Dynamic Secrets and Secret Keys Based Scheme for Securing Last Mile Smart Grid Wireless Communication

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

***Abstract*—An integrated and optimized smart grid cannot be achieved without a secure communication network. Due to the large-scale nature of the power system, the variety of technologies used, and limitations of communication bandwidth, supervisory applications for smart grid still use weak security in many deployments. Adversaries can potentially modify measurement values or inject bad commands over the network. In this paper, we propose a novel scheme based on dynamic secrets and encryption with secret keys. The scheme generates a series of dynamic secrets over the communication network, which are used to generate secret keys for data encryption. The generation of dynamic secret is frequent and no adversary can compromise the network for a longer period, even if he/she knows a secret key. The scheme is secure against eavesdropping, malicious communication injection, man-in-the-middle attack, replay attack, impersonation attack, and chosen-plaintext attack. The security analysis and performance evaluation show that our scheme is feasible to be used in the communication between supervisory and control nodes of various smart grid applications.**

***Index Terms*—Smart grid security, dynamic secret, secret key, secure wireless communication, measurement data.**

I. Introduction

THE Smart Grid (SG) is a promising platform that will bring utility electricity delivery systems into the 21st century by using two-way digital communications technologies and computer-based distributed control and automation [1], [2]. Today's electric power grid is being extended with several advanced power technology features, such as sensing, dynamic power flow analysis, real-time situation control analysis, etc. A communication network in the smart grid can be realized using various wired and wireless technologies overlaid on the power system. Several short-range wireless technologies, such as Zigbee, Bluetooth, and Wi-Fi, as well as wide-range wireless technologies, such as WiMAX, UMTS, and LTE are used in the smart grid [3], [4]. The security of the smart grid network also affects the availability, reliability, and efficiency of the power system. The integration of the communication network with the power system though causes important security challenges and exposes vulnerabilities, including the potential for various types of attacks, such as Man-in-the-Middle (MITM) attacks, replay attacks, and impersonation attacks [5]. Several researchers have demonstrated numerous mechanisms for attacks, which can take place in a smart grid setting [6]. It is therefore necessary to implement a cyber security element on top of the communication layer, which itself is conceptually implemented at the top of power system layer.

When the communication network is not secure, it is always possible for an adversary to perform security attacks. Consider the case where adversary $\mathcal{A}$ identifies vulnerabilities with the intention to compromise power system data and to disrupt power devices and apparatus. In case of secure communication network, adversary $\mathcal{A}$ still tries to capture the underlying system secrets, such as shared keys in symmetric communication or private keys used in asymmetric communications.

In this paper, we consider generic smart grid applications that have a Supervisory Node (*SN*) and a Control Node (*CN*), as shown in Figure 1. Here, a supervisory node can correspond to the control center, the aggregator, or a microgrid controller. The control node is closer to the electrical devices and can correspond to a substation Remote Terminal Unit (RTU), an Intelligent Electronic Device (IED), the electric vehicle, the smart meter etc. One example is the communication between the control center and the substation RTU. During polling request, the control center (*SN*) asks all the RTUs (*CN*) to send their periodic data and as a polling response, all the RTUs transmit their data to the control center over the network. The communications between the supervisory node and control nodes at the substations may use protocols, such as Distributed Network Protocol (DNP3), defined in IEEE Std. 1815-2010 [7]. The DNP3 protocol has several known security issues, such as absence of authentication, message integrity, and confidentiality of the message information. In 2012, a new version of the DNP3 protocol, named DNP3 secure authentication version 5, was announced [8]. This version of DNP3 provides methods to remotely change user update keys using either symmetric or asymmetric cryptography. However, this version defeats only spoofing, modification, and replay attacks over the network, and does not provide confidentiality to the message. This version is not backward compatible, which results in significant protocol replacement effort and cost. An alternative is to develop a cyber-security layer that can handle security issues over the communication network.

N. Saxena and S. Grijalva are with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, USA.
E-mail: mr.neetesh.saxena@ieee.org, sgrijalva@ece.gatech.edu

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

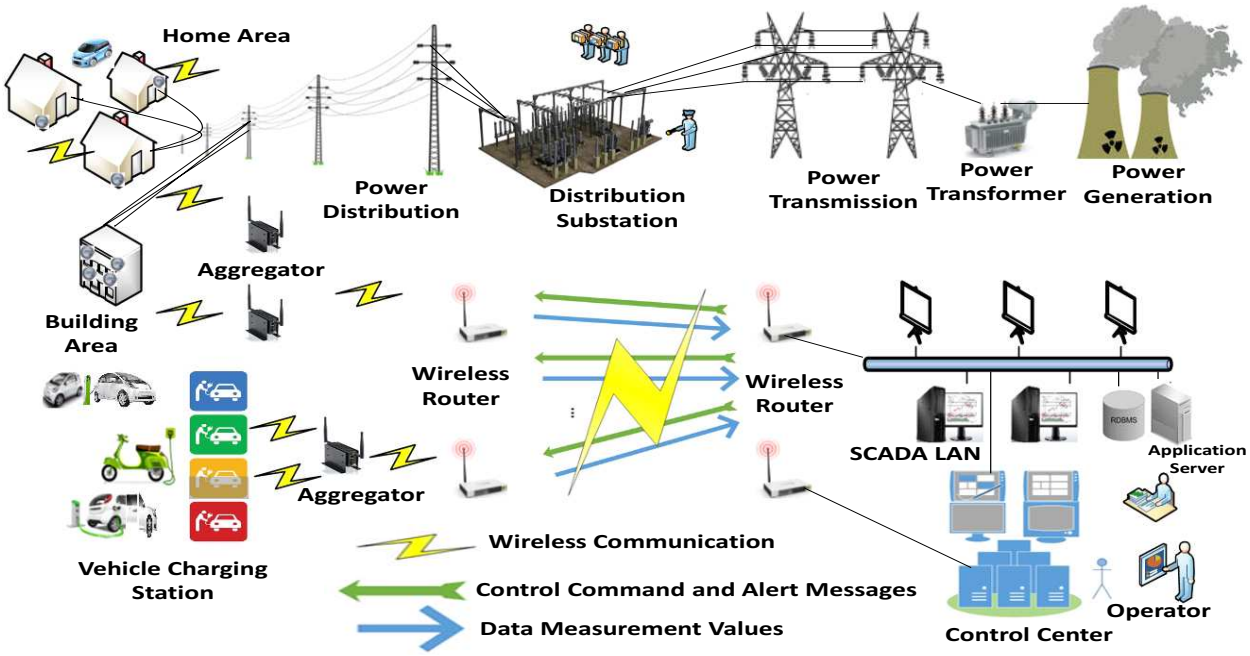


Fig. 1: A communication scenario between the supervisory node and the control node at a substation.

