Check for updates

Received: 3 January 2019 | Revised: 6 May 2019 | Accepted: 4 July 2019

DOI: 10.1002/dac.4137

RESEARCH ARTICLE

WILEY

An enhanced anonymous identity-based key agreement protocol for smart grid advanced metering infrastructure

Khalid Mahmood[1] | **Jehangir Arshad**[2] | **Shehzad Ashraf Chaudhry**[3] | **Saru Kumari**[4]

[1]Department of Computer Science, COMSATS University Islamabad, Sahiwal Campus, Sahiwal, Pakistan

[2]Department of Electrical and Computer Engineering, COMSATS University Islamabad, Sahiwal Campus, Sahiwal, Pakistan

[3]Department of Computer Engineering, Faculty of Engineering and Architecture, Istanbul Gelisim University Istanbul, Turkey

[4]Department of Mathematics, Ch. Charan Singh University, Meerut, India

Correspondence
Shehzad Ashraf Chaudhry, Department of Computer Engineering, Faculty of Engineering and Architecture, Istanbul Gelisim University Istanbul, Turkey.
Email: ashraf.shehzad.ch@gmail.com

Summary

Sprouting populace mass within the urban areas furnishes critical challenges of providing uninterruptible community services to fulfill the primitive needs of inhabitants in smart cities. Smart cities facilitate and uplift the living standards of inhabitants through various smart systems or infrastructures, and smart grid is one of them. Secure transmission is a key requirement in the advanced metering infrastructure (AMI) of most smart grids, and key establishment cryptographic protocols can be used to achieve such a requirement. Designing efficient and secure key establishment protocols for AMI remains challenging. For example, in this paper, we reveal several weaknesses in the identity-based key establishment protocol of Mohammadali et al (published in IEEE Trans Smart Grid, 2017), which is based on elliptic curves. We then improve their protocol and prove its security in the random oracle model. We also demonstrate that the improved protocol achieves both anonymity and untraceability, before presenting a comparative summary of the security and computational overheads of the proposed protocol and several other existing protocols.

KEYWORDS

advanced metering infrastructure, anonymity, identity-based key establishment, key establishment, smart grid

1 | INTRODUCTION

Urban areas are becoming populous as they were never before, and it is expected that by the end of 2050, more than 70% of the global population will be living in cities. In this context, expanding and introducing new cities is not a solution to cope with scarce resources and services. Therefore, to facilitate every inhabitant with adequate resources and services, we need to make the cities smart. Smart city environment enables the inhabitant to enjoy more managed and efficient social and community services through advanced information and communication technologies and Internet of Things (IoT). The smart city is the collection of smart and intelligent systems and infrastructures such as smart health care, smart traffic system, smart parking system, smart supply chain management system, and smart grid infrastructure.[1]

Conventional power grid systems are generally integrated systems that support the unidirectional flow of power from power generation plants to end users over power distribution lines. Increasingly, power grid systems are getting "smarter," and such smart grid systems generally have a bidirectional communication structure and support power movements among intelligent components of the system. The advanced metering infrastructure (AMI; also known as automated meter reading—AMR—system in the literature) is a key component of a smart grid system[2]—see also Figure 1. The networks of AMI are considered as the vital components for smart grids because they enable bidirectional communications

 © 2019 John Wiley & Sons, Ltd.

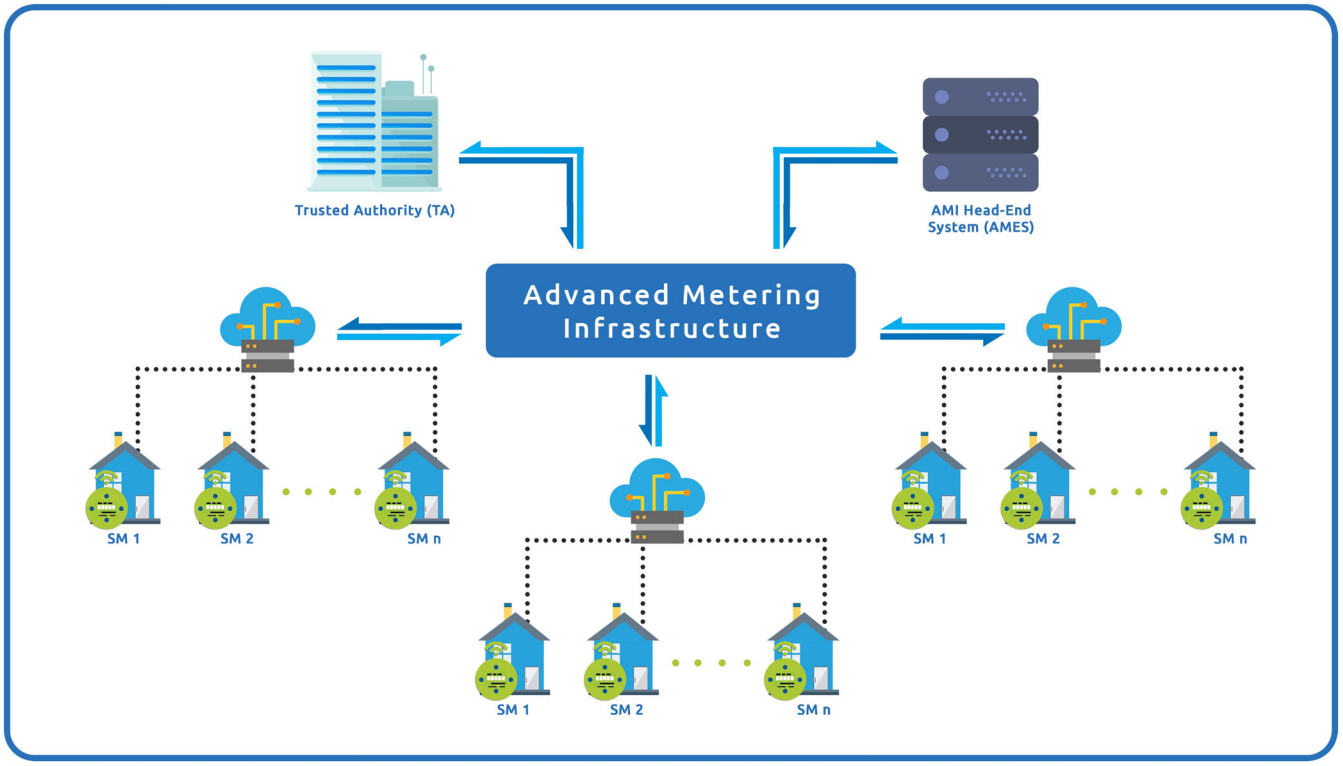


FIGURE 1 An advanced metering infrastructure (AMI) example

between consumers and power suppliers. This feature facilitates real-time coordination and monitoring at both ends, ie, power suppliers and consumers. The power supplier can perform real-time monitoring of power outages, fluctuations, power quality, and peak hours. Moreover, it enables them to coordinate announcements, policies, reports, bills, and rates with the consumers. Similarly, it also facilitates consumer to check and monitor their energy consumption patterns and respond to power suppliers, if necessary. However, the AMI applications demand more reliability at the cost of low latency.[3] AMI can facilitate business activities such as energy management (eg, detailed breakdown of power consumption by households, subdivisions, divisions, and county) and achieving quality of service (QoS).[4]

In an AMI, there are hundreds to thousands or tens of thousands of smart meters, depending on the scale of the smart grid system. Such smart meters collect information such as power consumption and send them over to the smart grid provider for billing and other purposes (eg, data analytics). In other words, smart meters are key to real-time measurement of power consumption. Moreover, AMI Head-End (AHE) handles the collection of arrival data and enables the communication with management systems and the rest of the operations.[5] In smart grid systems, all communication processes are conducted over wired links (eg, power line communication [PLC] or Ethernet) or over wireless connections (eg, GSM or radio frequency). Similar to any other communication systems, the ability to secure data exchange in the smart grid system is crucial.

