (sk_i, pk_i) of the appropriate security level. Now, the adversary's algorithm is executed, which takes a public key (pk_i) and a security parameter (n) as inputs. The adversary asks the challenger to generate a signature (s_i) for a message (msg_i) of its choice. The challenger does the same using a signature generation algorithm and the private key (sk_i) of the actual sender (control center). The adversary is deemed to have broken the system if the signature verification algorithm concludes that s_i, generated by $\mathcal{A}$, is a valid signature for the message msg_i.

Brute force attack analysis is one of the best ways of analyzing the security of the system. This security parameter typically dictates the appropriate length of the public and private keys in order to keep the system secure in today's world. In the brute force attack analysis, an adversary evaluates all the possible combinations of a key if it knows the length of the key. The adversary comes to a "true" conclusion if one of the key's combinations was used by the sender for signing a message. However, if the length of the key is large enough, it becomes infeasible and impractical for $\mathcal{A}$ to compute all the possible combinations within the key validity time.

2) Digital Signature Scheme: A digital signature is a source authentication segment appended at the end of a message. The sender generates a digital signature using a signing algorithm that takes a message to be signed and the user's private key as inputs. The recipient performs the signature verification using a verification algorithm, the message, the signature, and the sender's public key as input. Note that it is infeasible to produce a signature for a message without knowledge of the private key of the sender.

Definition 3: (Digital signature scheme). For a message space $\mathcal{M}$, a digital signature scheme Σ is a triple of algorithms $\Sigma = (\textit{KeyGen}, Sign, \textit{Ver})$.

KeyGen$() \rightarrow (sk, pk)$: A probabilistic key generation algorithm that takes no input, and generates a set of signing (private) and public keys.
$Sign(sk_i, msg_i) \rightarrow s_i$: A probabilistic signing algorithm that takes as input a signing key sk_i and a message $msg_i \in \mathcal{M}$, and outputs a signature s_i.
Ver$(pk_i, msg_i, s_i) \rightarrow \{0, 1\}$: A deterministic verification algorithm that takes as input a public key pk_i, a message msg_i, and a signature s_i, and outputs either 0 (invalid) or 1 (valid).

Definition 4: (Correctness). A digital signature scheme is correct if for all $msg_i \in \mathcal{M}$, all $(sk_i, pk_i) \leftarrow$ *KeyGen()*, and all $s_i \leftarrow Sign(sk_i, msg_i)$, the *Ver*$(pk_i, msg_i, s_i) = 1$ is a true statement.

An adversary may attempt several possible security breach goals: (i) *Key recovery:* tries to compute sk_i given the public key pk_i, and acts as a signer, (ii) *Universal forgery:* computes a valid signature s_i for a retrieved message msg_i, and (iii) *Existential forgery:* computes a valid signature s_i for a message msg_i of its choice. We can give the adversary different powers. The adversary: (i) *Key-only:* receives public key pk_i of the sender, (ii) *Known-message:* retrieves a list of the message-signature pairs from a pre-selected list of the messages, and (iii) *Adaptive chosen-message:* may adaptively obtain signatures for the messages of its choices. The best security notion is Existential Unforgeability under Adaptive Chosen Message Attack (EU-CMA). We want to verify $Succ_{\Sigma}^{EU-CMA}(\mathcal{A}) = Pr(Exp_{\Sigma}^{EU-CMA}(\mathcal{A}) = 1)$.

In order to forge a signature for a message msg_i, the adversary submits a query to a challenger, which is a random oracle. In our scheme, $\mathcal{H}$ is a hash family having different hash functions $H()$ that a challenger implements. The adversary can obtain the hash values of the messages from the challenger.

Challenger$_{\mathcal{H}}^{RO}(\mathcal{A})$
$H() : \{0,1\}^* \rightarrow \mathbb{Z}_n^*$
Initialization: Hash-list $\leftarrow \phi$
Query: If $(h_i, msg_i) \in$ Hash-list for a msg_i then return msg_i else $msg_i \leftarrow \mathbb{Z}_n^*$
Add (h_i, msg_i) to the Hash-list and return msg_i.

3) Game Approach: Let us consider a game-based scenario in order to prove the security of the proposed scheme.