The DNP3 protocol is based on a simplified model, named Enhance Performance Architecture (EPA) that consists of three layers: data-link layer, application layer, and transport pseudo layer. The transport pseudo layer partitions large messages into smaller messages that the data-link layer is capable of handling. Application, transport, and data-link layers in the transmitter add information to the message for enabling the same layers in the receiver to correctly process the messages [9]. If the message is very large, fragmentation of the message is performed over the application layer. Then, segmentation of each fragment is done by the transport pseudo layer, and thereafter segments fit into data-link layer frames. The DNP3 data-link layer frames are transmitted across the IP network using Transmission Control Protocol (TCP) or User Datagram Protocol (UDP) packets. In addition, the control node can also send unsolicited responses to the supervisory node. These are messages sent from a control node without a specific response to the supervisory node, when something of significance occurs. However, to make this possible, the supervisory node should be able to accept such messages from any control node at any time.

A. Research Problem and Challenges

Various wireless technologies are applied to meet specific requirements of the smart grid network. However, wireless communications lead to many potential vulnerabilities that provide a room for the adversary to steal the secrets (keys and private information). Also, the user is often not aware of the fact whether the secret is stolen or not.

During data acquisition, the supervisory node sends a poll request to each control node, and each control node dumps its data in a series of the DNP3 packets to the supervisory node. Adversary $\mathcal{A}$, present between the supervisory node and the control node, may compromise the transmitted information of the packets. $\mathcal{A}$ can also alter the measurement values in the transmitted packets [10]. As a result, the power system may become insecure. Therefore, the smart grid network must have a security scheme to protect these information over the communication network. The scenario considered has the following challenges:

1) Since each control node frequently communicates with the supervisory node in order to transmit measurement values, the scheme must be very efficient and should generate communication overhead as low as possible.
2) The storage and computation capabilities are limited at the control node. Hence, any over burden of computation decreases the efficiency of the system. The proposed solution must hence be lightweight in terms of lower storage and computational requirements.
3) As various control nodes and the supervisory node are deployed at different geographical locations, the system recovery from a critical security issue (over a wide communication network) may result in a significant delay. Therefore, a solution that is capable of self-managing is needed that can quickly recover the system from insecure state, and can be implemented independently from third-party involvement.
4) The transmission of information over the communication network between the control node and the supervisory node experiences many security attacks, such as reply attacks, MITM attacks, and impersonation attacks. Such attacks must be defeated by a security solution.

TABLE I: Symbols And Abbreviations

Symbol	Description	Size (bits)
SN	Supervisory node	–
CN	Control node	–
$H()$	Hash function	–
PT	Plaintext	Variable
CT	Ciphertext	Variable
id_{cn}/id_{sn}	Identity of the entity	128
DS_{cn}/DS_{sn}	Dynamic secret	160-384
SK_{cn}/SK_{sn}	Shared key between SN-CN	160-384
T	Timestamp	64
h	Hash value	160-384

B. Our Contribution

We propose a novel dynamic secret scheme for secure communications between the supervisory node and the control node implementing stop-and-wait protocol. Our solution is based on wireless transmission errors and randomness, as well as the security of hash functions. In our proposed scheme, a shared secret key is automatically generated at the supervisory node and the control node by a dynamic secret, and is used to encrypt and decrypt the data. This allows the following:

1) Transmission of encrypted measurement values, such as line active and reactive power, transformer current magnitude, and voltage magnitude and phase with message integrity from the control node to the supervisory node over the network.
2) Generation of a series of dynamic secrets and then obtains the secret keys using a lightweight technique based on hash functions and XOR operations. This overall scheme generates lower computation overhead as well as manageable communication overhead depending upon the quantity of measurement values to be delivered in each packet.
3) Frequent generation of dynamic secrets and shared secret keys in the system that always self-manage and recover the system, even if $\mathcal{A}$ retrieves a dynamic secret and/or secret key.

The rest of the paper is organized as follows. Section 2 presents related work in the context of dynamic secrets, hash functions, and encryption schemes. Section 3 presents the adversary model. Section 4 proposes and presents our scheme that generates dynamic secrets and secret keys, and performs encryption over measurement data. Section 5 and Section 6 present security analysis, and performance evaluation and prototype experimental analysis, respectively. Finally, Section 7 concludes the work. Table I describes abbreviations and symbols used in this paper along with their descriptions and sizes.

II. Related Work

This section presents related work on the smart grid wireless communication security schemes. Schemes [11] and [12] are based on physical layer security, while schemes [13], [14], [15], [16] are based on data-link layer security. Capar *et al.* [11] implemented elliptic curve Diffie-Hellman secret sharing technique using a fast frequency hopping method, whereas Wong *et al.* [12] proposed a secret sharing scheme using relaiability-based hybrid automatic-repeat-request over the fast fading channel. Sheng *et al.* [13] and Xiao *et al.* [14] proposed a method for securing communications in a wireless network by utilizing inherent randomness of propagation errors to generate a secret key. Sun *et al.* [15] used the idea in [13] of generating dynamic secrets at the data link layer, however, its traditional message encryption with DES and AES generates a very large overhead. Furthermore, Liu *et al.* [16] proposed a Dynamic Secret-based Encryption scheme (*DSE*) for the smart grid communication. This scheme generates retransmission sequence to obtain Dynamic Secret (*DS*), and then computes Dynamic Encryption Key (*DEK*). However, encryption function used in the scheme seems insecure, as it performs an XOR between the plaintext and the secret key, and does not provide randomness to the input. Same plaintext will produce the same ciphertext, which may reveal the secret key.

Many symmetric key algorithms, such as Triple-DES and AES have been proposed to be used in the smart grid network [17]. Furthermore, many researchers have used asymmetric key algorithms based on specific needs. Some algorithms based on latest cryptographic techniques, such as homomorphic encryption for secure aggregation [18] and mutli-party computation for different applications [19], have also been proposed for the smart grid network, which allow performing specific but limited operations over the encrypted data. However, these algorithms are not efficient in practice due to high speed and lightweight requirements. Moreover, symmetric key algorithms are faster than asymmetric key algorithms, but they have a short lifespan. Our system requirements do not fit well with using the traditional symmetric and/or asymmetric key algorithms. We tackle this problem by designing a new secure lightweight encryption algorithm.

