Abstract— Security is fundamental to the operation of smart grid and its advanced metering infrastructure (AMI). Key establishment is the core component in key management schemes and plays a crucial role in satisfying the security requirements. In recent years, many key establishment protocols have been proposed in the context of smart grids. However, their high overhead and complexity have undermined their ability to be practically employed in AMI. In this paper, we propose a novel identity-based key establishment protocol (NIKE) which employs elliptic curves at its core and has the lowest computational overhead among current secure protocols, especially at the meter side. We have given a detailed and comparative analysis of both security and computational burden for the proposed protocol as well as the previous efforts. We have evaluated the resilience of different protocols to well-known attacks by discussion, however, for the proposed protocol, we have also used AVISPA tool to formally verify its security. Compared to the previous solutions, the proposed protocol either is more secure or scores higher in performance benchmarks run on the same hardware.

Index Terms—Smart Grid, Advanced Metering Infrastructure, Key management, Identity-based Cryptography, AVISPA.

I. INTRODUCTION

TRADITIONAL power grids were centralized systems in which the power flow was unidirectional, from a few massive plants to consumers through transmission and distribution lines. The new idea of smart grids is a paradigm for distributed power generation and consumption which requires a complex two-way communication infrastructure, sustaining power flows between intelligent components.

The baby steps of creating a smart grid were taken when Automated Meter Reading (AMR) technologies were developed to reduce costs and improve the precision of meter readings. A growing consensus on the advantages of two-way interactions between system operators, consumers and their loads and resources made AMR evolve into something called the Advanced Metering Infrastructure (AMI) [1]. AMI is capable of providing consumers with energy management information (e.g., consumption data and outage reports) and other control information. It also enables devices to record the operation and security events and to collect power quality and load profile data [2]. As shown in Fig. 1, the major components of AMI are the smart meters (meter, for short), the data concentrators (collector), and finally the AMI Head-End (AHE) which is located in the operation center [1, 3].

Meter is a digital device that provides energy measurement, monitoring and control services, and is the communication gateway of consumer to the smart grid. It provides legacy AMR services including real-time measurement of energy consumption. AHE manages the data collection and facilitates the interaction with other operation and management systems [1].

The communications are carried out via either wired or wireless links such as Power Line Communication (PLC), Ethernet or RF ones. The messages exchanged over these links are exposed and thus, the AMI communication infrastructure is vulnerable to security threats. As Fig. 1 shows, messages are delivered via concentrators and possibly other meters. So in addition to the security concerns created by outsiders, one has to deal with the possible threats of malicious insiders too.

The basic solution usually prescribed for securing a system is to satisfy the confidentiality, integrity and availability constraints. The latter is more of a performance issue but the two other constraints are usually satisfied by cryptographic means which require secret keys. Therefore, the problem of securing the system is shrunk and translated to the problem of establishing and managing cryptographic keys.

A key management scheme consists of key generation, establishment, storage, updating and revocation [4]. However, among these, key establishment is the fundamental element. Many key establishment protocols have been proposed in the literature [4–19], and each has its own advantages. In the context of smart grids, the main focus is on security and overhead. A flaw in the security part renders a protocol totally useless. However, overhead is more of an efficiency matter. Some proposals have failed to satisfy the security requirements and are vulnerable to attacks [4, 10, 20]. Among those that have passed the security tests, high overhead, in terms of communication and computation load, is an issue [5, 9]. Despite the efforts made to lower the key establishment overhead, yet it is a challenging problem as many of the components in AMI (e.g., meters) have very low resources.

Authenticated Key Agreement (AKA) is a key establishment method enhanced to prevent active attacks and satisfy the security requirements by authenticating the participants at the same time the key is being established [21]. AKA can be realized using public-key infrastructure (PKI) [12–14] or identity-based cryptography [15–18]. PKI-based solutions suffer from heavy certificate management overhead. So identity-
A VISPA software is also provided. Section V concludes the paper. This paper is organized as follows: In Section II, a review of the previous efforts on key management in smart grids is presented. In Section III, our key establishment method is proposed. A detailed analysis of its security and computational burden is given in Section IV. Comprehensive comparisons have been made to show how much we have improved the key establishment performance with respect to the previous methods and compared them side by side with those of the proposed protocol. Moreover, we have thoroughly analyzed the security of the previous works and even presented some new attack scenarios. After making sure the proposed protocol resists all these attacks, we formally verified its security by an automated model verifier (AVISPA [22]).

This paper is organized as follows: In Section II, a review of the previous efforts on key management in smart grids is presented. In Section III, our key establishment method is proposed. A detailed analysis of its security and computational burden is given in Section IV. Comprehensive comparisons have been made to show how much we have improved the key establishment performance with respect to the previous protocols. The comparisons are further extended to also include security criteria such as resistance to well-known attacks. A formal security verification of the proposed protocol by AVISPA software is also provided. Section V concludes the paper and gives insights into future research directions.