Game 0. This is the original EU-CMA game for the proposed signature (PRO-SIG) scheme. Hence, $Succ_{PRO-SIG}^{EU-CMA}(\mathcal{A}) = Pr(s_i)$.
Game 1. The adversary makes the first attempt to guess a random private key sk_i. If the adversary can correctly guess or retrieve the actual private key of the sender, it can break the system by creating the correct but fake signature of the sender (control center). However, since sk_i was chosen at random from $\mathbb{Z}_n^*$ (generated by a secure pseudorandom generator PRG), the distributions are identical, so $Pr(sk_2) = Pr(sk_1)$. Therefore, $\mathcal{A}$ has no advantage in guessing the private key correctly. Also, the private key is never sent over the network, hence, $\mathcal{A}$ cannot retrieve the key from the network.
Game 2. In this game, an adversary tries to guess (h_i, msg_i) in the Hash-list. In this second query attempt, the adversary tries to correctly guess a message msg_i and asks the challenger to provide signature $s_i = H(sk_i)$ for the chosen message msg_i. If $\mathcal{A}$ correctly guess the private key sk_i and/or the message msg_i, the game 0 and (game 1 and/or game 2) will be equivalent. Assuming the adversary makes $query_r$ queries for generating random private keys sk_i and $query_h$ queries for the message integrity using $H()$. Therefore, $Pr(correct\_guess) = 1/(query_r + query_h + 1)$. Note $query_h$ will depend upon all the hash functions used in the scheme ($query_{h_{SHA1}}$ + $query_{h_{SHA256}}$ + $query_{h_{SHA384}}$).

Generally, the security of the cryptographic hash functions can be proved by three different properties: pre-image resistance, second pre-image resistance, and collision resistance.
Pre-image Resistance (PR): Given a hash h_i, finding any message msg_i such that $h_i = H(msg_i)$ is hard. Hash

1949-3053 (c) 2016 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.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSG.2017.2655014, IEEE Transactions on Smart Grid

functions lacking this property are vulnerable to pre-image attacks.

Second Pre-image Resistance (SPR): Given an input msg_1, finding another message msg_2 $(msg_2 \neq msg_1)$ is hard such that $H(msg_1) = H(msg_2)$. This is also known as weak collision resistance. One-way hash functions lacking this property are vulnerable to second pre-image attacks.

Collision Resistance (CR): It is hard to find two different messages msg_1 and msg_2 such that $H(msg_1) = H(msg_2)$. Such a scenario is known as hash collision. This property is also known as strong collision resistance. It requires a hash value to be at least twice as required for pre-image resistance, otherwise collisions may be found by a birthday attack.

We have used strong one-way hash functions, such as *SHA1*, *SHA256*, and *SHA384*. Therefore, our scheme is secure against these attack.

$$Pr[(msg_1, msg_2) \leftarrow \mathcal{A}(\Sigma, H) : msg_1 \neq msg_2 \wedge H(msg_1) = H(msg_2)] \leq \epsilon(n).$$

There is no polynomial-time (efficient) algorithm that breaks the pre-image of the hash. The bits of the message are nicely mixed in a hash function to produce a hash code. Various modular additions, bitwise operations (xor, rotations), and compression functions are used in iterative mode for ensuring high complexity and pseudo-randomness of the hash output. In this way, the security is proved by these operational functions.

Game 3. The adversary may attempt to analyze the m-bit private key sk_i and the generated hash codes by the hash functions $H()$. In the proposed scheme, the length of the sk_i key is 2048 bits, which generates 2^{2048} different combinations of a key. Therefore, it is impossible for an adversary to generate or guess the exact private key sk_i. Hence, $\mathcal{A}$ has no advantage over any other user in the system that could correctly guess the private key.

$$Adv_{sk_i}^{BruteForce}(\mathcal{A}) = Pr[b = b^{'}] - 1/2.$$

$\mathcal{A}$ has to make at least 2^{1024} attempts to correctly guess the private key using birthday paradox. Similarly, in order to retrieve a correct hash code, $\mathcal{A}$ needs to make at least 2^{80}, 2^{128}, and 2^{192} attempts, respectively, for *SHA1*, *SHA256*, and *SHA384*. $\mathcal{A}$ will only be successful if it can retrieve the correct private key from the signature.

Game 4. We will now prove our main theorem which states that the proposed scheme is secure as long as the used functions (families) provide standard security properties. These properties are fulfilled by secure cryptographic hash functions. We provide a standard security notion for the signature scheme that considers EU-CMA attack.

Definition 5: A signature scheme is called existentially unforgeable under a q-adaptive chosen message attack, if an adversary $\mathcal{A}$ makes at most q-queries but has only negligible success probability of winning the game.