Our work is different from the existing works and provides the following:

(a) Dynamic secret scheme implemented at the data-link layer as well as at the transport pseudo-layer instead of just working at the data-link layer [13], [15], [16] or the physical layer [11], [12] only. This enables the extension of security features at the transport pseudo-layer, as the smart grid network supports IP-based communications.
(b) Use of a set of hash functions instead of a single hash function to generate dynamic secrets, and to maintain data integrity.
(c) Use DNP3/C37.118 format for data delivery, which are wrapped inside the TCP/UDP packets. The beauty of the DNP3 packet is that it can support sending a set of data blocks in a single packet. When $\mathcal{A}$ performs attacks over the network, such as MITM attacks, and tries to make an attempt of extracting data, most likely our scheme will generate a new dynamic secret and secret key soon that will quickly recover the system.
(d) Propose an encryption function, which is based on XOR operations, randomness of input, and freshness of the

secret key. This encryption function is secure against Chosen-Plaintext Attack (CPA) and is different than the function used in [16], which is based on a single XOR operation between the plaintext and the secret key. The encryption function in [16] is not secure as repetition of same plaintext to be encrypted under the same key may reveal plaintext to $\mathcal{A}$.

III. Adversary Model

It is hard to perform perfect eavesdropping without losing information over the wireless network, even for a short period of time [13]. We assume that $\mathcal{A}$ tries to perform *eavesdropping* over the wireless network. The devices and users usually are not aware of a secret key being stolen, until $\mathcal{A}$ performs an active attack. We further assume that $\mathcal{A}$ can steal dynamic secret and decrypts the ciphertext, and makes an attempt to *inject malicious communications*. We also assume that $\mathcal{A}$ knows the possible selection of hash functions used in our scheme and tries to learn information from the generated hash. As the communications between the supervisory node and the control node take place in the plaintext form, $\mathcal{A}$ can easily *capture relevant information* over the network. Moreover, $\mathcal{A}$ can also attempt an *man-in-the-middle* attack by establishing two active connections, one between the supervisory node and $\mathcal{A}$, and other between $\mathcal{A}$ and the control node, and tries to modify or extract message information. $\mathcal{A}$ can also tries to perform *impersonation* attacks on behalf of the supervisory or control node.

IV. Proposed Scheme

In this section, we describe a novel scheme to mitigate communication vulnerabilities between the supervisory node and the control node. Our scheme presents a lightweight dynamic secret-based encryption scheme. Our goal is to generate a shared Dynamic Secret (*DS*) through a publicly known function and to make sure that $\mathcal{A}$ does not know the *DS*. In the wireless communication, it is almost impossible to eavesdrop link layer communication for a long period without errors [13]. The shared secret key is frequently and dynamically computed by the supervisory node as well as the control node. These shared secret keys are used to encrypt measurement values as well as unsolicited responses. A communication scenario with our proposed scheme is shown in Figure 2. We use hash functions, XOR, and other low computation based operations in our scheme in order to keep overheads as low as possible.

A. Dynamic Secrets and Encryption of Measurement Values

We propose a lightweight scheme that keeps the system secrets secure and also self-manages the security of the system, even if a secret is leaked over the network. This is possible by observing the fact that losing information over the wireless network is generic due to wireless transmission errors. The scheme generates a set of hash values at different time intervals at the supervisory node as well as at the control node in

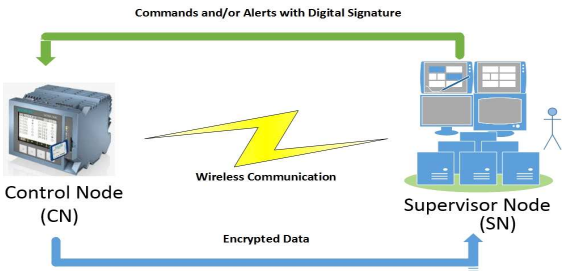


Fig. 2: A scenario of secure delivery of measurement values over the network.

order to maintain data integrity of measurement values. The supervisory node and the control node compute hash on the total number of successful delivery of (i) transmitted frames of all the packets (pkt) as well as (ii) each packet itself, and (iii) correctly received acknowledgments (ack). Note that long messages are usually segmented and de-segmented at the sender and the receiver ends, respectively. This is the reason that we also count successful delivery of each packet. However, adversary $\mathcal{A}$ cannot keep its eyes over the transmitted channel all the time. $\mathcal{A}$ cannot predict the actual value of successful transmitted pkt and ack due to frame/packet loss. If $\mathcal{A}$ misses even a single frame, it cannot retrieve the lost value. As a result, $\mathcal{A}$ will have a different computation value. Hence, $\mathcal{A}$ cannot generate the same dynamic secret.

Our scheme also provides an insight of a desirable computation model, in which $\mathcal{A}$ makes more efforts in order to retrieve the dynamic secret, not only for the current secret key but also for all the subsequent secret keys being generated in next few minutes. On the other hand, the supervisory node and the control node generate a dynamic secret using the shared information, hash values, and other operations.

1) Generation of Dynamic Secrets: In the presence of a potential adversary $\mathcal{A}$, it may happen that a single dynamic secret may not be sufficiently secured. However, with a series of new and frequently generated dynamic secrets, $\mathcal{A}$ will not be able to derive or capture all the secrets due to information loss. Nevertheless, the system may still be vulnerable, if the encryption function is not properly designed or selected from the existing secure ciphers.

Initially, the supervisory and the control nodes mutually agree on a predefined starting value str (a set of octets) as an initial secret generated by using Diffie-Hellman key exchange [22]. Note that Diffie-Hellman key exchange allows the generation of secret str at both ends without actually sent it over the network. The public parameters are sent over the network whereas the private parameters are kept secret. Thereafter, the control node sends a few random streams to the supervisory node for generating a dynamic secret at both ends. This makes even more difficult for $\mathcal{A}$ to retrieve or compute the correct secret. If $\mathcal{A}$ does not start from the beginning, it cannot recover the lost information, and therefore, will not be

able to generate the correct dynamic secret. The generation of dynamic secrets at the control node and the supervisory node are described in Algorithm 1 and Algorithm 2, respectively. Note that for each control node, the IP address of the supervisory node, *i.e.*, sn_{addr_i}, may be same with different port numbers and/or interfaces. We count the successful delivery of each frame of a packet and the completion of a packet delivery with message integrity as the total count computed by the supervisory node and the control node. We refer to the $count$ value as cn_count and sn_count at the control node and the supervisory node, respectively. Here, ϵ_1 and ϵ_2 are reasonable propagation times of a packet to be received by the supervisory node and the control node, respectively. The str is a pre-shared string shared between each control node and the supervisory node to initialize Algorithm 1 and Algorithm 2, and is different for each control node-supervisory node pair. Also, the adversary has no knowledge about the length of the random string and the actual length of the information to predict the counter value. A single bit changed the counter value will result in a significant change in the hash of the counter. In order to make the process more strong, we consider the length of counter at least 128 bits, which will provide 2^{128} possible combinations. The actual counter value starts with (str+counter) value. The adversary does not have the correct knowledge about the delivery failure of exact number of packets and frames over the wireless network. In addition, there is no need of synchronization, as both ends generate counter value based on successful delivery of the frames and packets and the acknowledgments received. If a control node is reset, the process starts again. Since str is generated using secure Diffie-Hellman key exchange, the adversary cannot retrieve the value of str in any case. Also, it is very hard (almost impossible) for an adversary to keep track of all the successful delivery of frames, packets, and acknowledgments due to wireless transmission errors. Hence, the adversary cannot derive the actual value of the counter, which is str + counter. In the worst scenario, if an adversary retrieves the encrypted strings, it cannot decrypt the messages, because it cannot derive dynamic secret and secret key considering the fact that the adversary cannot obtain correct and actual counter value.