II. RELATED WORK

Many key management schemes have been proposed for AMI so far and each has tried to achieve a specific security goal. Here, we review these efforts and critically analyze some. In [5], Nicanfar et al. put forward an authentication protocol for key establishment and installation of keys in newly-deployed meters. This authentication protocol was part of a bigger key management scheme designed for smart grids. However, we have found that [5] suffers from security flaws which make it vulnerable to the impersonation attack. In Appendix A we have thoroughly analyzed this protocol and have shown how the attack can be launched. We have also analyzed the communication and computation overhead of the protocol whose results are reported in Section IV.

Based on the X.1053 standard [11], Nicanfar et al. [6] later proposed a key exchange protocol. At its core, X.1053 is a classic Diffie-Hellman protocol. Nicanfar et al. mixed the notion of X.1053 with ECC and made a new key exchange protocol called EPAKE (ECC-based Password Authenticated Key Exchange). Although our analyses, both theoretically and by means of AVISPA model verifier [22], did not reveal any security issue with this protocol, we found that it has high computational costs, especially at the meter. We provide a detailed view to this protocol in Section IV and Appendix A.

In another try, Nicanfar et al. [7] again proposed a key agreement protocol for smart grids. But this time they adopted the Secure Remote Password (SRP) protocol [8] as their source of inspiration. There are a total of six message exchanges in SRP, but Nicanfar et al. managed to reduce them to three. This protocol requires undergoing several symmetric and asymmetric encryption operations which excessively increase the computational load. Moreover, using public key cryptography, especially in home applications where device resources are limited, is not efficient and creates significant delays.

Liu et al. [4] proposed another key management framework based on key graphs. They suggested having three management schemes for unicast, multicast, and broadcast modes of operation. They tried to use simple cryptographic algorithms for key generation and refreshing to cope with the storage and computation constraints of smart meters. However, Wan et al. [9] later analyzed their scheme and showed that it does not resist the desynchronization attack. Since the session key is computed based on the previous data, once an adversary blocks the data path, what is stored in the meter and the center will be different and thus, they will not be able to communicate with each other afterwards due to the mismatch of the keys.

Learning from Liu et al.’s experience, Wan et al. [9] proposed a new key establishment protocol called $SKM$. In this protocol, each meter uses an end-to-end key to agree with AHE on a session key, $SKM$, and its improved version $SKM^+$, both have very high computational costs. This protocol has been analyzed in Appendix A and its computational cost has been calculated in Section IV.

Wu and Zhou [20] constructed a key management scheme based on Kerberos standard combining both PKI and Trusted Authority (TA) to increase the security. In their scheme, three authentication protocols have been used for different parts of AMI. In each of them, a session key for the communicating entities is generated. This implies that key establishment is done at the middle of these authentication protocols.

Xia and Wang [23] analyzed Wu and Zhou’s protocol and proved that it is vulnerable to the man in the middle (MITM) attack. Then, they proposed an alternative key distribution scheme with low computational cost. However, their scheme required a higher number of message exchanges compared to the questioned scheme and this originated from employing TA in every run of the protocol. Later, even this work, which was supposed to fix the MITM issue, was proven to be vulnerable...
to the MITM and desynchronization attacks [7]. Kamto et al. [10] proposed a lightweight key management protocol based on Diffie-Hellman key agreement, somewhat similar to Nicanfar et al. ’s work [6]. However, it is not based on ECC and is not resistant to the MITM and desynchronization attacks either. We have analyzed this protocol in Appendix A. Lately, Benmalek and Challal [19] published a paper on a new key management scheme based on multi-group key graph techniques. However, the key establishment part of their scheme is based on the classic elliptic curve Diffie-Hellman protocol which is not much different from the previous works.

III. THE PROPOSED SCHEME

In this section, we present two new key establishment protocols of the same family which are tailored to the needs of smart grids. The proposed protocols take advantage of elliptic curve cryptosystems as well as the discrete logarithm problem.

A. The Proposed Protocol

As discussed in Section I, there are two ways to realize AKA. PKI suffers from the high certificate management load. Therefore, identity-based cryptography is a better candidate for smart grids with resource-limited components such as meters. However, since bilinear pairing [24] is always defined over a super singular elliptic curve group with a large element size, the operation time for pairings is even longer than RSA [25].