We denote our signature scheme with security parameter n as $Sig(1^n)$, key generation as *KeyGen*(1^n), signature generation as $Sign(sk_i, .)$, and signature verification as *Ver*(pk_i, msg_1, s_i), where s_i is a generated signature. Let $\{(msg_i, s_i)\}_1^q$ be the query-answer pairs of $Sign(sk_i, .)$. The standard security notion for our signature scheme under EU-CMA can be represented using the following experiment:

$$\textbf{Experiment } Exp_{PRO-SIG}^{\textit{EU-CMA}}(\mathcal{A})$$
$$Setup : (sk_i, pk_i) \leftarrow \textit{KeyGen}(1^n),$$
$$Execution : (msg_1, s_i) \leftarrow \mathcal{A}^{Sign(sk_i,.)}(pk_i).$$

Return 1, iff *Ver*(pk_i, msg_1, s_i) = 1 and $msg_1 \notin \{msg_i\}_1^q$. else return 0.

For a success probability of $\mathcal{A}$ in this experiment, we define: $Succ_{PRO-SIG}^{EU-CMA}(\mathcal{A}) = Pr[Exp_{PRO-SIG}^{EU-CMA}(\mathcal{A}) = 1]$.

However, in our scheme, $\mathcal{A}$ can neither generate the same signatures nor forge the public keys of the supervisory node. The reason is because $\mathcal{A}$ cannot generate a shared secret key that is used to transmit the public keys (in cipher form) from the supervisory node to the control node.

We prove security of the scheme when $H()$ is modeled as a random oracle. Let t be an upper bound on the number of instances of the scheme overall. Thus, we have a set $\{h_i, msg_i\}_{i=1}^t$, where $msg_i \neq msg_j$ for $i \neq j$. For the exact statement in the proof we use the notion of insecurity functions. An insecurity function $InSec_{PRO-SIG}^{EU-CMA}(s, t, q)$ describes the maximum success probability of an adversary against the system of primitive s, running in time $\leq t$, and making no more than q queries to a signing oracle.

$$InSec_{PRO-SIG}^{EU-CMA}(s, t, q) \stackrel{def}{=} \max_{\mathcal{A}}\{Succ_{PRO-SIG}^{EU-CMA}(\mathcal{A})\} = negl(n).$$

Theorem. Let $n, v, msg \in \mathbb{N}$, $\mathcal{F} : \{f : \{0,1\}^n \rightarrow \{0,1\}^n\}$ a Second Pre-image Resistant (SPR) and Undetectable (UD) one-way function family that maintains One-Wayness (OW). Then, $InSec_{PRO-SIG}^{EU-CMA}(1^n, m), t, 1)$, the insecurity of PRO-SIG against EU-CMA attack is bounded by

$$InSec_{PRO-SIG}^{EU-CMA}(1^n, m), t, 1) \leq InSec_{PRO-SIG}^{UD}(\mathcal{F}, t') + m.\max_{\mathcal{A}}\{InSec_{PRO-SIG}^{OW}(\mathcal{F}, t''), InSec_{PRO-SIG}^{SPR}(\mathcal{F}, t'')\}$$

where $t' = t + 3m$ and $t'' = t + 3m + l$ [23].

Proof: Let us use contradiction. We assume that there exists an adversary $\mathcal{A}$ that can produce existential forgeries for PRO-SIG running an adaptive chosen message attack in time $\leq t$ and with success probability $Succ_{PRO-SIG}^{EU-CMA}(\mathcal{A})$ greater than the claimed bound $InSec_{PRO-SIG}^{EU-CMA}(1^n, m), t, 1)$. A random oracle first runs the *KeyGen*() key generation to obtain a key pair (sk_i, pk_i). In our scheme, the control center sends a set of encrypted public keys pk to the substation using a secret key SK_{sn_i}. Therefore, $\mathcal{A}$ does not have the access to a public key, say $pk_i^{'}$. $\mathcal{A}$ must know the one-way function $f()$ in order to generate a public key of the control center from the retrieved signature, say, $s_i^{'}$. Even if $\mathcal{A}$ does this, the generated $pk_i^{'}$ must be the same as the one used from a set of the public keys based on h_j. This provides no advantage to $\mathcal{A}$, as first $\mathcal{A}$ has to correctly guess l and then choose the correct $pk_i^{'}$ used by the control center. This proves that the scheme is secure against a random guessing.