Potential attack vectors include both external attackers and internal attackers (eg, malicious insiders). Generally, key management or key establishment cryptographic protocols are used to provide a secure communication channel.[6] There have been a large number of key establishment protocols proposed in the literature, and examples include those presented in the previous studies.[6-20]

In smart grid systems, key considerations for the deployment of any cryptographic and solution solution are computation and communication overheads. Existing protocols are generally not suited for smart grid deployment, either due to insecurity[6,7,9] or large overheads in terms of computation and communication.[10,13] There have been also attempts to reduce the overheads in key establishment protocols. For example, in Mohammadali et al,[21] an identity-based key establishment protocol for AMI was proposed, based on elliptic curve cryptography (ECC). The protocol has two variants, with a slight difference in terms of processing and computation steps. The authors claimed that their protocol is designed in such a way that it does not involve intensive computation especially at the meter end. However, we reveal several

10991131, 2019, 16, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/dac.4137 by Istanbul Gelisim Universitesi, Wiley Online Library on [03/01/2025]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

weaknesses in both variants (see Section 4) and explain how these weaknesses can be addressed (see Section 6). We also evaluate the security and performance of the improved protocol in Section 8.

In the next two sections, we review related literature and the relevant background materials.

2 | RELATED LITERATURE

In this section, we briefly review several related protocols designed for AMI. For example, as early as 2011, Nicanfar et al[10] presented a protocol to achieve key establishment and key installation in smart meters. However, Mohammadali et al[21] demonstrated that this protocol is vulnerable to impersonation attacks.

Nicanfar et al[11] proposed a key interchange protocol based on the X.1053 standard.[22] As the X.1053 is a standard Diffie-Hellman scheme, Nicanfar and Leung[11] substituted ECC for Diffie-Hellman scheme in their proposed key interchange protocol, referred to as ECC-based Password Authenticated Key Exchange (EPAKE). In a later work, Nicanfar et al[12] adapted the secure remote password (SRP) scheme from Wu TD et al[23] in their protocol. Specifically, they reduced the six-message exchange cycles in SRP to a three-message exchange. Despite having a reduced message exchange, the proposed protocol has a significant computational cost.

Liu et al[6] proposed a key managing framework, based on key graphs. The authors suggested taking three management schemes, namely, one for unicast, one for multicast, and for broadcast operation modes. However, it was revealed in Wan et al[13] that the approach of Liu et al is not secure against desynchronization attacks. As a session key is calculated on the basis of prior data, and when the data path is blocked by adversary, then the parties involved will not be able to connect with each other because of key mismatch.[13] Wan et al[13] then suggested an alternative key-formation scheme, SKM. Both SKM and its enhanced version SKM + incur high computational costs.

Wu and Zhou[9] also presented a key managing scheme built upon the Kerberos standard and leverages a trusted authority (TA) and the public key infrastructure (PKI). The scheme comprises three different authentication protocols, which facilitate the generation of an individual session key for the communicating individuals. However, it was subsequently revealed in Xia and Wang[24] that the scheme is susceptible to man-in-the-middle (MiTM) attacks. Xia and Wang[24] then proposed a key dissemination scheme, but this scheme requires a greater number of message exchanges as compared with the prior scheme. Again, the improved scheme is revealed to be insecure against MiTM and desynchronization attacks in Nicanfar et al.[12]

These examples highlight the challenges in designing secure and efficient cryptographic protocols.

3 | PRELIMINARIES

Table 1 summarizes the notations used in this paper.

3.1 | Adversarial threat model

In the adversarial threat model, there is a probabilistic polynomial time (PPT) adversary $\mathcal{A}$ with the following capabilities:

1. $\mathcal{A}$ can intercept and read the intercepted messages between participants. If the intercepted messages are not encrypted, then $\mathcal{A}$ can also modify the messages.
2. $\mathcal{A}$ is able to extract secret credentials stored in the memory of the target users via some explicit insider attacks.[25]

TABLE 1 Notation summary

Notation	Description
$\mathcal{SM}_i$, $\mathcal{HE}_j$	Smart Meter, *AMI*-head-end system (AHES)
ID_i	Identity of $\mathcal{SM}_i$
ID_j	Identity of $\mathcal{HE}_j$
$H(.)$, $H_1(.)$, $H_2(.)$	Three one-way hash functions
s	The master private key of *TA*
P_k	The master public key of *TA*
s_i, R_i	The key pair of $\mathcal{SM}_i$
s_j, R_j	The key pair of $\mathcal{HE}_j$
t_i, t_j	The time stamps

10991131, 2019, 16, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/dac.4137 by Istanbul Gelisim Universitesi, Wiley Online Library on [03/01/2025]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

Hence, a secure authentication protocol should ensure if some secret information (eg, session-specific secrets, private key, or session key) are compromised, and other secret information of the communicating user(s) are minimally affected.

In the Canetti-Krawczyk (CK) adversarial threat model,[25] the capabilities of the adversary are captured using session state reveal, party corruption, and session key query, each with different level of information exposed. For example, the schemes of Mohammadali et al are not secure against session state reveal attacks.

4 | REVIEW OF MOHAMMADALI ET AL'S SCHEMES

In the protocol of Mohammadali et al,[21] shown in Figures 2 and 3, there are three phases as follows (see Sections 4.1 to 4.3):

4.1 | Trusted authority setup phase

The smart grid provider initializes k and sets it as an intrinsic parameter of the system. Then, (TA) proceeds as follows:

- TA takes k-bit prime q and establishes $\{F_q, G_q, P, E/F_q, \}$.
- TA embeds its master secret key $s \in {Z_q}^*$ keeps it confidential and derives its corresponding public key $P_k = sP \in E/F_q$.
- Selects hash functions $H_1 : \{0,1\}^* \times G_q \rightarrow {Z_q}^*$ and $H_2 : \{0,1\}^* \times {Z_q}^* \rightarrow {Z_q}^*$
- TA publicizes the system parameters $\{F_q, G_q, P, P_k, E/F_q, H_1, H_2\}$