Algorithm 1 *Count Generation at the Control Node*

count = 0;
for each sent packet pkt_i **do**
  create frames frm_i of the message msg_i;
  define cn_{addr_i} and sn_{addr_i} to each packet;
  compute a hash of its identity $H(id_{cn_i}||str)$ and generate a timestamp t_{cn_1};
  create $hash_i = H(H(id_{cn_i}||str)||msg_i||t_{cn_1})$ over the pkt_i payload and append it to the msg_i;
  send packet pkt_i to the supervisory node with msg_i as $H(id_{cn_i}||str)||\ msg_i||hash_i||t_{cn_1}$;
while ! end of sent packets per session **do** {
wait for acknowledgments ack_i of all frm_i as well as pkt_i;
  if all ack_i are received within $t_{cn_2} \leq t_{sn_2} + \epsilon_2$ **then**
    proceed;
  else
    set pkt_flag_j = 1 for all missing frm_j's ack_j;
  for each correct frame ack_i **do**
    set count++;
  for each correct packet ack_i **do**
    set count++;
    set pkt_flag_i = 0;
  for each unsuccessful frame of the packet pkt_j (no ack_j or negative ack_j) **do**
    if pkt_flag_j == 1 **then**
      re-transmit the missing frame frm_j to the supervisory node;
  }
set cn_count = count;

Algorithm 2 *Count Generation at the Supervisory Node*

count = 0;
for each received packet pkt_i **do**
  receive frames frm_i of message msg_i;
  verify sn_{addr_i} and sender cn_{addr_i} in each packet;
  if (timestamp $t_{sn_1} \leq t_{cn_1} + \epsilon_1$) && (current timestamp $t_{curr} \leq t_{sn_1}$) are true **then**
    proceed and store $H(id_{cn_i}||str)$;
  else
    terminates the connection;
  if all the frames are received within t_{sn_1} **then**
    proceed;
  else
    wait for a random time t_{rand};
  if updated timestamp $t_{curr} \leq t_{sn_1} + t_{rand}$ is true **then**
    proceed;
  else
    terminates the connection;
  if message integrity $hash_i \stackrel{?}{=} hash_i^{'}$ of the receive msg_i in packet pkt_i from the control node holds **then**
    proceed;
  else
    terminates the connection;
while ! end of received packets per session **do** {
send acknowledgments ack_i for all received frames frm_i as well as the pkt_i;
  for each correct frame frm_i **do**
    set count++;
  for each correct packet pkt_i **do**
    set count++;
    set pkt_flag_i = 0;
  for each unsuccessful frame of the packet pkt_j (no ack_j or negative ack_j) **do**
    set pkt_flag_j = 1;
    wait to receive missing frame frm_j from the control node;
  }
set sn_count = count;

2) Generation of Secret Keys: The processes of deriving secret keys from the generated dynamic secrets at the control node and the supervisory node are as follow:

Generation of Dynamic Secrets and Secret Keys at the CN: The control node generates a dynamic secret as:

$$DS_{cn_i} = H(cn\_count);$$

and the corresponding secret key as:

$$SK_{cn_i} = (DS_{cn_i} \oplus str);$$

Here, i represents the iteration number to generate the dynamic secret and the secret key. This secret key is used to encrypt measurement values in the packets in order to transmit them securely over the network from the control node to the supervisory node. However, it is important to verify whether the generated key is sufficiently eligible to generate secure symmetric cipher. For example, the generated SK_{cn_i} may be short in length because of many most significant bits as zeros. Therefore, this process of key verification is important and required. The secret keys for the subsequent sessions are computed as:

$$SK_{cn_{i+1}} = (DS_{cn_{i+1}} \oplus SK_{cn_i});$$

Generation of Dynamic Secrets and Secret Keys at the SN: In a similar way to secret keys generation at the control node, the supervisory node also generates the same set of secret keys. The supervisory node generates a dynamic secret as:

$$DS_{sn_i} = H(sn\_count);$$

Thereafter, the secret key is derived as:

$$SK_{sn_i} = (DS_{sn_i} \oplus str);$$

The subsequent secret keys are derived as:

$$SK_{sn_{i+1}} = (DS_{sn_{i+1}} \oplus SK_{sn_i});$$

We also design a secure cipher function for data encryption and decryption, which converts each plaintext PT_i (i = 1, 2, ..., d; d is the number of data values in each message of a packet) into ciphertext CT_i and vice versa, as follows:
At the control node:

$$CT_i = f_{cipher}(PT_i, counter_i, SK_{cn_i});$$

At the supervisory node:

$$PT_i = f_{cipher}(CT_i, counter_i, SK_{sn_i});$$

The definition of f_{cipher} for encrypting each plaintext PT_i of a message into ciphertext CT_i is as follows:

$$\begin{aligned} &f_{cipher}(PT_i,\ counter_i,\ SK)\{ \\ &CT_i = (PT_i + counter_i) \oplus SK; \\ &counter_i + +; \\ &return\ CT_i; \} \end{aligned}$$

Here, $counter_i$ is used to change the original plaintext before passing as input to the function f_{cipher}, and gets increment for each encryption of the plaintext message. The $counter_i$ works as an initialization vector for each message. The value of $counter_i$ is $H(lseqno \oplus lackno)$, where $lseqno$ and $lackno$ are the sequence number and the acknowledgement number of last successful delivered packet. Our scheme, based on XOR and $counter_i$, has computational advantages over other modes of operations. The f_{cipher}'s computations on different plaintext messages can be obtained in parallel, as the computation on each plaintext is independent of other plaintext message. Decryption is independent of the order of generating ciphertext, as each plaintext message is added with a different value of $counter_i$. Once the supervisory node and the control node have a shared dynamic secret ($DS_{sn_i} = DS_{cn_i}$), they also generate a shared secret key as SK_{sn_i} and SK_{cn_i}, respectively. Each control node sends $H(id_{cn_i}||str_i)$ to the supervisory node, and then the supervisory node stores it in the database, where id_{cn_i} is the identity of the control node and str_i is a pre-shared string. Thereafter, the control node encrypts message msg_i and sends $(id_{cn_i}||E_{SK}[msg_i]||hash_i||t_{cn_1})$ to the supervisory node, where t_{cn_1} is the timestamp and $hash_i$ = $H(id_{cn_i}||E_{SK}[msg_i]||t_{cn_1})$. On receiving the message, the supervisory node first computes $H(id_{cn_i}||str_i)$ and compares it with the stored values. If it matches with one of the values, the supervisory node stores id_{cn_i} in its database and removes the older stored value. The purpose of sending and computing $H(id_{cn_i}||str_i)$ at initial stage is that the SN could verify not only the sender's identity but also the shared str. Also, it is not a good idea of sending str as a plaintext over the network. In order to improve security, we used $H(id_{cn_i}||str_i)$ for initial communication. Once the SN confirms the identity of the CN, it will use id_{cn_i} for the subsequent communications within the session until the expiry time of str. This process reduces the overall computation overhead.