1949-3053 (c) 2016 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.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSG.2017.2655014, IEEE Transactions on Smart Grid

Now, let assume that $\mathcal{A}$ can retrieve a public key $pk_i^{'}$ of the control center and a signature $s_i^{'}$. An adversary correctly guesses l, runs $\mathcal{A}^{Sign(sk_i,.)}(pk_i^{'})$ and submits a query to a random oracle for signing a message $msg_{i}^{'}$. The oracle runs $Sign(sk_i^{'}, msg_i^{'})$ and generates a signature s_i for $\mathcal{A}$, where $sk_i^{'}$ is one of the randomly generated strings and $s_i^{'}$ is a signature (hash computed), *i.e.*, $s_i^{'} = H(sk_i^{'})$. Now, $\mathcal{A}$ has $msg_i^{'}$, $s_i^{'}$, and $pk_i^{'}$, and its goal is to retrieve the private key $sk_i^{'}$. The existence of a function that combines these properties is equivalent to the existence of a one-way function. Note that the second pre-image resistant functions exist if a one-way function exists [24]. We know the same for the undetectable functions, *i.e.*, the pseudorandom generators [25]. Using the assumptions about the one-wayness and the second pre-image resistance of $\mathcal{F}$, we can bound the success probability of $\mathcal{A}$, *i.e.*, $Succ_{RO}^{P}(\mathcal{A})$, when accessing m-signatures using a random oracle:

$$m.\max_{\mathcal{A}}\{InSec_{PRO-SIG}^{OW}(\mathcal{F}, t''), InSec_{PRO-SIG}^{SPR}(\mathcal{F}, t'')\}$$

where $t'' = t + 3m + l$ is an upper bound for $\mathcal{A}$'s runtime accessing all three algorithms. The public key generation maintains one-wayness by using a one-way function $f()$ whereas the signature generation requires hashing of the private key that maintains the pre-image resistance. We use $f()$ as a cryptographic hash function, which is also pre-image resistant.

Next, we bound the difference between the success probability $Succ_{RO}^{P}(\mathcal{A})$ using a random oracle and its success probability $Succ_{EXP}^{P}(\mathcal{A})$ in the original experiment (*EXP*). If the first probability is greater than the latter, we already have a contradiction. We assume that $\mathcal{A}$ asks for the signature on one message $msg_i^{'}$ and observes the behavior over $\mathcal{M}$ distribution. We now consider an oracle that uses the possibly different behavior of $\mathcal{A}$ when given different distributed inputs to distinguish between the Public Key Distribution (*PKD*) and Message Distribution (*MD*). We compute the distinguishing advantage

$$Adv_{PKD,MD}(\mathcal{A}) = Succ_{EXP}^{P}(\mathcal{A}) - Succ_{RO}^{P}(\mathcal{A}).$$

We only need to consider the case when $Succ_{EXP}^{P}(\mathcal{A}) \geq Succ_{RO}^{P}(\mathcal{A})$. In order words,

$$Succ_{EXP}^{P}(\mathcal{A}) = Adv_{PKD,MD}(\mathcal{A}) + Succ_{RO}^{P}(\mathcal{A}).$$

But, $\mathcal{A}$ has negligible advantage, as a pseudorandom generator is used for the key distribution. This implies that:

$$Adv_{PKD,MD}(\mathcal{A}) \leq InSec_{PRO-SIG}^{UD}(\mathcal{F}, t').$$

where $t' = t + 3m$. Now, putting all together,

$$Succ_{EXP}^{P}(\mathcal{A}) \leq InSec_{PRO-SIG}^{UD}(\mathcal{F}, t') + m.\max_{\mathcal{A}}\{InSec_{PRO-SIG}^{OW}(\mathcal{F}, t''), InSec_{PRO-SIG}^{SPR}(\mathcal{F}, t'')\}$$

where $t' = t + 3m$ and $t'' = t + 3m + l$. This is a contradiction. This proves that there is no such adversary $\mathcal{A}$ that can produce existential forgeries in time $\leq t$ with success probability $Succ_{PRO-SIG}^{EU-CMA}(\mathcal{A})$ greater than $InSec_{PRO-SIG}^{EU-CMA}(1^n, m), t, 1)$.

VI. Performance Evaluation and Experimental Analysis

This section presents performance evaluation of our scheme. We also discuss an experimental analysis of our simulation.