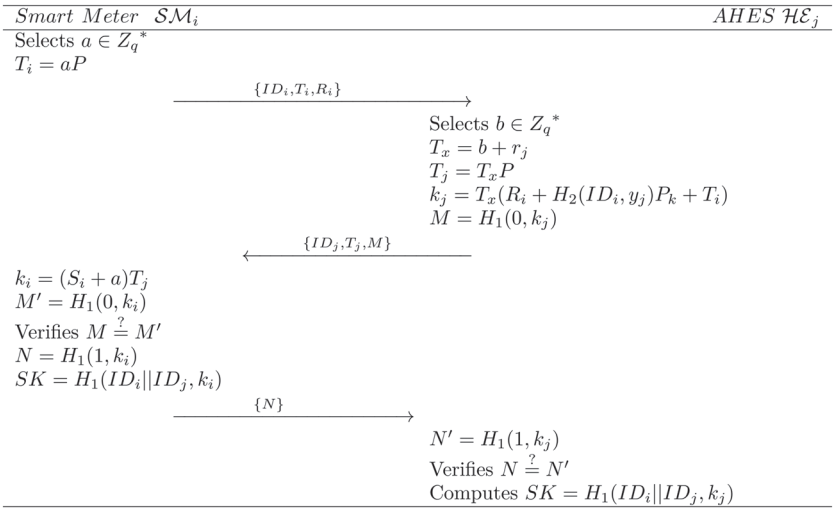


FIGURE 2 Mohammadali et al's first variant[21]

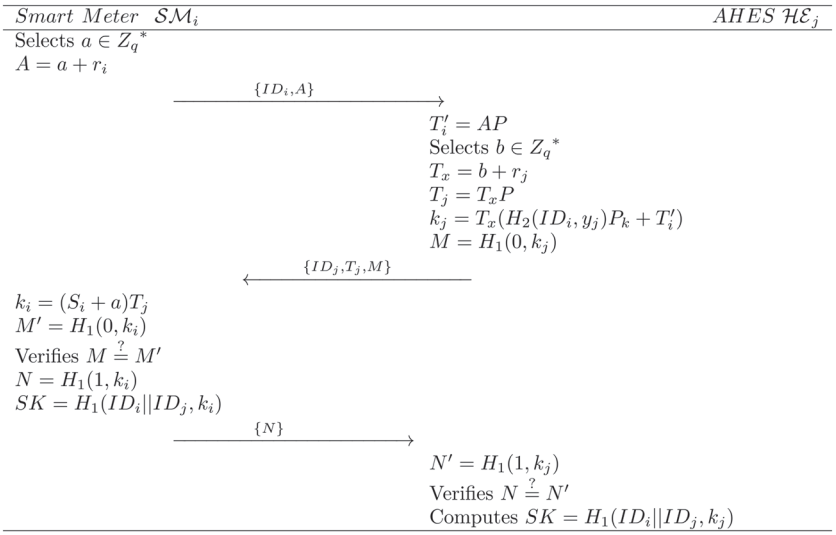


FIGURE 3 Mohammadali et al's second variant[21]

10991131, 2019, 16, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/dac.4137 by Istanbul Gelisim Universitesi, Wiley Online Library on [03/01/2025]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

4.2 | Registration phase

This phase explains how (*AMI*)-head-end system (*AHES*) and smart meter register with *TA*.

4.2.1 | AHES registration

- AHES HE_j selects a random number $r_j \in Z_q{}^*$, computes $R_j = r_jP$, and sends both ID_j and R_j as a registration request over a secure communication channel.
- *TA* receives the registration request and computes and sends $y_j = H_1(ID_j, R_j).s$ in response to the registration request from HE_j.
- If $(r_j + y_j)P \stackrel{?}{=} R_j + H_1(ID_j, Rj)P_k$ holds, then the secret key is considered valid.

At the conclusion of the registration process, AHES will have $\{y_j, r_j\}$.

4.2.2 | Smart meter registration

- Each smart meter SM_i selects a random number $r_i \in Z_q{}^*$, computes $R_i = r_iP$, and sends both ID_i and R_i as a registration request over a secure communication channel.
- *TA* receives the registration request and computes and sends $y_i = H_2(ID_i, y_j).s$ in response to the registration request from SM_i.
- SM_i computes $S_i = r_i + y_i$ and denotes R_i and S_i as the public and private keys, respectively.

At the conclusion of the registration process, each smart meter will have $\{S_i, R_i, y_i, r_i\}$.

4.3 | Key agreement phase

4.3.1 | First variant

The key agreement phase consists of a three-message round, as follows:

1. $\mathcal{SM}_i$ selects an arbitrary number $a \in Z_q{}^*$ and computes $T_i = aP$. $\mathcal{SM}_i$ sends a tuple $\{ID_i, T_i, R_i\}$ to $\mathcal{HE}_j$.
2. In response to the initial message from $\mathcal{SM}_i$, $\mathcal{HE}_j$ selects $b \in Z_q{}^*$ and computes $T_x = b + r_j$, $T_j = T_xP$, $k_j = T_x(R_i + H_2(ID_i, y_j)P_k + T_i)$ and $M = H_1(0, k_j)$. Then, $\mathcal{HE}_j$ sends $\{ID_j, T_j, M\}$ to $\mathcal{SM}_i$.
3. $\mathcal{SM}_i$ calculates $k_i = (S_i + a)T_j$, $M^{'} = H_1(0, k_i)$ and verifies $M \stackrel{?}{=} M'$. $\mathcal{SM}_i$ authenticates $\mathcal{HE}_j$ if the verification holds and sets $SK = H_1(ID_i||ID_j, k_i)$. $\mathcal{SM}_i$ computes $N = H_1(1, k_i)$ and transmits it to $\mathcal{HE}_j$.
4. After receiving $\{N\}$, $\mathcal{HE}_j$ computes $N^{'} = H_1(1, k_j)$ and checks $N \stackrel{?}{=} N'$. If the latter holds, then $\mathcal{HE}_j$ authenticates $\mathcal{SM}_i$ and computes the session key $SK = H_1(ID_i||ID_j, k_j)$

4.3.2 | Second variant

The key agreement phase in the second variant also utilizes three messages to establish an authenticated session key between $\mathcal{SM}_i$ and $\mathcal{HE}_j$, which is as follows:

1. $\mathcal{SM}_i$ selects an arbitrary number $a \in Z_q{}^*$ and computes $A = a + r_i$. $\mathcal{SM}_i$ sends a tuple $\{ID_i, A\}$ to $\mathcal{HE}_j$.
2. In response to the initial message from $\mathcal{SM}_i$, $\mathcal{HE}_j$ computes $T_i' = AP$, $b \in Z_q{}^*$, $T_x = b + r_j$, $T_j = T_xP$, $k_j = T_x(H_2(ID_i, y_j)P_k + T_i')$ and $M = H_1(0, k_j)$. Then, $\mathcal{HE}_j$ sends $\{ID_j, T_j, M\}$ to $\mathcal{SM}_i$.
3. $\mathcal{SM}_i$ calculates $k_i = (S_i + a)T_j$, $M^{'} = H_1(0, k_i)$ and verifies $M \stackrel{?}{=} M'$. $\mathcal{SM}_i$ authenticates $\mathcal{HE}_j$ if the verification holds and sets $SK = H_1(ID_i||ID_j, k_i)$. $\mathcal{SM}_i$ computes $N = H_1(1, k_i)$ and transmits it to $\mathcal{HE}_j$.
4. After receiving $\{N\}$, $\mathcal{HE}_j$ computes $N^{'} = H_1(1, k_j)$ and checks $N \stackrel{?}{=} N'$. If it holds, then $\mathcal{HE}_j$ authenticates $\mathcal{SM}_i$ and computes the session key $SK = H_1(ID_i||ID_j, k_j)$.