In the current subsection, an identity-based key establishment protocol for AMI will be proposed which does not rely on pairing. It consists of three steps: Setup, Installation, and Key Agreement whose descriptions are given below:

1) Setup: The grid owner takes a value for $k$ and considers it as the system parameter. Given $k$, TA does the following:
   - Chooses a $k$-bit prime $q$ and constructs $\{F, E/F, a, q, P\}$.
   - Chooses the master key $x \in Z_q^*$ and computes the system public key $P_{pub} = xP \in E/F_q$.
   - Chooses two hash functions $H_1 : \{0, 1\}^* \rightarrow Z_q^*$, and $H_2 : \{0, 1\}^* \rightarrow Z_q^*$.
   - Publishes $\{F, E/F, a, g, P, P_{pub}, H_1, H_2\}$ as the system parameters and keeps the master key $x$ secret for itself.

2) Installation: TA takes the system parameters and the master key $(x)$ as inputs and sends $AHE$s and meters their share of the private keys. First, TA goes through the following procedure for each AHE with the identifier $ID_{AHE}$.
   - AHE chooses the random number $r_{AHE} \in Z_q^*$, computes $R_{AHE} = r_{AHE}P$ and sends it along with $ID_{AHE}$ to TA over a secure channel.
   - The TA computes $y_{AHE} = H_1(ID_{AHE}, R_{AHE}).x$ and sends it back to AHE.
   - This private key is valid if the equation $(r_{AHE} + y_{AHE})P = R_{AHE} + H_1(ID_{AHE}, R_{AHE})P_{pub}$ holds.

In the second step, TA goes through the following procedure for each meter with the identifier $ID_i$:
   - Meter $i$ chooses the random number $r_i \in Z_q^*$, computes $R_i = r_iP$ and sends $ID_i$ to TA.
   - TA computes $y_i = H_2(ID_i, y_{AHE}).x$ and sends it back to the meter over a secure channel.
   - Meter $i$ computes $S_i = r_i + y_i$ and considers $S_i$ and $R_i$ as the private and public values, respectively.

At the end of these steps, each meter will have its own $\{S_M, R_M, y_M, r_M\}$ and each AHE will have $\{y_{AHE}, r_{AHE}\}$.

In practice, the factory making the meters and AHEs can play the role of TA. TA goes offline after the installation phase.

3) Key Agreement: Between AHE and a meter, an authenticated session key is established with three messages:

   Step 1. First, the meter chooses a random number $a \in Z_q^*$ and computes $T_M = aP$. Then, it sends the tuple $(T_M, ID_M, R_M)$ to AHE.

   Step 2. Upon receiving the initiation message from the meter, AHE does the following:
      1) Chooses the random number $b \in Z_q^*$ and calculates $T_x = b + r_{AHE}, T_{AHE} = T_xP$ and $k_{AHE-M} = T_x(R_M + H_2(ID_M, y_{AHE})P_{pub} + T_M)$.
      2) Computes $M_1 = H_1(0, k_{AHE-M})$ and sends $T_{AHE}, ID_{AHE}$ and $M_1$ to the meter.

   Step 3. The meter computes $k_{M\rightarrow AHE} = (S_M + a)T_{AHE}$, and $M'_1 = H_1(0, k_{M\rightarrow AHE})$. Then, it checks if $M'_1 = M_1$.
      If they are equal, the meter authenticates AHE and sets $K = H_1(ID_M||ID_{AHE}, k_{M\rightarrow AHE})$ as the session key.

   Step 4. Finally, the meter computes $M_2 = H_2(1, k_{M\rightarrow AHE})$ and sends it to AHE.

   Step 5. Upon receiving $M_2$, AHE computes $M'_2 = H_1(1, k_{AHE-M})$. If $M'_2 = M_2$, it authenticates the meter and sets $K = H_1(ID_M||ID_{AHE}, k_{AHE-M})$ as the session key.

In the next subsection, we present another variation of this protocol in which some of the load is shifted to AHE.

B. Decreasing the Computational Burden

It is highly desirable to reduce the computational burden at the meters side. The proposed protocol has this unique feature that can shift the processing load from one side to the other. In fact, 50% of the costly multiplications can be moved from the meter to AHE, leaving only one for the meter. This variation will be referred to as NIKE+ and is shown in Fig. 3:

   Step 1. First, the meter chooses the random number $a \in Z_q^*$ and computes $A = a + r_M$. Then, it sends $\{A, ID_M\}$ to AHE.
Step 2. Upon receiving the initiation message from the meter, AHE does the following:
1) Computes \( T'_M = AP \). Then it chooses a random number \( b \in Z^*_p \) and calculates \( T_a = b + r_{AHE} \), \( T_{AHE} = T_aP \) and \( k_{AHE-M} = T_a(T'_M + H_2(IDM, y_{AHE})P_{pub}) \).
2) Computes \( M_1 = H_1(0, k_{AHE-M}) \) and sends \( T_{AHE}, ID_{AHE} \) and \( M_1 \) back to the meter.