A. Performance Evaluation

Our signature scheme requires only two evaluations of the hash function: one for signature generation and other for signature verification. In addition, l evaluations of one-way function $f()$ and hash function $H()$ for signature verification are also required. Note that l varies in different scenarios of unicast, multicast, and broadcast communications. In our scheme, we consider a set of hash algorithms, such as {*SHA1*, *SHA256*, *SHA384*} for selecting a hash function $H()$ as well as a one-way function $f()$. One of the advantages of using $H()$ instead of bijective function is that it removes the restriction of a fixed length message (m-bit in length), as $H()$ can process the input of variable length. Our scheme allows the mapping of elements in subset N starting from one element and up to at most l elements. Hence, the supervisory node can utilize efficient mapping based on the types of communication (unicast, multicast, and broadcast) as well as the number of communicated control nodes. Furthermore, our scheme generates signatures of size $l \times m^*$, signature cost as O(1), key generation cost as O(n), public keys of size $n \times m^*$ and verification cost (l+1) as O(l), where m^* size is 160-384 bits.

B. Experimental Analysis

In this work, we develop an experimental setup with a co-simulator. The co-simulator uses JDK1.7 with Gridsim, PowerWorld, MATLAB, and Java Agent Development Framework (JADE) to implement communication and power system scenarios between the control and the supervisory nodes. We consider a 24 substations system case with 42 buses, 62 lines, 7 generators, 27 loads, 6 transformers, and 9 shunts. Table II describes the selected ranges of the communication parameters for our simulation, where C37.118 protocol supports data packets with message passing. We simulate the system by varying the baud rate from 100-9600 bits/sec, propagation delay ranges from 10-100 ms, Maximum Transmission Unit (MTU) ranges from 32-1024 bytes (with transport segment 1-249 bytes), and packet size varies from 50-1500 bytes.

The command types and its actions include changing (open/close) the status of various power system components, such as line status (LineStatus), generator status (GenStatus), load status (LoadStatus), transformer status (modeled as LineStatus), and shunt status (SSStatus). Figure 5 depicts

TABLE II: Parameters for Simulation Setup

Parameters	Range Value	Unit
Baud rate	100-9600	bits/sec
Propagation delay	10-100	ms
MTU	32-1024	bytes
Packet size	50-1500	bytes

1949-3053 (c) 2016 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.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSG.2017.2655014, IEEE Transactions on Smart Grid

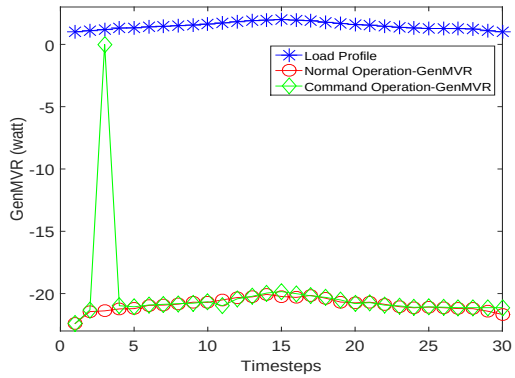
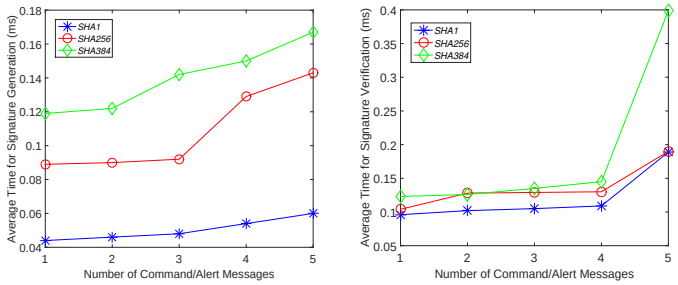


Fig. 5: A load profile for 30 timesteps as one timestep per minute, and pass “Open Gen" command at timestep = 3.



(a) Signature generation. (b) Signature verification.

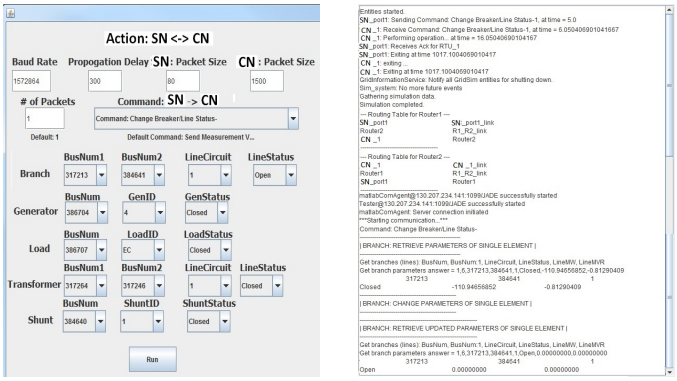
Fig. 6: Average time for signature generation and verification using *SHA1*, *SHA256*, and *SHA384* in a scenario with baud rate = 100 bits/sec, propagation delay = 100 ms, MTU = 237 bytes, and packet size = 500.