Figure 3 presents a communication scenario of transmitting measurement values from the control node to the supervisory node. First, analog data measurement values of different components of the power system are sensed through sensors and then digital values are stored in the memory of the control node. Furthermore, the control node encrypts these values using f_{cipher} with recent SK_{cn_i} key. Thereafter, the control node generates several packets based on the maximum size of payload and the total number of measurement values to be sent, and then sends these packets to the supervisory node over the communication network. On receiving such packets, the supervisory node verifies the integrity of the received message by computing $hash_i^{'}$ and comparing it with the received $hash_i$, *i.e.*, $hash_i \stackrel{?}{=} hash_i^{'}$. Then, the supervisory node extracts measurement values from the packets, and decrypts their original values using same function with SK_{sn_i} key.

Our scheme achieves the following properties:
Storage and computation capabilities: The proposed scheme is based on lightweight operations, such as XOR and addition. Therefore, the scheme generates low overheads and needs less storage.
Self management: The proposed scheme generates the dynamic secret and the secret key after each short period of time. In the

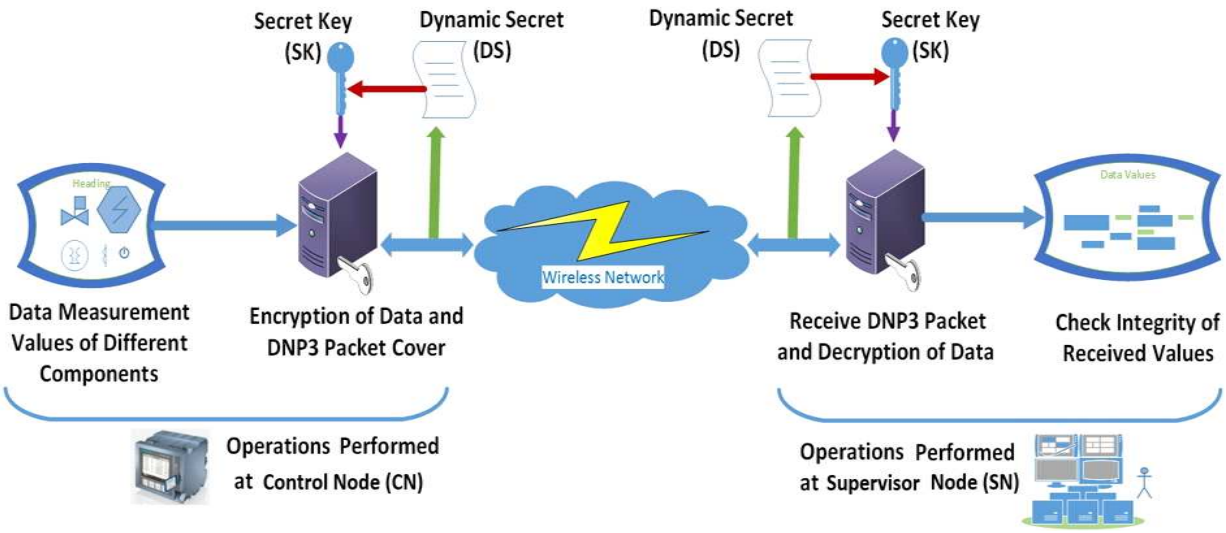


Fig. 3: Dynamic secret and secret key generation scheme.

worst case, even if the adversary compromises the secret key, the system will be shortly recovered by self by regenerating new dynamic secret and secret key.

Security: The proposed scheme is secure against eavesdropping (in worst case, it protects from a long time eavesdropping), man-in-the-middle attack, replay attack, and impersonation attack.

Efficiency: The generation time of the dynamic secret and the execution time of the proposed scheme will be very short, as the scheme uses lightweight operations to be performed, such as XOR, addition, and hash.

V. Security Analysis

In this section, we present the security analysis of our scheme considering the following properties:

Property 1: *Adversary $\mathcal{A}$ cannot eavesdrop the communication channel for a long time.*

Please note that there is no "secret exchange" during the protocol run. Each end computes its own secret key without sending it over the network. During the initial run of the protocol, the supervisory and control nodes mutually agree on a predefined starting value str (a set of octets), as an initial secret that is derived using Diffie-Hellman key exchange. The control node sends a random stream to the supervisory node for generating a dynamic secret at both ends. In the worst case scenario, $\mathcal{A}$ cannot eavesdrop the communication for a long time. $\mathcal{A}$ can generate neither dynamic secrets nor secret keys because of inevitable transmission errors in wireless network. Note that the control node and the supervisory node generate the same dynamic secret based on the str and the counter value on successful transmission of each frame and the packet, as well as each correct ack following stop-and-wait and DNP3/C37.118 protocols. The dynamic secret depends on the hash value of the counter not only the counter itself. A single bit changed the counter value will result in a significant change in the hash of the counter. In order to make the scheme stronger, we consider the length of the counter to be at least 128 bits, which will provide 2^{128} possible combinations. The actual counter value starts with (str+counter) value, where the str is a random value pre-shared between the SN and the CN, and is different for each such pair per session. In our scheme, the CN also sends a random stream to the SN, which will make it difficult for the adversary to correctly guess and count the value of the counter.

The proposed method does not rely on the adversary having poor equipment, but the presence of random wireless transmission errors. If an adversary tries to impersonate to an entity, it will not be successful, because the adversary cannot retrieve/derive the actual counter value. Hence, the adversary cannot derive the actual session secret key. Even if we consider that $\mathcal{A}$ generates the same dynamic secret, it cannot generate the shared secret key for the current session because the current secret key also depends upon the previous secret key (or str during initialization phase) and a new dynamic secret. Using previous and current dynamic secrets, $\mathcal{A}$ cannot generate the secret key. In the worst scenario, if we consider that $\mathcal{A}$ can obtain secret key and perform encryption/decryption, our scheme self recovers from eavesdropping. The reason is that in our scheme, dynamic secrets are generated frequently after sending few packets. Hence, once the new secret key comes into the effect, $\mathcal{A}$ can no longer eavesdrop the session.