Step 3. The meter computes \( k_{M \rightarrow AHE} = (SM + a)TAHE \), and \( M'_1 = H_1(0, k_{M \rightarrow AHE}) \). If \( M'_1 = M_1 \), it authenticates AHE and sets \( K = H_1(IDM || ID_{AHE}, k_{AHE-M}) \) as the key.

Step 4. The meter computes \( M_2 = H_1(1, k_{M \rightarrow AHE}) \) and sends it to the other party.

Step 5. Upon receiving \( M_2 \), AHE computes \( M'_2 = H_1(1, k_{AHE-M}) \). If \( M'_2 = M_2 \), it authenticates the meter and takes \( K = H_1(IDM || ID_{AHE}, k_{AHE-M}) \) as the session key.

IV. SECURITY AND PERFORMANCE ANALYSIS

In this section, security of the proposed protocol is analyzed and its performance is compared with that of the competitors. In the analyses, it is assumed that the active attacker is capable of eavesdropping, modifying and also injecting messages.

A. Security Analysis by Discussion

1) Replay Attack: In a replay attack, the attacker eavesdrops the exchanged messages. Then, he may resend them at his discretion. In the proposed protocol, each party is challenging the other. Meter generates the fresh nonce \( \alpha \) and expects something in return in the second message that matches its expectation from the confidential key made based on it. AHE is doing the same with the freshly chosen \( \beta \). Therefore any try to convince either side to accept an old message constructed based on old nonces is thwarted. For example, imagine the attacker starts a new session as a meter. In the first step, he resends the set of \( IDM, RM \) and \( T_M \) (or \( A \)) from a previous run. AHE generates a random number and computes \( T'_M \rightarrow AHE \) and \( M''_1 \) that are different from the previous session. Since the attacker does not have \( \alpha \) and \( \beta \), he cannot compute \( M_2 \) and the session is terminated thus the replay attack is thwarted.

2) Impersonation and MITM Attacks: The attacker is capable of impersonating, if he can communicate as an authenticated party with the other. In our protocol, both parties are mutually authenticated via the asymmetrically pre-shared secrets put in their memories by TA; namely \( y_M \) and \( y_{AHE} \). Bear in mind that \( y_M = H_2(IDM, y_{AHE})x \). In the protocol, the common secret is effectively calculated as \( y_M \times P \) and \( H_2(IDM, y_{AHE}) \times P_{pub} \) by the meter and AHE, respectively. Even insiders like malicious AHE or meters cannot impersonate another entity as they do not have the right secret for that claimed entity. Therefore, impersonation is not possible.

In the MITM attack, the attacker secretly relays and possibly changes the communication between two parties so that they believe they are talking directly, while they are actually talking through the attacker. Note that by verifying \( M_1 = M'_1 \), the meter is actually verifying the authenticity of the received messages. \( M_1 \) has a dual-purpose role and also works as a message authentication code similar to the HMAC standard. AHE is doing the same by checking \( M_2 = M'_2 \). Mutual message authentication thwarts the MITM attack.

3) Desynchronization: An attacker could (partly) block messages transmitted between AHE and a meter to make them lose key synchronization permanently. As a result, they will no longer be able to communicate with each other. In our protocol, the (session) key is constructed based on the main secrets and freshly made nonces. We have not made any connection between the newly generated key and the previous session keys. Therefore, desynchronization is not possible and even if the third message is blocked, resynchronization can be regained at the cost of running the protocol once more.

The security features of our protocols have been compared with those of others against multiple criteria in Table I.

B. Formal Security Verification

Designing complex security protocols is not error-free, and finding their vulnerabilities is too difficult by trial and error. Therefore, similar to [6, 7], we decided to also formally verify the security of our protocol using the AVISPA tool [22] aiming to achieve these security goals:

1) Secrecy of the private values among relevant agents: \( y_M, k_{M \rightarrow AHE}, k_{AHE-M}, r_{AHE}, y_{AHE} \) and \( x \).

\[ \begin{array}{ccc}
\text{Meter} & \text{AHE} \\
\text{Generates: } \alpha & (SM + a)TAHE & \text{Generates: } b \in Z^*_p \\
A = a + r_M & T'_M = AP & T_a = b + r_{AHE} \\
& \text{AHE} & T_{AHE} = T_aP \\
& & k_{AHE-M} = T_a(T'_M + H_2(IDM, y_{AHE})P_{pub}) \\
& & M_1 = H_1(0, k_{AHE-M}) \\
\end{array} \]

\[ \begin{array}{ccc}
& & T_{AHE-M} \\
& & M_2 = H_1(1, k_{AHE-M}) \\
& & \text{check } M_2 \end{array} \]

\[ \begin{array}{ccc}
& & M'_2 = H_1(1, k_{AHE-M}) \\
& & \text{check } M'_2 \end{array} \]

\[ \begin{array}{ccc}
& & K = H_1(IDM || ID_{AHE}, k_{AHE-M}) \\
& & \text{ID_{AHE}, k_{AHE-M}} \end{array} \]

Fig. 3: \( NIKE^+ \): Variation 2 of the Proposed Key Establishment Protocol.