a simple scenario of load profile for 30 timesteps along with the GenMVR when normal and “Open Gen" command operations are performed. Figure 6 shows the average times for the signature generation and verification by our scheme considering three hash functions, *i.e.*, *SHA1*, *SHA256*, and *SHA384*. In the multicast and broadcast communications, the supervisory node announces a hash function name (one-way function by which a set of public keys is generated) at the time of revealing its signatures. We consider a scenario with the baud rate = 100 bits/sec, propagation delay = 100 ms, MTU = 237 bytes, and packet size = 500 bytes. In this experiment, we find that the signature generation and verification times for 1-5 data packets are (0.044-0.060, 0.096-0.189), (0.089-0.143, 0.104-0.190), and (0.119-0.167, 0.123-0.399) ms, respectively, for *SHA1*, *SHA256*, and *SHA384*. *SHA1* takes lesser times

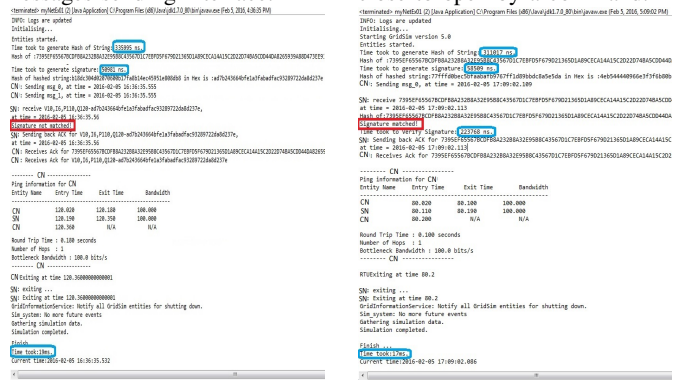
TABLE III: Comparison of Signature Schemes

Scheme	Signature size (bits)	Verification cost	Private key cost	Signing cost	Public key size (bits)
BiBa [8]	ml	$2l$+1	n	$2n$	mn
Powerball [11]	ml	$2l$+1	$2n$	$2n$	mn
HORS [9]	ml	l+1	n	1	mn
Unicast Scheme$^{\#}$	m^*	1	1	1	m^*
Multicast Scheme$^{\#}$	m^*l	l+1	l	1	m^*l
Broadcast Scheme$^{\#}$	m^*l	l+1	n	1	m^*n

m^* is at least 160-384 bits, $Scheme^{\#}$ is our scheme.



(a) A GUI of command and alert messages sending interface. (b) A line status changes from close to open by a command.

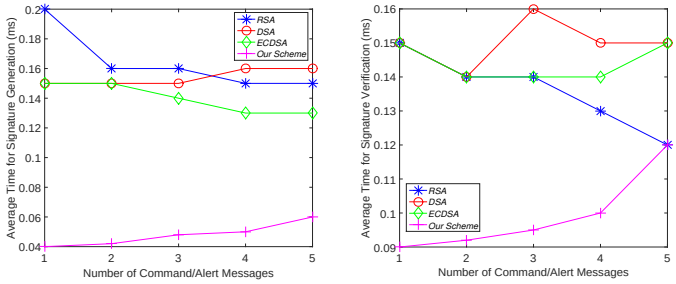


(c) A malicious signature generation and its detection. (d) A legitimate signature generation and verification.

Fig. 7: A prototype of our scheme: transmission of a malicious and legitimate commands from the control center to the RTUs with signature generation and verification.

for the signature generation and verification as compared to *SHA256* and *SHA384*. However, *SHA384* provides better hash code security. The DNP3 protocol standard has a default control duration time of 250 ms, and the channel latency could be lowered to 15 ms with 9600 baud and 32 bytes of data [5]. In these scenarios, our scheme works very well and provides an efficient and secure service to deliver control commands and alert messages.

We also compare our signature scheme with the existing schemes. As shown in Table III, our scheme is efficient than



(a) Signature generation. (b) Signature verification.

Fig. 8: Average time for signature generation and verification by *RSA*, *DSA*, *ECDSA*, and our scheme using *SHA1*.

1949-3053 (c) 2016 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.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSG.2017.2655014, IEEE Transactions on Smart Grid

the other schemes, and is suitable for the command and alert messages to be sent from the supervisory node to the control node over the network. Figure 7(a) shows a working prototype of our scheme by which transmission of malicious as well as legitimate commands are allowed from the control center to the RTUs (such as change a line status in Figure 7(b)). The scheme can detect and verify whether the sent command was malicious (Figure 7(c)) or legitimate (Figure 7(d)).