5 | CRYPTANALYSIS OF MOHAMMADALI ET AL'S PROTOCOL

This section explains how both variants fail to ensure session key security and privacy of the TA, if $\mathcal{SM}_i$'s credentials are leaked (under the assumptions of CK-adversary model, explained in section 3.1). We posit the importance of key agreement protocols to ensure the security of user's session keys, even if the session state reveal attack exposes the ephemeral

10991131, 2019, 16, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/dac.4137 by Istanbul Gelisim Universitesi, Wiley Online Library on [03/01/2025]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

secrets to an adversary $\mathcal{A}$. We will also show how the master secret key of TA can be compromised, and how the secret key of $\mathcal{SM}_i$ can be revealed using an explicit insider attack. Thus, $\mathcal{A}$ can impersonate any legitimate $\mathcal{SM}_i$ to share or exchange a session key. Moreover, it is shown that both protocol variants do not offer user anonymity, and they are susceptible to replay and distributed of service (DoS) attacks.

5.1 | Compromise of TA's master secret key

Step 1: Since H_1 is defined as $H_1 : \{0,1\}^* \times G_q \rightarrow {Z_q}^*$, H_1 is an integer value.
Step 2: Since H_1, ID_j, and R_j are public, and y_j is the private key of AHES E_j, any malicious AHES E_j can compute $H_1(ID_j, R_j)$and $\frac{y_j}{H_1(ID_j,R_j)} = s$, where s is the master private key of the TA.

Hence, TA's master secret key can be compromised by the AHES. This can consequently result in a successful impersonation of TA.

5.2 | $\mathcal{SM}_i$'s secret key reveal

The secret key S_i can be easily revealed via an explicit insider attack, because of ephemeral credential or secret leakage attack. Since $\mathcal{SM}_i$ keeps its credential $\{R_i, y_i, r_i\}$ in its memory, $\mathcal{A}$ can attain the secret key S_i of $\mathcal{SM}_i$ as follows:

Step 1: If $\mathcal{A}$ successfully obtains r_i and y_i using secret leakage insider attack, then $\mathcal{A}$ can compute the corresponding secret key of $\mathcal{SM}_i$ as $S_i = r_i + y_i$.
Step 2: Hence, the private key S_i of $\mathcal{SM}_i$ is compromised.

Moreover, if an adversary has access to the memory of the smart meter $\mathcal{SM}_i$, then the adversary can also directly access the secret key S_i.

5.3 | $\mathcal{SM}_i$ impersonation attack

Since both protocol variants are insecure against ephemeral secret leakage attack, they also fail to achieve strong credential privacy of $\mathcal{SM}_i$ when ephemeral secret $a \in {Z_q}^*$ is exposed to $\mathcal{A}$. Furthermore, its secret key S_i can be compromised as explained in Section 5.2. Therefore, $\mathcal{SM}_i$ can be easily impersonated in the following manner:

Step 1: $\mathcal{HE}_j$ responds to the initiated message from $\mathcal{SM}_i$ and performs the relevant computation. Then, $\mathcal{HE}_j$ sends $\{ID_j, T_j, M\}$ to $\mathcal{SM}_i$.
Step 2: $\mathcal{A}$ intercepts the tuple $\{ID_j, T_j, M\}$ and computes $k_a = (S_i + a)T_j$ because $\mathcal{A}$ has obtained the secret key S_i of $\mathcal{SM}_i$ as shown in Section 5.2. $\mathcal{A}$ then computes $SK = H_1(ID_i||ID_j, k_a)$ and $N = H_1(1, k_a)$ and transmits it to $\mathcal{HE}_j$.
Step 3: After receiving $\{N\}$, $\mathcal{HE}_j$ computes $N^{'} = H_1(1, k_j)$ and checks $N \stackrel{?}{=} N'$. If it holds, then $\mathcal{HE}_j$ authenticates $\mathcal{A}$ on behalf of $\mathcal{SM}_i$ and computes the session key $SK = H_1(ID_i||ID_j, k_j)$.
Step 4: Hence, $\mathcal{A}$ has successfully impersonates $\mathcal{SM}_i$ and establishes a session key SK with $\mathcal{HE}_j$.

5.4 | Man-in-the-middle (MiTM) attacks

Since $\mathcal{A}$ can compute $k_a = (S_i + a)T_j$ as shown in Section 5.3, $\mathcal{A}$ can also compute $M^{'} = H_1(0, k_i)$ and successfully verify $M \stackrel{?}{=} M'$. In other words, the authenticity of the message can also be compromised. Hence, we can conclude that both protocol variants do not prevent MiTM attacks.

5.5 | Anonymity violation and untraceability

As analysis of the relevant session information can reveal the credentials (specially identity) of any user to the adversary $\mathcal{A}$, a key establishment protocol can only provide anonymity and untraceability if it fulfills two conditions, namely, (a) $\mathcal{A}$ fails to identify the absolute identity of a particular user; (b) $\mathcal{A}$ fails to guess, whether two unique sessions initiated at different times belong to the same user. It is clear from our earlier discussion that both protocol variants do not achieve anonymity and untraceability.

5.6 | Replay and DoS attacks

Since an adversary can obtain the ephemeral secret $a \in {Z_q}^*$ using a session state reveal attack,[25] $\mathcal{A}$ can easily compute $N = H_1(1, k_i)$. Thus, both protocol variants do not prevent replay attack and, thus, vulnerable to DoS attacks.

10991131, 2019, 16, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/dac.4137 by Istanbul Gelisim Universitesi, Wiley Online Library on [03/01/2025]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