<table>
<thead>
<tr>
<th>TABLE III: The outputs of OFMC and CL-AtSe backends in AVISPA for ( NIKE^+ ) protocol.</th>
<th>OFMC Backend</th>
<th>CL-AtSe Backend</th>
</tr>
</thead>
<tbody>
<tr>
<td>% OFMC</td>
<td>%CL-AtSe</td>
<td>SUMMARY</td>
</tr>
<tr>
<td>SAFE</td>
<td>SAFE</td>
<td>DETAILS</td>
</tr>
<tr>
<td>BOUNDED_NUMBER_OF_SESSION</td>
<td>BOUNDED_NUMBER_OF_SESSION</td>
<td>TYPED_MODEL</td>
</tr>
<tr>
<td>PROTOCOL</td>
<td>PROTOCOL</td>
<td>/papers/nikeplus/nkp.if</td>
</tr>
<tr>
<td>GOAL</td>
<td>GOAL</td>
<td>as specified</td>
</tr>
<tr>
<td>BACKEND</td>
<td>BACKEND</td>
<td>OFMC</td>
</tr>
<tr>
<td>STATISTICS</td>
<td>STATISTICS</td>
<td>parseTime: 0.00s</td>
</tr>
<tr>
<td></td>
<td></td>
<td>searchTime: 0.12s</td>
</tr>
<tr>
<td></td>
<td></td>
<td>visitedNodes: 77 nodes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>depth: 8 plies</td>
</tr>
</tbody>
</table>
### Table I: Resilience of Key Establishment Methods to Attacks

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Replay Attack</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Impersonation Attack</td>
<td>✗</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>MITM Attack</td>
<td>✗</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>De-synchronization Attack</td>
<td>✗</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>

2) Authentication of AHE and meter at their partners’ sides.
3) Sameness of the computed session keys at the two sides. Since security-wise NIKE and NIKE* are equivalent, we only modelled NIKE* in HLPSL language and constructed an equivalent protocol for it subject to the constraints of AVISPA backends. Out of the four AVISPA backends only two support exponentiation which is a suitable replacement for ECC point multiplication. The output results of those two backends (OFMC and cl-AtSe) have been reported in Table III. Both acknowledge the correctness and safety of our protocol. Interested readers may find the complete HLPSL code along with further details in the paper supplementary material.

### Table II: Computational Costs of Different Key Establishment Methods

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Meter</th>
<th>AHE</th>
<th>TA or Relaying Meter</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Xia and Wang [23]</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nicanfar et al. [5]</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nicanfar et al. [7]</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nicanfar et al. [6]</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wan et al.’s SKM+ [9]</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wu &amp; Zhou* [20]</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kaimoto et al. [10]</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NTIE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NIKE*</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* Here, we have only analyzed the key establishment part of Nicanfar et al.’s two-part protocol since the other part was not relevant.

** Wu and Zhou’s protocol consists of two phases. The first phase includes two steps for generation of a shared key between a collector and the TA, and the second phase is used to obtain a session key between the collector and the aggregator. This table shows the computation cost of the first step assuming the collector is a meter.

Table II shows the computational costs of different key establishment protocols. Notice that we have not considered the above amortization trick in comparisons and assumed that all the calculations are carried out online. The notations are:
- $C_{RN}$: computation time of generating a random number
- $C_M$: computation time of a point multiplication
- $C_P$: computation time of pairing
- $C_{Es}$: computation time of a symmetric encryption
- $C_{Ds}$: computation time of a symmetric decryption
- $C_{Er}$: computation time of a public-key encryption
- $C_{Dr}$: computation time of a public-key decryption
- $C_{hash}$: computation time of a hash function
- $C_{Sig}$: computation time of a digital signature
- $C_{Ver}$: computation time of verifying a signature

In order to realistically compare the proposed protocols with the others in terms of computational cost, we took a MICAZ device as the meter and a 3GHz Pentium IV PC as AHE. With the processing times given in Table IV (adopted from [9, 26]), the computation times have been found as in Table V.