Figure 8(a) and Figure 8(b) show the average times for the signature generation and verification, respectively, by the traditional signature schemes, such as *RSA*, *DSA* and *ECDSA*, and our scheme using *SHA1* hash function. These figures clearly depict that the proposed scheme is efficient and have lower signature generation and verification times in comparison to the traditional signature schemes. Therefore, the proposed scheme is practical, efficient, and reasonably effective in order to deploy over the real smart grid network.

VII. Conclusions

This paper has described an efficient signature scheme for secure communications between the supervisory node and the control node in power system control and smart grid applications. The supervisory node sends an acknowledgement of the received frame/packet as well as encrypted public keys (or public keys in a secure channel) to the control node. Our efficient signature scheme provides non-repudiation and message integrity protection during unicast, multicast, and broadcast communications of critical and non-critical commands and/or alert messages. The security analysis of our scheme ensures that our scheme defeats repudiation, replay, and impersonation attacks over the network. Furthermore, performance evaluation and experimental analysis of our scheme show that the process of signature generation and verification is lightweight with lower overhead and faster execution. We use all three functions *SHA1*, *SHA256*, and *SHA384* in our scheme to provide randomness of selecting hash function, which also improves the overall security level of our scheme.

References

[1] Introduction to NISTIR 7628 Guidelines for Smart Grid Cyber Security, The Smart Grid Interoperability Panel Cyber Security Working Group, 2012. [Online]. http://www.nist.gov/smartgrid/upload/nistir-7628_total.pdf

[2] 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.

[3] IEC 60870-5-101, IPCOMM, [Online]. http://www.ipcomm.de/protocol/IEC101/en/sheet.html

[4] P. Maynard, K. McLaughlin, and B. Haberler, “Towards understanding man-in-the-middle attacks on IEC 60870-5-104 SCADA networks," in *Proc. International Symposium for ICS & SCADA Cyber Security Research*, 2014, pp. 30-42.

[5] IEEE Standard for Electric Power Systems Communications, IEEE Std 1815-2012, Distributed Network Protocol, 2012. [Online]. https://standards.ieee.org/findstds/standard/1815-2012.html.

[6] W. G. Temple, B. Chen, N. O. Tippenhauer, “Delay makes a difference: smart grid resilience under remote meter disconnect attack," *IEEE International Conference on Smart Grid Communications (SmartGridComm)*, 2013, pp. 462-467.

[7] H. Lin, A. Slagell, Z. Kalbarczyk, P. W. Sauer, R. K. Iyer, “Semantic security analysis of SCADA networks to detect malicious control commands in power grids," in *Proc. ACM workshop on Smart energy grid security*, 2013, pp. 29-34.

[8] Adrian Perrig, “The BiBa one-time signature and broadcast authentication protocol," in *Proc. ACM CCS*, 2001, pp. 28-37.

[9] L. Reyzin and N. Reyzin, “Better than BiBa: short one-time signatures with fast signing and verifying," in *Proc. Information Security and Privacy Conference*, LNCS 2384, Jun. 2002, pp. 144-153.

[10] Y. Park and Y. Cho, “Efficient one-time signature schemes for stream authentication," *Journal of Information Science and Engineering*, vol. 22, pp. 611-624, 2006.

[11] M. Mitzenmacher and A. Perrig, “Bounds and improvements for BiBa signature schemes," No. TR-02-02, CS Group, Harvard University, 2002, pp. 1-15.

[12] Q. Wang, H. Khurana, Y. Huang, and K. Nahrstedt, “Time valid one-time signature for time-critical multicast data authentication," in *Proc. IEEE INFOCOM*, Apr. 2009, pp. 1233-1241.

[13] J. Pieprzyk, H. Wang, and C. Xing, “Multiple-time signature schemes against adaptive chosen message attacks," in *Proc. Selected Areas in Cryptography*, LNCS 3000, 2004, pp. 88-100.

[14] G. M. Zaverucha and D. R. Stinson, “Short one-time signatures," *Cryptology ePrint Archive*, Report 2010/446, 2010. [Online] http://eprint.iacr.org/

[15] D. Li, Z. Aung, J. R. Williams, and A. Sanchez, “Efficient authentication scheme for data aggregation in smart grid with fault tolerance and fault diagnosis," in *Proc. IEEE PES Innovative Smart Grid Technologies (ISGT)*, Washington, USA, 2012, pp. 1-8.