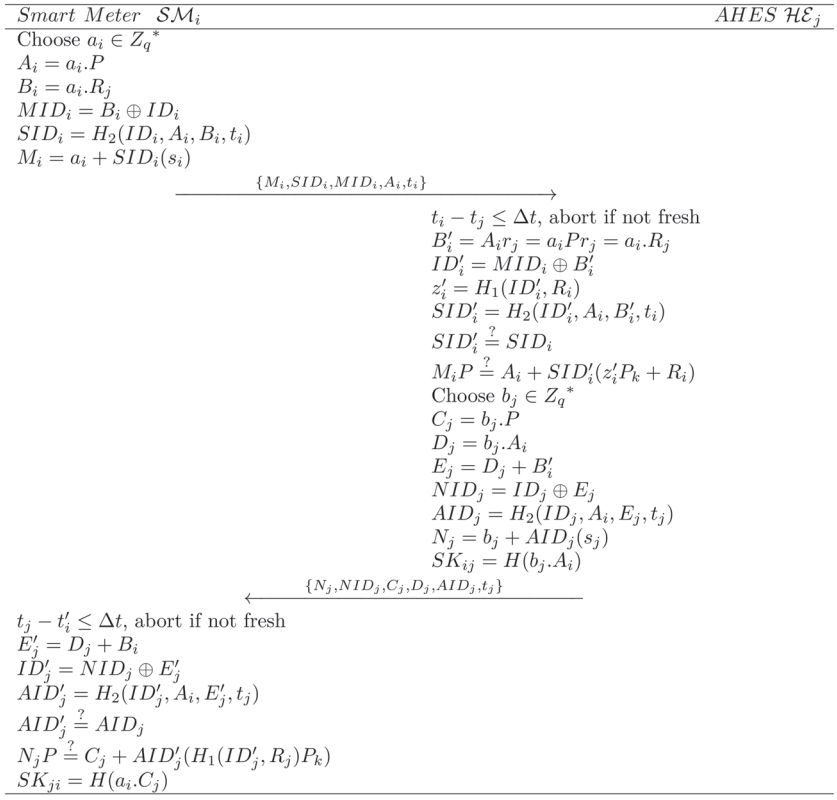


FIGURE 4 Proposed scheme

6 | IMPROVED SCHEME

This section presents the improved scheme, which is designed to achieve additional security by preventing popular security attacks. The improved scheme is designed in a way that master secret key of trusted agent and secret key of smart meter cannot be compromised easily as they are in the schemes of Mohammadali et al.[21] Moreover, instead of two separate variants, we have introduced a single more secure solution as validated and proved in Sections 7 and 8. The proposed scheme is presented in Figure 4. The proposed scheme comprises the following three phases: (*a*) trusted authority setup phase, (*b*) registration phase, and (*c*) key agreement phase. The trusted authority setup phase follows that of the original protocol.

6.1 | Trusted authority setup phase

The smart grid provider initializes and sets k as the intrinsic parameter of the system. Then, (*TA*) proceeds as follows:

- *TA* takes k-bit prime q and establishes $\{F_q, G_q, P, E/F_q, \}$.
- *TA* selects its master secret key $s \in Z_q^*$ and keeps it confidential and derives its corresponding public key $P_k = sP \in E/F_q$.
- Selects three hash functions $H : \{0,1\}^* \times G_q \rightarrow Z_q^*$, $H_1 : \{0,1\}^* \times G_q \rightarrow Z_q^*$, and $H_2 : \{0,1\}^* \times G_q \rightarrow Z_q^*$. It is assumed that the hash functions employed here belongs to the latest member of secure hash algorithms family $SHA-3$. The important property of any hash algorithm is that it is one-way means irreversible. In other words, it is computationally infeasible to determine m, if $H(m) = R$ specified. The resultant of hash function is known as hash digest, and its common purpose is to verify message integrity.
- At the end, *TA* publicizes the system parameters $\{F_q, G_q, P, P_k, E/F_q, H, H_1, H_2\}$.

6.2 | Registration phase

In this phase, the *AMI*-head-end system (AHES) and smart meter register with *TA* over a secure communication channel. The registration of the AHES and smart meter is enhanced by shifting the processing load (ie, point multiplication) towards *TA*. This allows us to reduce the computational burden at the AHES and meter.

10991131, 2019, 16, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/dac.4137 by Istanbul Gelisim Universitesi, Wiley Online Library on [03/01/2025]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

6.2.1 | Smart meter registration

- Each smart meter $\mathcal{SM}_i$ selects its identity ID_i and sends it to TA as part of the registration request.
- TA selects $r_i \in Z_q{}^*$ and computes $R_i = r_iP$, $z_i = H_1(ID_i, R_i)$ and $s_i = r_i + sz_i$.
- Both $\mathcal{SM}_i$ and TA have preshared some secret key. Therefore, TA can use this secret key to encrypt the ordered pair (s_i, R_i), which can be then sent to $\mathcal{SM}_i$.
- $\mathcal{SM}_i$ receives and decrypts the ordered pair and subsequently keeps the ordered pair in its tamper-proof module as his/her secret key.
  Then, $\mathcal{SM}_i$ publicizes R_i as its public key.

6.2.2 | AHES registration

- AHES as HE_j selects a random number $r_j \in Z_q{}^*$, computes $R_j = r_jP$, and sends both ID_j and R_j as a registration request over a secure communication channel.
- TA receives the registration request and computes $s_j = H_1(ID_j, R_j).s$.
- Both $\mathcal{HE}_j$ and TA have already exchanged a secret key. Therefore, TA can use this secret key to encrypt the ordered pair (s_j, R_j), which is later sent to $\mathcal{HE}_j$.
- $\mathcal{HE}_j$ receives and decrypts the ordered pair and subsequently keeps the ordered pair in its tamper-proof module as his/her secret key.
- If $(r_j + s_j)P \stackrel{?}{=} R_j + H_1(ID_j, Rj)P_k$ holds, then the secret key is considered valid.
- Then, $\mathcal{HE}_j$ publicizes R_j as its public key.

6.3 | Key agreement phase for proposed scheme

The key agreement phase uses two messages to establish a session key between $\mathcal{SM}_i$ and $\mathcal{HE}_j$, as follows:

1. $\mathcal{SM}_i$ chooses an arbitrary number $a \in Z_q{}^*$ and computes $A_i = a_i.P$, $B_i = a_i.R_j$, $MID_i = B_i \oplus ID_i$, $SID_i = H_2(ID_i, A_i, B_i, t_i)$ and $M_i = a_i + SID_i(s_i)$. $\mathcal{SM}_i$ sends a tuple $\{M_i, SID_i, MID_i, A_i, t_i\}$ to $\mathcal{HE}_j$.
2. In response to the initial message from $\mathcal{SM}_i$, $\mathcal{HE}_j$ verifies the freshness of time stamp as $t_i - t_j \leq \Delta t$. Thereafter, $\mathcal{HE}_j$ computes $B_i' = Ar_j = a_iPr_j = a_i.R_j$ and corresponding identity of the smart meter as $ID_i' = MID_i \oplus B_i'$. $\mathcal{HE}_j$ uses B_i' and ID_i' to compute $SID_i' = H_2(ID_i', A_i, B_i', t_i)$. Then, both the received SID_i and derived SID_i' are verified. If this equality holds, then the authentication procedure proceeds; otherwise, it terminates instantly. Moreover, $\mathcal{HE}_j$ computes $z_i' = H_1(ID_i', R_i)$ and verifies $M_iP \stackrel{?}{=} A_i + SID_i'(z_i'P_k + R_i)$. If it does not hold, then the session is aborted by $\mathcal{HE}_j$. After that, $\mathcal{HE}_j$ chooses an arbitrary number $b_j \in Z_q{}^*$, $C_j = b_j.P$, $D_j = b_j.A_i$, $E_j = D_j + B_i'$, $NID_j = ID_j \oplus E_j$, $AID_j = H_2(ID_j, A_i, E_j, t_j)$, $N_j = b_j + AID_j(s_j)$ and session key $SK_{ij} = H(b_j.A_i)$. $\mathcal{HE}_j$ replies to $\mathcal{SM}_i$ by sending a tuple $\{N_j, NID_j, C_j, D_j, AID_j, t_j\}$.
3. $\mathcal{SM}_i$ receives $\{N_j, NID_j, C_j, D_j, AID_j, t_j\}$ and verifies the freshness of time stamp as $t_j - t_i' \leq \Delta t$. Thereafter, $\mathcal{SM}_i$ computes $E_j' = D_j + B_i$ and corresponding identity of $\mathcal{HE}_j$ as $ID_j' = NID_j \oplus E_j'$. $\mathcal{SM}_i$ uses E_j' and ID_j' to compute $AID_j' = H_2(ID_j', A_i, E_j', t_j)$. Then, the equality of the received AID_j and derived AID_j' is verified. If this equality holds, then the authentication procedure proceeds; otherwise, it terminates instantly. Moreover, $\mathcal{SM}_i$ verifies $N_jP \stackrel{?}{=} C_j + AID_j'(H_1(ID_j', R_j)P_k)$. If it does not hold, then the session is aborted by $\mathcal{SM}_i$. In the end, $\mathcal{SM}_i$ computes $SK_{ji} = H(a_i.C_j)$.