The table shows that Nicanfar et al.’s [5] has the longest computation time because of using public-key cryptography. The least amount is related to Xia and Wang’s protocol [23] due to adoption of lightweight cryptographic operations (like

### Table IV: Computation time of cryptographic operations

<table>
<thead>
<tr>
<th>Operation</th>
<th>MICAZ</th>
<th>Pentium IV 3GHz</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paring Operation</td>
<td>5.72s</td>
<td>3.88ms</td>
</tr>
<tr>
<td>Point Multiplication</td>
<td>2.45s</td>
<td>1.82ms</td>
</tr>
<tr>
<td>AES Encryption/Decryption</td>
<td>0.023ms</td>
<td>~0ms</td>
</tr>
<tr>
<td>Hash Function</td>
<td>0.023ms</td>
<td>~0ms</td>
</tr>
<tr>
<td>Public Encryption</td>
<td>0.79s</td>
<td>0.57ms</td>
</tr>
<tr>
<td>Public Decryption</td>
<td>21.5s</td>
<td>16ms</td>
</tr>
<tr>
<td>Sign()</td>
<td>21.5s</td>
<td>16ms</td>
</tr>
<tr>
<td>Verify Signature()</td>
<td>0.79s</td>
<td>0.57ms</td>
</tr>
</tbody>
</table>

1) please visit the authors’ page at anslab.org for further material.
TABLE V: Comparison of the protocols computation times

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Meter</th>
<th>AHE</th>
<th>TA or relaying meter</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Xia &amp; Wang’s [23]</td>
<td>0.80s</td>
<td>0.37ms</td>
<td>1.04ms</td>
<td>0.80s</td>
</tr>
<tr>
<td>Nicanfar et al.’s [5]</td>
<td>1.09s</td>
<td>48.7ms</td>
<td>51.8s</td>
<td>44.4s</td>
</tr>
<tr>
<td>Nicanfar et al.’s [7]</td>
<td>25.4s</td>
<td>34.2ms</td>
<td>NA</td>
<td>29.4s</td>
</tr>
<tr>
<td>Nicanfar et al.’s [6]</td>
<td>4.9s</td>
<td>5ms</td>
<td>NA</td>
<td>4.91s</td>
</tr>
<tr>
<td>Nicanfar et al.’s [9]</td>
<td>7.78s</td>
<td>5.7ms</td>
<td>NA</td>
<td>7.78s</td>
</tr>
<tr>
<td>Nicanfar et al.’s [5]</td>
<td>7.35s</td>
<td>9.29ms</td>
<td>NA</td>
<td>7.36s</td>
</tr>
<tr>
<td>Wu and Zhou’s [20]</td>
<td>4.85s</td>
<td>NA</td>
<td>17.01ms</td>
<td>43.8s</td>
</tr>
<tr>
<td>Nicanfar et al.’s [10]</td>
<td>4.9s</td>
<td>5.46ms</td>
<td>NA</td>
<td>4.91s</td>
</tr>
<tr>
<td>NIKE</td>
<td>2.45s</td>
<td>7.28ms</td>
<td>NA</td>
<td>2.46s</td>
</tr>
<tr>
<td>NIKE +</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table shows the computation times at the meter side are much longer than the AHE side. AHE usually does not have the memory and computational limitations of its counterparts. Therefore, a wise protocol designer usually tends to somehow transfer the computations from meter to AHE. For example, in the protocol of [5] protocol, initially SKM was proposed, which had a high computational cost at the meter side, but then, by transferring some of the computations to AHE, a new protocol (called SKM + ) was created. However, this shift merely saved us 0.42 seconds as reported in Table V.

NIKE was designed with a similar motive in mind, though its construction is fundamentally different. Its flexibility allowed transferring half of the multiplication load to AHE and the result was NIKE + . Compared to SKM, the reduction of computation time is quite noticeable (see Table V).

2) Communication Cost: The proposed protocols require three communications between a meter and AHE. These comprise the exchange of seven and six messages in NIKE and NIKE + , respectively. The communication costs of all the protocols are listed in Table VI. The highest is nine which belongs to Nicanfar et al.’s protocol [5]. The least is six and pertains to Wu and Zhou’s [20]. Although Wu and Zhou and Nicanfar et al.’s [9] protocols look better than ours in terms of communication cost, they fall short of other required criteria. Wu and Zhou’s protocol cannot resist the MITM attack as explained in Section II. Wu et al.’s SKM and SKM + do not have security problems. However, as described above, they are by far less efficient than the proposed protocols.

### Appendix A

**Security Analysis of Key Management Schemes**

In this section, we will analyze four recent key establishment protocols proposed in the context of smart grids with the aim of revealing their weaknesses. These analyses are genuine and have been done as a part of the work at hand.