[16] B. Vaidya, D. Makrakis, and H. Mouftah, “Secure communication mechanism for ubiquitous Smart grid infrastructure," *The Journal of Supercomputing*, vol. 64, no. 2 , pp. 435-455, 2013.

[17] S. Fu, J. Ma, H. Li, and Q. Jiang, “A robust and privacy-preserving aggregation scheme for secure smart grid communications in digital communities," *Security Comm. Networks*, 2015, doi: 10.1002/sec.1188.

[18] A. A. Yavuz, “An efficient real-time broadcast authentication scheme for command and control messages," *IEEE Trans. on Information Forensics and Security*, vol. 9, no. 10, pp. 1733-1742, 2014.

[19] T. Liu, Y. Liu, Y. Mao, Y. Sun, X. Guan, W. Gong, and S. Xiao, “A dynamic secret-based encryption scheme for smart grid wireless communication," *IEEE Trans. on Smart Grid*, vol. 5, no. 3, pp. 1175-1182, May 2014.

[20] J. L. Carter and M. N. Wegman, “Universal classes of hash functions," *J. of Computer and System Sciences*, vol. 18, pp. 396-407, 1979.

[21] R. Canetti, R. Rivest, M. Sudan, L. Trevisan, S. Vadhan, and H. Wee, “Amplifying collision resistance: a complexity-theoretic treatment," in *Proc. Advances in Cryptology*, 2007, pp. 264-283.

[22] Security Advisory 2880823: Recommendation to discontinue use of SHA-1, 2012. [Online]. http://blogs.technet.com/b/srd/archive/2013/11/12/security-advisory-2880823-recommendation-to-discontinue-use-of-sha-1.aspx.

[23] A. Hulsing, “W-OTS+âĂŞ shorter signatures for hash-based signature schemes," in *Proc. Progress in Cryptology âĂŞ AFRICACRYPT*, LNCS vol. 7918, 2013, pp. 173-188.

[24] J. Rompel, “One-way functions are necessary and sufficient for secure signatures," in *Proc. ACM Symposium on Theory of Computing*, 1990, pp. 387-394.

[25] J. Hastad, R. Impagliazzo, L. A. Levin, M. Luby, “A pseudorandom generator from any one-way function." *SIAM Journal of Computing*, vol. 28, pp. 1364-1396, 1999.

1949-3053 (c) 2016 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.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSG.2017.2655014, IEEE Transactions on Smart Grid

Neetesh Saxena [S'10-M'14] is a Lecturer in Cyber Security with the Department of Computing and Informatics at Bournemouth University, UK. Before joining BU, he was a Post-Doctoral Researcher in the School of Electrical and Computer Engineering at the Georgia Institute of Technology, USA. Prior to this, he was with the Department of Computer Science, The State University of New York (SUNY) Korea, South Korea as a Post-Doctoral Researcher and a Visiting Scholar at the Department of Computer Science, Stony Brook University, USA. He earned his PhD in Computer Science and Engineering from Indian Institute of Technology, Indore, India. In 2013-14, I was a Visiting Research Student and a DAAD Scholar at Bonn-Aachen International Center for Information Technology (B-IT), Rheinische-Friedrich-Wilhelms University, Bonn, Germany. He was also a TCS Research Scholar during Jan. 2012 - Apr. 2014. His current research interests include cyber security, cyber-physical system security in the smart grid and vehicle-to-grid, security and privacy in the cellular networks, securing end-to-end systems, and secure mobile applications. He is a member of ACM.

Santiago Grijalva [M'02-SM'07] is the Georgia Power Distinguished Professor of Electrical and Computer Engineering and Director of the Advanced Computational Electricity Systems (ACES) Laboratory at The Georgia Institute of Technology. His research interest is on decentralized power system control, power system informatics and economics, and future sustainable energy systems. He has been principal investigators for various research projects under Department of Energy, ARPA-E, EPRI, PSERC, NSF and other industry and Government sponsors. From 2002 to 2009 he was with PowerWorld Corporation as a software architect and consultant. From 2013 to 2014 he was on assignment to the National Renewable Energy Laboratory (NREL) as founding Director of the Power System Engineering Center (PSEC). Dr. Grijalva's graduate degrees in Electrical and Computer Engineering, M.Sc. (99), Ph.D. (02) are from the University of Illinois at Urbana-Champaign.

1949-3053 (c) 2016 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.