Property 2: *Our scheme provides security against malicious communication injection. The scheme detects malicious injection performed by adversary $\mathcal{A}$.*

Our scheme provides encryption over the transmitted data sent from the control node to the supervisory node. Malicious communication injection is not possible, as $\mathcal{A}$ cannot correctly decrypt the data values from the packet. If $\mathcal{A}$ sends a malicious packet to the supervisory node, the supervisory node will

discard the packet, as it will not be able to correctly decrypt the packet. Therefore, our scheme is secure against malicious communication injection.

Property 3: *Adversary $\mathcal{A}$ cannot perform replay, MITM, and impersonation attacks over the communication network between the supervisory node and the control node.*

In our scheme, each packet is sent with a timestamp value from the control node to the supervisory node. This timestamp value is valid till a short period of time based on the propagation time between the respective control node and the supervisory node. Once threshold time of accepting the packet is over, the supervisory node discards the packet. If $\mathcal{A}$ sends a previously captured packet to the supervisory node, the supervisory node discards the packet as the timestamp is not valid. Since communication from the control node to the supervisory node is encrypted, $\mathcal{A}$ cannot successfully perform MITM attack, and an attempt of establishing connections between the control node and the supervisory node is discarded. In the case of compromised router in a wireless communication system, the adversary can drop, delay, or block the packet, but it cannot extract the information from the packets over the network. Our goal of preventing system against MITM is to restrict him from extracting or modifying the transmitted packets over the network. If $\mathcal{A}$ pretends as a control node and sends a message, the supervisory node tries to decrypt the message and makes an unsuccessful attempt. Also, hash of the message will be different, if $\mathcal{A}$ modifies the packet values.

Property 4: *Our encryption scheme is secure against Chosen-Plaintext Attack (CPA), and is a CPA Indistinguishable (CPA-IND) scheme.*

We analyze the security of our scheme against CPA attack in which adversary $\mathcal{A}$ selects various messages on which it wishes to obtain encrypted messages while communicating from the control node to the supervisory node. Our encryption scheme generates sufficient randomness to the input message based on $counter$ value, and then XORed with the secret key to finally generate the ciphertext. Note that we also monitor the generated secret key and the ciphertext, and ensure that the obtained ciphertext is significantly different from the original message (plaintext). Since dynamic secrets and secret keys change frequently, $\mathcal{A}$ cannot establish any relationship among received encrypted messages. Our scheme also maintains a security level of hash code to at least 160 bits and uses collision free hash functions, such as *SHA1*, *SHA256*, and *SHA384*.

Definition 1: A hash family $\mathcal{H}$ is (t, ϵ)-collision resistant, if no adversary $\mathcal{A}$ at t-time has an advantage at least in breaking the collision resistance of $\mathcal{H}$.

Let $\mathcal{H}$ = H be a hash family of functions $H : \{0,1\}^* \rightarrow \{0,1\}^n$. We say that algorithm $Algo$ has advantage ϵ in breaking the collision-resistance of $\mathcal{H}$ if

$$Pr[Algo(msg) = (msg_0, msg_1) : msg_0 \neq msg_1, H(msg_0) = H(msg_1)] \geq \epsilon,$$

where *Pr* is the probability over a random choice of $H()$ and random bits of $Algo$. We consider random selection of hash function from *SHA1*, *SHA256*, and *SHA384*. Our scheme is secure and provides collision resistant hashing. Alternatively, a hash chain can also be used [20].

Furthermore, the security of f_{cipher} depends upon $counter_i$ and SK (SK_{sn_i}/SK_{cn_i}). In the proposed scheme, $counter_i$ acts as a counter to enable Counter (CTR) mode-based ciphertext. Since $\mathcal{A}$ cannot guess the correct values of $lseqno$ and $lackno$ due to wireless transmission errors, it cannot compute correct $counter_i$. Furthermore, $\mathcal{A}$ does not know SK_{sn_i}/SK_{cn_i}, it can neither encrypt the plaintext with a correct pair of ($counter_i$, SK) nor decrypt the ciphertext successfully. We can compute an upper bound on the number of ciphertexts of a message that can be generated using a single $counter_i$ and SK. The standard definition of the confidentiality is given by the indistinguishabity from randomness. The ciphertexts generated in our scheme are considered secure against $\mathcal{A}$, as it cannot distinguish ciphertexts from the output of a truly random source. $\mathcal{A}$ has an advantage (difference of true positive probability and false positive probability), if it can distinguish them. We apply Bellare *et al.* [21] analysis to our scheme and assume that the encryption function is a Pseudo-Random Function (*PRF*), not a Peudo-Random Permutation (*PRP*). Then,

$$Adv[PRF](t,q) \leq Adv[PRP](t,q) + q^2 2^{-l-1},$$

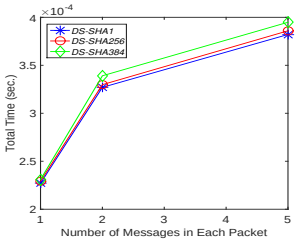
where q is the number of known plaintexts, t is the time used to perform attack, l is the bit-length of the ciphertext of a message, and the advantage when distinguishing mechanism *PRF/PRP* from random is denoted as Adv[*PRF/PRP*]. Considering Adv[*PRF*] = ϵ, Adv[*PRP*](t, q) = 0, and l = {160, 256, 384} (as the size of a secret key), we get $q = \{\sqrt{\epsilon}.2^{80.5}, \sqrt{\epsilon}.2^{128.5}, \sqrt{\epsilon}.2^{192.5}\}$. Hence, $\mathcal{A}$'s advantage is limited to ϵ, if we do not generate more than $q \approx \{2^{80}, 2^{128}, 2^{192}\}$. A true random source will produce two identical ciphertexts after generating a total of $\{2^{80}, 2^{128}, 2^{192}\}$ ciphertexts. Therefore, in order to generate indistinguishable plaintext, we use at most $\{2^{80}, 2^{128}, 2^{192}\}$ ciphertexts.

VI. Performance Evaluation and Experimental Analysis

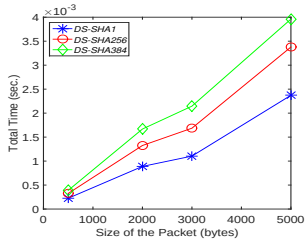
This section presents the performance evaluation and experimental analysis of the proposed scheme.