7 | SECURITY PROOF

We will now demonstrate the security of the proposed protocol in the random oracle model.

A. Security Model

There are three participants in the protocol $\mathcal{P}$, namely, Smart Meter, AMI Head End System, and Trusted Authority (denoted as SM, HE, and TA, respectively). TA's role is limited to the time at registration. However, communications between SM and HE can be frequent.

Each instance of SM and HE is denoted as SM^i and HE^j, respectively. In other words, the subscripts i and j refer to the i^{th} and j^{th} instances of participants SM and HE, respectively. Each oracle returns one of three possible results,

10991131, 2019, 16, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/dac.4137 by Istanbul Gelisim Universitesi, Wiley Online Library on [03/01/2025]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

namely, accept, reject, or $\perp$, based on the adversary's interactions via one of the following queries is capable of performing the following:

1) **Extract**(ID_i): This query allows the adversary *Atk* to gain access to the identity information of the target user.
2) **Send**($msg, \mathcal{P}^k$): This query allows *Atk* to send message *msg*.
3) **H**($l/l1/l2/l3, rec$): This allows *Atk* to obtain the random hashed value r for m, where r is maintained as a record (rec, r) in the respective list (ie, Hl, Hl_1, or Hl_1, which are initially empty).
4) **Execute**($\mathcal{P}^m, \mathcal{P}^n$): This is a faithful execution of the protocol interactions between $\mathcal{P}^m$ and $\mathcal{P}^n$.
5) **Reveal**($\mathcal{P}^k$): This query allows *Atk* to obtain the session key of the target user.
6) **Corrupt**($\mathcal{P}^k$): This query allows *Atk* to obtain the secret key of the target user.

The advantage of *Atk* in breaking the security of the Advanced Metering Infrastructure Protocol (AMIP) is quantified as

$$Adt^{AMIP}(Atk) = |Pr[\omega = \omega'] - 0.5|. \tag{1}$$

Theorem 1. *The security of our protocol assumes the intractability of the Elliptic Curve Discrete Logarithm Problem (ECDLP), and H, H_1, and H_2 are defined as random oracles. Alternatively, if an attacker Atk is able to successfully break the security of the protocol, then it can be assumed that there exists an algorithm $\mathcal{L}$ that can solve ECDLP in polynomial time t with minimal probability ϵ'. Here, multiple H_1 queries are confined as $_{h1}$, and corrupt ones are confined by $_c$.*

$$\epsilon' \geq (1/_{h1})(1 - {}_c/_{h1})\,\epsilon \tag{2}$$

Proof. **Game 0:** If *Atk* can break the bilateral authentication between SM^i and HE^j, then ECDLP can be solved using a polynomial time algorithm $\mathcal{L}$. This algorithm manages the following four lists L_h^l, $L_{h_1}^{l1}$, $L_{h_2}^{l2}$ and $L_{h_{Aux}}^{l3}$:

L_h^l List: It maintains the tuples to hash function H as (in_{z1}, out_{z1}), where input and output tuples are denoted as in_{z1} and out_{z1}, respectively.

$L_{h_1}^{l1}$ List: It maintains the tuples to hash function H_1 as (in_{z2}, out_{z2}), where input and output tuples are denoted as in_{z2} and out_{z2}, respectively.

$L_{h_2}^{l2}$ List: It maintains the tuples to hash function H_2 as (in_{z3}, out_{z3}), where input and output tuples are denoted as in_{z3} and out_{z3}, respectively.

$L_{h_{Aux}}^{l3}$ List: It maintains the tuples as (s_i, R_i, P_{ki}).

- Suppose *Atk* initiates H query with an input (b_jA_i), then $\mathcal{L}$ acknowledges with tuple (b_jA_i, SK_{ij}), If L_h^l contains it. Otherwise, $\mathcal{L}$ picks up an arbitrary value SK_{ij}, produces an outcome, and stores the fresh tuples (b_jA_i, SK_{ij}) in L_h^l.
- Suppose *Atk* initiates H_1 query with an input (ID_i, R_i), then $\mathcal{L}$ acknowledges with tuple (ID_i, R_i, z_i), If $L_{h_1}^{l1}$ contains it. Otherwise, $\mathcal{L}$ picks up an arbitrary value z_i, produces an outcome, and stores the fresh tuples ((ID_i, R_i, z_i) in $L_{h_1}^{l1}$.
- Suppose *Atk* initiates H_2 query with an input (ID_i, B_i, t_i), then $\mathcal{L}$ acknowledges with tuple (ID_i, B_i, t_i, A_i), If $L_{h_2}^{l2}$ contains it. Otherwise, $\mathcal{L}$ picks up an arbitrary value A_i, produces an outcome, and updates the fresh tuples ((ID_i, B_i, t_i, A_i) in $L_{h_2}^{l2}$.

1) **Extract**(ID_k): On receipt of *Extract*(ID_k), $\mathcal{L}$ chooses a pair of numbers z_k, $r_k \in Z_q{}^*$, and sets $s_k = r_k$, $H_1(ID_k, R_k) = z_k$ and $R_k = s_kP - z_kP_k$. It is to be noted that $s_k = r_k$ satisfies the respective equation $R_k + H_1(ID_k, R_k)P_k = s_kP$. $\mathcal{L}$ proceeds as given below:
   If $ID_k \in \{ID_i, ID_j\}$, then $\mathcal{L}$ sends $S_k = \perp$ to *Atk* and updates (ID_k, R_k, s_k) and ($ID_k, R_k, \perp$) in $L_{h_1}^{l1}$ and $L_{h_{Aux}}^{l3}$ in the respective order.
   Else, $\mathcal{L}$ sends $s_k = r_k$ to *Atk* and updates the tuples (ID_k, R_k, z_k) and (ID_k, R_k, s_k) in $L_{h_1}^{l1}$ and $L_{h_{Aux}}^{l3}$ in the respective order.

2) **Corrupt**(ID_k): By receiving *Corrupt*(ID_k) query, $\mathcal{L}$ performs the following:
   If $ID_k \in \{ID_i, ID_j\}$, then fetches (ID_k, R_k, z_k) and ($ID_k, R_k, \perp$) from $L_{h_1}^{l1}$ and $L_{h_{Aux}}^{l3}$ and sends them as a reply to *Atk*.
   Else, fetches (ID_k, R_k, z_k) and (ID_k, R_k, s_k) from $L_{h_1}^{l1}$ and $L_{h_{Aux}}^{l3}$ and sends them as a reply to *Atk*.