A. Nicanfar et al.’s Protocol [5]

In [5], Nicanfar et al. proposed a key establishment protocol which we found is not resistant to the impersonation attack carried out by a malicious meter. First, we review the protocol:

1) Review of the Protocol: The protocol of [5] consists of six steps as illustrated in Fig. 4. Initially, the new meter sends the session key (key1), ID_M and TS1 to AHE via the Authentication Agent (AG). These are encrypted first by pubk_AG and then decrypted and re-encrypted by pubk_AHE when AG relays the messages. Note that AG can be one of the previously installed meters. In the next step, AHE transfers the meter Serial Number (SN_M) to ID_AHE to AG and the meter, respectively. In the third step, the meter transfers SN_M to the AG using key1. AG will authenticate the meter and AHE, if the serial numbers received from them are equal. Then, AG sends two encrypted packets to both parties. In fifth step, the meter sends a private key request to AHE that includes ID_M, TS1 and a new session key (key2). Finally, AHE sends the initial and current meter private keys encrypted with key2 to the meter. For more details, one can refer to [5].

![Fig. 4: Nicanfar et al.’s [5] authentication protocol](image-url)
2) Weaknesses of the Protocol: We managed to spot security weaknesses in Nicanfar et al.’s [5] work which make it vulnerable to impersonation attacks.

SM Impersonation: Nicanfar et al.’s authentication protocol relies on the meters that have been installed previously. Once a new meter is authenticated by AG and AHE, it receives a confirmation (called the success message) which is like e_{c_{k_{j+1}}}(ID_{AHE},TS_1,OK,F_{c_{j+1}},C_{n}). After the new meter verifies the success message, it sends e_{pubk_{AHE}}(ID_{M},TS_1,k_{j+1}) to AHE. If AG is malicious, it can hinder the delivery of this message and replace it with e_{pubk_{AHE}}(ID_{M},TS_1,k_{j}). AG knows ID_{M} and TS_1 and can fabricate any message and send it in the place of e_{pubk_{AHE}}(ID_{M},TS_1,k_{j}). Consequently, AHE encrypts a response with k_{j} that only AG can decrypt. Finally, the malicious AG decrypts the AHE message and obtains the parameters of the new meter and then communicates with AHE impersonating the new meter. This way, one meter can impersonate all the meters that are installed through it.

B. Kamto et al.’s Protocol [10]

Kamto et al. proposed a lightweight key management protocol based on Diffie-Hellman which we found is not resistant to the MITM and desynchronization attacks.

1) Review of the Protocol: In Kamto et al.’s protocol (Fig. 5), the gateway (considered as AHE) and a neighboring node (considered as M) initiate the Diffie-Hellman protocol. AHE and the meter select two random numbers x_{AHE} and x_{M} in a multiplicative group and compute the corresponding public values X_{AHE} and X_{M}. First, they concatenate their IDs with these public values, and then, compute the hash of those. Second, they exchange messages that are made by concatenation of their hash values, their IDs, and their public values. Upon receiving the messages, both will do message authentication. Then, each computes the key and finally, they transfer the hash values of the generated keys for confirmation.

The key is updated by k_{j+1} = H(k_{j}) relation every time the two parties exchange a message. Once done, k_{j} is destroyed.

2) Weaknesses of the Protocol:

MITM Attack: When m_{AHE} is sent to the meter, an adversary can block it and send m_{AHE} instead. In m_{AHE}, ID_{AHE} is not encrypted, therefore, the adversary can put together a new set of X_{AHE}, h_{AHE} and m_{AHE}. In the second phase, the meter sends m_{M} to AHE but the adversary can change it similarly. The changed message is verified by AHE because h_{M}' is made of valid AHE. At the end, the adversary has established a session key with AHE (i.e. k = a^{x_{AHE}}) and another one with the meter (i.e. k = a^{x_{SM}}).

Desynchronization Attack: The two parties update the session key by replacing k_{j} with k_{j+1} = H(k_{j}) upon exchange of any message. Once one party updates its key, sends a message as a signal to the other. An adversary can drop this message and the other party will never know of this change. So they will not be able to communicate with each other anymore.

C. Wan et al.’s Protocol [9]

1) Review of the Protocol: Wan et al. [9] proposed a key establishment method called SKM. In the first step, the meter generates the random number a and computes aH(ID_{M}) and sends it to AHE. AHE similarly computes bH(ID_{AHE}) along with K_{AHE}, K_{AHE} and a message authentication code M_{1} as shown in Fig. 6 and replies. Note that e(.) is a bilinear mapping operation, s is the private share of TA and sH(ID_{1}) is the private key of i. Upon receiving the reply, the meter generates the same keys and authenticates AHE. Finally, the meter send M_{2} to AHE to complete the mutual verification.