TABLE II: Parameters for Simulation Setup

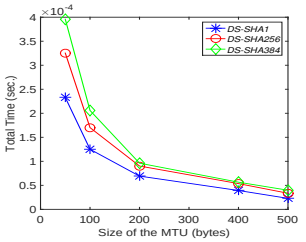
Parameters	Range Value	Unit
Baud Rate	100-9600	bits/s
Propagation Delay	10-500	ms
MTU	50-500	bytes
Packet Size	500-5000	bytes
Number of packets	1-5	–



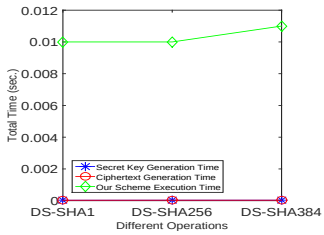
(a) *DS* generation with variable number of messages in each packet.



(b) *DS* generation with variable packet size, when propagation delay = 50 ms, *MTU* = 500 bytes.

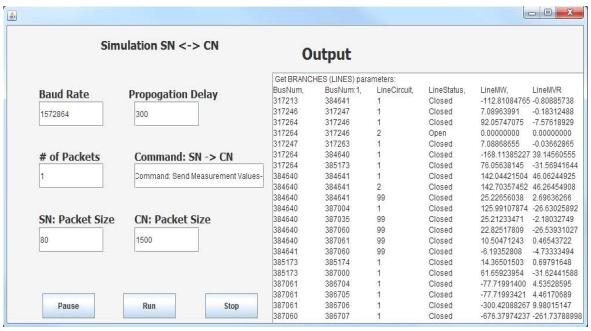


(c) *DS* generation with variable size of *MTU*, when propagation delay = 50 ms, *MTU* = 500 bytes.



(d) *SK* key and ciphertext generation, and computation of execution time with *DS-SHA1*, *DS-SHA256*, and *DS-SHA384*.

Fig. 4: Generation of dynamic secret, secret key, and ciphertext, and the total execution time of our scheme.



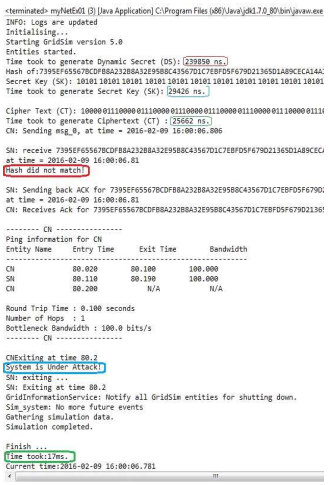
(a) GUI: sending data and outputting lines statistics.

Output

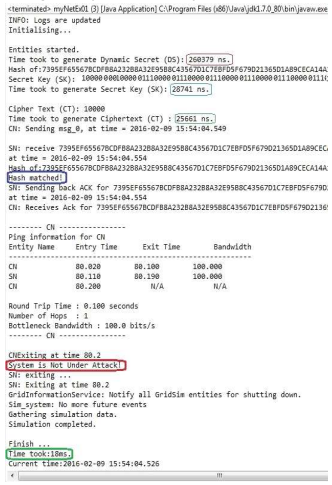
Get BRANCHES (LINES) parameters:

BusNum,	BusNum:1,	LineCircuit,	LineStatus,	LineMW,	LineMVR
317213	384641	1	Closed	-115.00329326	-0.83127227
317246	317247	1	Closed	6.98956457	-0.18846666
317264	317246	1	Closed	93.14012563	-7.70986159
317264	317246	2	Open	0.00000000	0.00000000
317247	317263	1	Closed	6.98865056	-0.03963427
317264	384640	1	Closed	-170.68176700	39.89319815
317264	385173	1	Closed	77.54164136	-32.18333673
384640	384641	1	Closed	144.77015121	46.85600468
384640	384641	2	Closed	145.44215755	47.06156544
384640	384641	99	Closed	25.70751844	2.73307433
384640	387004	1	Closed	123.67163723	-28.46526850
384640	387035	99	Closed	23.78311925	-2.36342843
384640	387060	99	Closed	28.09845856	-26.35149780
384640	387061	99	Closed	9.62781599	1.41030729
384641	387060	99	Closed	-5.46190944	-4.71017195
385173	385174	1	Closed	14.30853617	0.69262160
385173	387000	1	Closed	63.20016900	-32.22729315
387061	386704	1	Closed	-77.70999556	18.87093913
387061	386705	1	Closed	-77.71008722	18.79108233
387061	386706	1	Closed	-300.13473036	65.10331639
387060	386707	1	Closed	-500.57493353	-248.52177717

(b) Lines statistics are changes by adversary.



(c) A successful attempt of malicious data transmission.



(d) A successful attempt of legitimate data transmission.

Fig. 5: A GUI of our scheme: transmission of a malicious and legitimate measurement data from the *CN* to the *SN*.

A. Performance Evaluation

Our dynamic secret and encryption scheme requires two hash functions for generating dynamic secrets, two XOR operations for deriving secret keys, and two XOR and two addition operations for performing encryption and decryption of data at the control node as well as the supervisory node. In addition, two hash functions are required for message integrity over the communication network. In our scheme, we consider a set of hash algorithms, such as $\{SHA1, SHA256, SHA384\}$ for selecting a hash function $H()$.

B. Experimental Analysis

In this work, we develop an experimental setup with a co-simulator. The co-simulator uses JDK1.7 with Java, Java Agent Development Framework (JADE), MATLAB, and PowerWorld to implement communication scenarios between the control nodes and the supervisory node. Table II describes the selected ranges of communication parameters for our simulation, where C37.118 protocol supports packets with message passing [23].

We generate dynamic secrets by computing *SHA1*, *SHA256*, and *SHA384* hash functions. Thereafter, we generate secret keys and the ciphertexts of the data, and compute the total execution time of our scheme. Figure 4 shows simulation results in which dynamic secrets are obtained using all three hash functions, *i.e.*, *SHA1*, *SHA256*, and *SHA384* with (a) variable number of messages in each packet (range 1-5), (b) variable size of packets (range 500-5000 bytes), and (c) variable size of Maximum Transmission Unit (*MTU*) per frame (range 50-500 bytes). Furthermore, Figure 4 (d) computes a secret key, the ciphertext of the data, and the overall execution time of our scheme. The generation times for the dynamic secret by *SHA1*, *SHA256*, and *SHA384* are between 0.2 ms to 4 ms. The execution time of the proposed scheme is ranged between 0.01 sec to 0.011 sec. The results show that *DS-SHA1* outperforms *DS-SHA256* and *DS-SHA384*. We can prefer *DS-SHA1* till the end of the year 2016 as it will be no more secure and useful [24]. But, we also use other two functions to enable dynamic selection of a hash function in our scheme. Figure 5 presents a GUI for sending measurement data, change in line statistics by adversary, and successful attempts of malicious and legitimate data transmission.