10991131, 2019, 16, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/dac.4137 by Istanbul Gelisim Universitesi, Wiley Online Library on [03/01/2025]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

3) **Send query**: When *Atk* initiates a $Send(ID_k)$ for MID_k, $\mathcal{L}$ fetches the lists maintained as $L_{h_1}^{l1}$ and $L_{h_{Aux}}^{l3}$ and participates as follows:

If $ID_k = ID_i$, then $\mathcal{L}$ proceeds as given below:

a Selects $MID_k \in Z_q^*$ and computes $B_i = MID_i \oplus ID_i$, $SID_k = H_2(ID_k, A_k, B_k, t_k)$ and $M_k = a_k + SID_k(s_k)$.
b Outputs $(M_k, SID_k, MID_k, A_k, t_k)$.

Else, $\mathcal{L}$ proceeds as given below:

- Selects an arbitrary number $a_k \in Z_q^*$ and computes $A_k = a_k P$.
- Computes $B_k = a_k R_k, MID_k = B_k \oplus ID_k, SID_k = H_2(ID_k, A_k, B_k, t_k)$ and $M_k = a_k + SID_k(s_k)$.
- Outputs $(M_k, SID_k, MID_k, A_k, t_k)$.

Output: By using the forking lemma,[26] *Atk* can produce distinct authentication solicits messages $\{M_i, SID_i, MID_i, A_i, t_i\}$ and $\{Mi', SID_i, MID_i, A_i, t_i\}$, with distinct hash values $SIDi' \neq SID_i$. For legitimate request message, we can pursue as follows:

$$M_i P = A_i + SID_i(z_i P_k + R_i), \tag{3}$$

$$M_i' P = A_i + SID_i'(z_i P_k + R_i). \tag{4}$$

From Equations (3) and 4, we can obtain

$$(M_i' - M_i)P = (SID_i' - SID_i)(z_i P_k + R_i). \tag{5}$$

Since $P_k = sP$ and $R_i = r_i P$, therefore Equation 5 can be manipulated as follows:

$$sP = \left[\frac{(M_i' - M_i)}{z_i(SID_i' - SID_i)} - \frac{r_i}{z_i}\right] P. \tag{6}$$

Thus, we obtain

$$s = \left[\frac{(M_i' - M_i)}{z_i(SID_i' - SID_i)} - \frac{r_i}{z_i}\right]. \tag{7}$$

Finally, $\mathcal{L}$ can break ECDLP as $s = \left[\frac{(M_i' - M_i)}{z_i(SID_i' - SID_i)} - \frac{r_i}{z_i}\right]$.

Success Probability:

- EV_1: The *Atk* successfully produces a forgery for ID_k, with the exception of $ID_k = ID_l$.
- EV_2: The *Atk* initiates a corrupt query against ID_l.

We keep $Pr[EV_1] = 1 - 1/_{h1}$ and $Pr[EV_2] = {}_c(1/_{h1})$. As a result, the probability that $\mathcal{L}$ will not terminate is $Pr[\neg EV_1 \wedge \neg EV_2] = (1 - 1/_{h1})(1 - {}_c/_{h1})$. Thus, $\mathcal{L}$ can solve ECDLP with a probability of $\epsilon' \geq (1/_{h1})(1 - {}_c/_{h1})\epsilon$, where ϵ refers to the *Atk*'s winning probability. □

Theorem 2. *Our protocol provides bilateral authentication between SM^i and HE^j under the assumption that ECCDHP is computationally hard to solve.*

Proof. **Game 1:** If *Atk* can break the bilateral authentication between SM^i and HE^j, then ECCDHP can be solved by developing a polynomial time algorithm $\mathcal{L}$. This algorithm manages the following four lists L_h^l, $L_{h_1}^{l1}$, $L_{h_2}^{l2}$, and $L_{h_{Aux}}^{l3}$:

L_h^l List: It maintains the tuples to hash function H as (in_{z1}, out_{z1}), where input and output tuples are denoted as in_{z1} and out_{z1}, respectively.

$L_{h_1}^{l1}$ List: It maintains the tuples to hash function H_1 as (in_{z2}, out_{z2}), where input and output tuples are denoted as in_{z2} and out_{z2}, respectively.

10991131, 2019, 16, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/dac.4137 by Istanbul Gelisim Universitesi, Wiley Online Library on [03/01/2025]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

$L_{h_2}^{l2}$ List: It maintains the tuples to hash function H_2 as (in_{z3}, out_{z3}), where input and output tuples are denoted as in_{z3} and out_{z3}, respectively.

$L_{h_{Aux}}^{l3}$ List: It maintains the tuples as (s_i, R_i, P_{ki}).

- Suppose *Atk* initiates H query with an input (b_jA_i), then $\mathcal{L}$ acknowledges with tuple (b_jA_i, SK_{ij}), If L_h^l contains it. Otherwise, $\mathcal{L}$ picks up an arbitrary value SK_{ij}, produces an outcome, and stores the fresh tuples (b_jA_i, SK_{ij}) in L_h^l.

(a) **Extract**(ID_k): On receipt of query $Extract(ID_k)$, $\mathcal{L}$ performs as follows:

- If $(ID_k = ID_i)$, then $\mathcal{L}$ stores the tuple (ID_i, R_i, z_i) into $L_{h_1}^{l1}$ and computes $R_i = uP - z_iP_k$. As a result, $R_i + z_iP_k = uP - z_iP_k + z_iP_k = uP$. Thereafter, $\mathcal{L}$ sets $s_i = \perp$ as the secret key of ID_i and produces $s_i = \perp$ in response to *Atk*. Then, tuples $(ID_i, R_i, \perp)$ and (ID_i, R_i, z_i) are updated in $L_{h_{Aux}}^{l3}$ and $L_{h_1}^{l1}$ in the respective order.
- If $(ID_k = ID_j)$, then $\mathcal{L}$ stores the tuple (ID_j, R_j, z_j) into $L_{h_1}^{l1}$ and computes $R_j = vP - z_jP_k$. As a result, $R_j + z_jP_k = vP - z_jP_k + z_jP_k = vP$. Thereafter, $\mathcal{L}$ sets $s_j = \perp$ as the secret key of ID_j and produces $s_j = \perp$ in response to *Atk*. Then, tuples $(ID_j, R_j, \perp)$ and (ID_j, R_j, z_j) are updated in $L_{h_{Aux}}^{l3}$ and $L_{h_1}^{l1}$ in the respective order.
- Else, $\mathcal{L}$ chooses a pair of numbers $z_k, r_k \in Z_q^*$. It sets $s_k = r_k$, $H_1(ID_k, R_k) = z_k$, and $R_k = s_kP - z_kP_k$. It is to be noted that $s_k = r_k$ satisfies the respective equation $R_k + H_1(ID_k, R_k)P_k = s_kP$. $\mathcal{L}$ sends s_k to *Atk* and updates the lists $L_{h_{Aux}}^{l3}$ and $L_{h_1}^{l1}$ by adding tuples $(ID_k, R_k, \perp)$ and (ID_k, R_k, z_k), respectively.