2) Analysis of the Protocol: This protocol uses pairing that is multiple times more time-consuming than exponentiation. Wan et al. tried to partly transfer the load from meter to AHE in SKM'. Although this transfer reduced the meter load, it was trivial and barely had any practical effect. Table V shows that the meter computation times of SKM and SKM' are 7.77s and 7.35s respectively thus the saving made is negligible.


Nicanfar et al. [6] took the RSA-based X.1053 [11] and turned it into an ECC-based key exchange protocol (Fig. 7).

1) Review of the Protocol: First, Alice (meter) generates a random number d_{A} and multiplies it by the group generator G to obtain Q_{A} = d_{A}G = (x_{A},y_{A}), and then finds X = E_{H(P)}(Q_{A}), where P is a pre-shared password. Finally, she sends X to Bob (AHE). Similar to Alice, Bob computes Q_{B} = d_{B}G = (x_{B},y_{B}). Using H(P), Bob decrypts X and computes Q_{AB} = d_{B}Q_{A} = (x_{AB},y_{AB}), S_{B} = H(P||y_{A}||y_{B}) and Y = E_{H(P)}(Q_{B}). Then, Bob sends S_{B} and Y to Alice. Upon receiving them, she computes Q_{AB} = d_{A}Q_{B} and S_{A} = H(P||y_{B}||y_{AB}). If S_{A} = S_{B}, she authenticates Bob and sends T_{A} = H(P||x_{A}||x_{B}||x_{AB}) to him. Alice computes K at her side. Bob computes T_{B} =
time is much lower in meter’s shoulder and this load cannot be shifted to AHE side to two point multiplications in each run. As Table I shows, he authenticates Alice and computes $K$.

Fig. 7: Nicanfar et al.’s [6] key establishment protocol $H(P|x_a||x_b||x_{ab})$ and compares it with $T_A$, if they are equal, he authenticates Alice and computes $K$ similarly.

2) Analysis of the Protocol: Both parties have to undergo two point multiplications in each run. As Table I shows, there is no security flaw in the protocol. However, compared to NIKE$, it puts much more computational load on the meter’s shoulder and this load cannot be shifted to AHE side either. Although NIKE has one more multiplication, only one of them should be done by the meter. Since AHE is resource-limited, in a practical setup, the overall computation time is much lower in NIKE (see Table V).

REFERENCES
Amin Mohammadali received the B.Sc. degree in electrical engineering from KIAU University, Alborz, Iran, in 2008 and the M.Sc. degree in communication engineering from IHU University, Tehran, Iran, in 2011. Ever since, he has been working in R&D positions for industry leading companies. He is currently a senior research scientist at the Department of Renewable Energy, Monenco, Iran. His research interests include smart grid security as well as interoperability, wireless network security, remote frequency identification, Internet of Things security, and privacy-preserving algorithms.

Mohammad Sayad Haghighi is an assistant professor at the School of Electrical and Computer Engineering, University of Tehran, Iran. Prior to that, he was an assistant professor at Iran Telecom Research Center. From 2009 to date, he has held research positions with Australian universities. He has done his postdoctoral studies at Deakin University, Australia and he is also holding a research fellowship position with Macquarie University, Australia now. He has also served as a program committee member of many conferences such as IEEE WNS, IEEE SICK and IEEE LCN. Dr Sayad is the director of Advanced Networking and Cyber Security research Lab (ANSLab). He has had several positions in industry too. He has worked in senior positions as an IT architect as well as a consultant. His research interests include both distributed networks and network security.

Mohammad Hesam Tadayon received the M.Sc. degree in mathematics from the University of Tarbiat Modares, Tehran, Iran, in 1997, and the Ph.D. degree in applied mathematics (coding and cryptography) from the University of Tarbiat Moallem of Tehran (Kharazmi), Tehran, Iran, in 2008. He has been holding an Assistant Professorship position with Iran Telecommunication Research Center (ITRC) since 2008. He is the director of IT Systems group at ITRC and also a member of national councils in the Iranian Ministry of Science and Technology. He has served in many research and industrial projects. His research interests include information theory, error-control coding and data security.

Alireza Mohammadi-Nodooshan received his B.S. degree in Computer Engineering from Isfahan University of Technology, Iran and his M.S. degree (with highest honors) in Information Technology from K. N. Toosi University of Technology, Iran. He has been an academic board member and an instructor at the Department of Computer Engineering, Vali-e-Asr University, Rafsanjan, Iran since 2013. His domains of interest include VoIP signaling protocols, design and formal analysis of computer security protocols, VANET broadcast storm mitigation techniques, optimization and metaheuristic algorithms, data mining and its applications in business intelligence. He has published many quality papers in these areas among which are the best paper of the computer security track at CSICC 2008 and the best poster paper of ICNRAECE 2016.