VII. Conclusions

We have described an efficient and secure scheme for the communications between supervisory and control nodes in smart grid applications. Frequent generation of dynamic secrets and secret keys of the encryption scheme provides sufficient security to the measurement values that are sent from the control node to the supervisory node. The generation of dynamic secrets is based on the fact that the inevitable and unpredictable errors in the wireless communication prevents the system against tracing and obtaining the secrets by the adversary. The security analysis ensures that the proposed scheme defeats man-in-the-middle attacks, replay attacks, malicious injection communications, and impersonation attacks over the wireless network. Furthermore, performance evaluation and experimental analysis of our simulation show that the processes of generating dynamic secrets, secret keys, and ciphertexts are lightweight. It results in lower and manageable overheads, and enables fast execution. We use all three functions in our scheme to provide randomness of selecting hash function where *DS-SHA1* is efficient than *DS-SHA256* and *DS-SHA384*.

References

[1] Smart Grid, Office of Electricity Delivery & Energy Reliability, Energy.gov. [Online] http://energy.gov/oe/services/technology-development/smart-grid.

[2] X. Fang, S. Misra, G. Xue, and D. Yang, "Smart grid - the new and improved power grid: a survey," *IEEE Communications Surveys & Tutorials*, vol. 14, no. 4, pp. 944-980, 2012.

[3] A. Mahmooda, N. Javaida, and S. Razzaq, "A review of wireless communications for smart grid," *Renewable and Sustainable Energy Reviews*, vol. 41, pp. 248-260, 2015.

[4] V. C. Gngr, D. Sahin, T. Kocak, S. Ergt, C. Buccella, C. Cecati, and G. P. Hancke, "Smart grid technologies: communication technologies and standards," *IEEE Transactions on Industrial Informatics*, vol. 7, no. 4, pp. 529-539, 2011.

[5] W. Wang and Z. Lu, "Cyber security in the smart grid: survey and challenges," *Computer Networks*, vol. 57, pp. 1344-1371, 2013.

[6] 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.

[7] 1815-2010, IEEE Standard for Electric Power Systems Communications - Distributed Network Protocol (DNP3), Dist-1815 WG, 2010. [Online]. https://standards.ieee.org/ findstds/standard/1815-2010.html.

[8] DNP3 Secure Authentication V5, IEEE Standards - 1815-2011, Apr. 2012. [Online]. https://www.dnp.org/Lists/ Announcements/Attachments/7/ Secure Authentication v5 2011-11-08.pdf.

[9] 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.

[10] J. Hao, R. J. Piechocki, D. Kaleshi, W. H. Chin, and Z. Fan, "Sparse Malicious False Data Injection Attacks and Defense Mechanisms in Smart Grids," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 5, pp. 1-12, Oct. 2015.

[11] C. Capar, M. Zafer, D. Goeckel, D. Towsley, and D. Agrawal, "Physical-layer-enhanced wireless secret key exchange," in *Proc. Conference of the International Technology Alliance*, 2010, pp. 1-8.

[12] C. W. Wong, J. M. Shea, and T. F. Wong, "Secret sharing in fast fading channels based on reliability-based hybrid ARQ," in *Proc. MILCOM*, Nov. 2008, pp. 1-7.

[13] X. Sheng, G.Weibo, and D. Towsley, "Secure wireless communication with dynamic secrets," in *Proc. IEEE INFOCOM*, 2010, pp. 1-9.

[14] S. Xiao and W. Gong, "Wireless network security using randomness", U.S. Patent 8 204 224 B2, pp. 1-7, Jun. 2012.

[15] Y. Sun, Y. Mao, T. Liu, Y. Sun, Y. Liu, and X. Guan, "A dynamic secret-based encryption method in smart grids wireless communication," in *Proc. ISGT - Asia*, 2012, pp. 1-5.

[16] 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.

[17] S. Iyer, "Cyber security for smart grid, cryptography, and privacy," *Inter. J. of Digital Multimedia Broadcasting*, vol. 2011, pp. 1-8, 2011.

[18] N. Saputro and K. Akkaya, "Performance evaluation of smart grid data aggregation via homomorphic encryption," in *Proc. IEEE WCNC*, Apr. 2012, pp. 2945-2950.

[19] M. R. Clark and K. M. Hopkinson, "Transferable multiparty computation With applications to the smart grid," *IEEE Trans. on Info. Forensics & Security*, vol. 9, no. 9, pp. 1356-1366, Sep. 2014.

[20] M. Long and C.-H. Wu, "Energy-efficient and intrusion-resilient authentication for ubiquitous access to factory floor information," *IEEE Transactions on Industrial Informatics*, vol. 2, no. 1, pp. 40-47, 2006.

[21] M. Bellare, A. Desai, E. JokiPii, and P. Rogaway, "A concrete security treetment of symmetric encryption : analysis of the DES modes of operation," in *Proc. IEEE Symposium on Foundations of Computer Science*, 1997, pp. 1-32.

[22] What is Diffie-Hellman? RSA Laboratory. [Online]. http://www.emc.com/emc-plus/rsa-labs/standards-initiatives/what-is-diffie-hellman.htm.

[23] D. Anderson, C. Zhao, C. H. Hauser, V. Venkatasubramanian, D. E. Bakken, and A. Bose, "A virtual smart grid," *IEEE Power & Energy*, pp. 49-57, 2012. [Online]. http://magazine.ieee-pes.org/january-february-2012/a-virtual-smart-grid.

[24] Eduard Kovacs, New Collision Attack Lowers Cost of Breaking SHA1, Oct. 2015. [Online]. http://www.securityweek.com/new-collision-attack-lowers-cost-breaking-sha1.

Neetesh Saxena [S'10-M'14] is a Post-Doctoral Researcher at the Georgia Institute of Technology, USA. Prior to this, he was with the State University of New York Korea as a Post-Doctoral Researcher and a Visiting Scholar at Stony Brook University, USA. He earned his Ph.D. in Computer Science & Engineering from IIT Indore, India. In 2013-14, he was a Visiting Research Student and a DAAD Scholar at B-IT, Rheinische-Friedrich-Wilhelms Universitt, Bonn, Germany. He was also a TCS Research Scholar during Jan. 2012 - Apr. 2014. His current research interests include smart grid security, vehicle-to-grid security and privacy, cryptography, security and privacy in the cellular networks, and secure mobile applications. He has published several papers in international peer-reviewed journals and conferences. 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. Grijalvas graduate degrees in Electrical and Computer Engineering, M.Sc. (99), Ph.D. (02) are from the University of Illinois at Urbana-Champaign.