(b) **Corrupt**(ID_k): Upon receiving $Corrupt(ID_k)$ query from *Atk*, $\mathcal{L}$ responds as follows:

- $\mathcal{L}$ sends $\perp$, if $ID_k \in \{ID_i, ID_j\}$ is satisfied.
- Else, $\mathcal{L}$ locates the respective list $L_{h_{Aux}}^{l3}$ and sends s_k if there exists a tuple (ID_k, R_k, s_k). Alternatively, carries out $H(ID_k)$ and $Extract(ID_k)$ queries for a pair of tuples (ID_k, R_k, z_k) and (ID_k, R_k, s_k). This results in s_k being used as the secret key. $\mathcal{L}$ updates the lists $L_{h_{Aux}}^{l3}$ and $L_{h_1}^{l1}$ by adding tuples (ID_k, R_k, z_k) and (ID_k, R_k, s_k), respectively.

(c) **Send query**: Whenever *Atk* initiates a *Send* query, $\mathcal{L}$ fetches $\{b_jA_i\}$ as tuple from list L_h^l and sends SK_{ij} to *Atk*.

(d) **Reveal query**: Whenever *Atk* initiates a *Reveal* query on ID_k, $\mathcal{L}$ drops the execution of protocol if $ID_k \in \{ID_i, ID_j\}$. $\mathcal{L}$ returns $r_k \in Z_q^*$ in the case of nonsuccessful matching session. Alternatively, $\mathcal{L}$ returns the current session key.

(e) **Test query**: If $\lfloor = 1$, $\mathcal{L}$ sends the real key to *Atk*. Alternatively, $\mathcal{L}$ generates and sends some arbitrary value.

Output: If *Atk* can choose a reply message $NID_j, C_j, D_j, AID_j, t_j$ then it can be considered that *Atk* can successfully imitate or forge HE_j.

- The probability that *Atk* can guess $SK_{ij} = H(b_jA_i)$ without fabricating H and in the absence of any information about b_j is $\frac{1}{2^x}$.
- Since SM^i has $R_i = uP - z_iP_k$ and HE^j has $R_j = vP - z_jP_k$, we have $uvP = SK_{ij} - a_i(uP) - b_j(vP) - a_ib_j$. Thus, *Atk* can accurately guess SK_{ij} in the absence of any information about b_j, if *Atk* can successfully solve the ECCDHP.

Success Probability: The probability of breaking ECCDHP by *Atk* is $\epsilon' \geq \left(\epsilon - \frac{1}{2^x}\right)$ □

8 | SECURITY AND PERFORMANCE EVALUATION

This section initially highlights the security peculiarities of our protocols with the correlated protocols [6,12,13,21,24] in the literature. Then, performance is expressed in terms of computation and communication cost. The performance of our protocol is also consolidated through comparison with correlated protocols.

10991131, 2019, 16, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/dac.4137 by Istanbul Gelisim Universitesi, Wiley Online Library on [03/01/2025]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

TABLE 2 Security features: Comparative summary

Scheme→ Security Features↓	Proposed	Mohammadali et al[21]	Xia and Wang[24]	Liu et al[6]	Nicanfar et al[12]	Wan et al[13]
Secret Key Reveal Attack Resilience	✓	×	×	✓	✓	✓
Replay and DOS Attack Resilience	✓	×	×	✓	✓	✓
Man-in-the-middle Attack Resilience	✓	×	×	✓	✓	✓
Impersonation Attack Resilience	✓	×	×	✓	✓	✓
Promises User Anonymity	✓	×	×	×	×	×
Promises User Un-traceability	✓	×	×	×	×	×

10991131, 2019, 16, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/dac.4137 by Istanbul Gelisim Universitesi, Wiley Online Library on [03/01/2025]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

TABLE 3 System specification for experiments

Item	Details
Motherboard	Intel
Processor	Pentium Dual CPU E2200
Processor Clock Speed	2.20 GHz
Primary Memory (RAM)	2048 MB
Operating System	Ubuntu 12.04.1
Word Size	32 bit operating system

8.1 | Security features

Table 2 summarizes the security features of our protocol and those presented in the previous studies.[6,12,13,21,24] It can be observed that our protocol has aided security features, like it promises user anonymity and untraceability. As discussed earlier, key establishment protocol can only provide anonymity and untraceability if it fulfills two conditions, namely, (a) $\mathcal{A}$ fails to identify the absolute identity of a particular user. (b) $\mathcal{A}$ fails to guess, whether two unique sessions initiated at different times belong to the same user. Our protocol fulfills both conditions as it does not exposes its identity during public message exchanges between participants. Moreover, it is hard for an adversary to identify similarity between two different session request from same user because we have utilized the session-specific credentials to prevent the traceability. Whereas, the correlated protocol can be observed with various security weaknesses against different attacks.

8.2 | Performance

The performance of our protocol is perused from two aspects, ie, computation and communication cost, which is discussed as below. Before presenting computation and communication cost comparison, the experimental setup, assumptions, and calculations necessary for performance comparison are presented first.

8.2.1 | Experimental setup and parametric assumptions

The following notations are assumed to furnish the computation cost of each protocol.

- E_{bp}: Execution time of pairing operation
- E_e: Execution time of modular exponentiation operation
- E_{pm}: Execution time of point multiplication operation
- E_{sed}: Execution time of symmetric encryption/decryption
- E_{ped}: Execution time of public key encryption/decryption
- E_h: Execution time of one-way hash function
- E_r: Execution time to perform pseudo-random function
- E_{sig}: Execution time of digital signature
- E_{ver}: Execution time of signature verification

The per unit execution cost (in terms of time) of the above-listed fundamental cryptographic operations are assumed as determined by Kilinc and Yanik in Kilinc and Yanik.[27] Kilinc and Yanik observed these values over a personal computer with defined specifications. These values are observed by implementing the concerned primitive cryptographic operations over a dedicated system. The specification of dedicated system is presented in Table 3. The proposed protocol and related protocols are implemented using predefined Pairing-based Cryptography Library (PBC) (version 0.5.12) that is developed over GMP Library (version 5.0.5).[28] According to their observations, E_{bp}: takes 5.8110ms, E_e: takes 3.8500ms, E_{pm}: takes 2.2260ms, E_{sed}: takes 0.0046ms, E_{ped}: takes 3.8500ms, E_h: takes 0.0023ms, E_r: takes 0.5390ms, E_{sig}: takes 3.8500ms and E_{ver}: takes 0.1925ms. However, some insignificant operations such as *XOR* are ignored.

8.2.2 | Computation cost

The computation cost is observed by identifying the primitive cryptographic operations involved in the concerned protocol as discussed in Section 8.2.1. Thereafter, associated cost (in terms of time taken for execution of each operation) of these primitive operations is accumulated to determine the total computation cost of the particular protocol.

Since our protocol facilitates smart meter and AHES to establish key agreement. The computations on smart meter side are carried out in 2.7719ms during key agreement. Whereas, AHES side computations are executed in 4.9979ms. The computation cost comparison of our protocol with correlated protocols is presented in Table 4. The comparison shows

10991131, 2019, 16, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/dac.4137 by Istanbul Gelisim Universitesi, Wiley Online Library on [03/01/2025]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

TABLE 4 Performance analysis

Protocol	Computation Cost			Communication Cost
	Smart Meter	AHES	TA or Relaying Meter	
Our	$E_r + E_{pm} + 3E_h = 2.7719$ ms	$E_r + 2E_{pm} + 3E_h = 4.9979$ ms	N/A	896 bits
[21]	$E_r + 2E_{pm} + 3E_h = 4.9979$ ms	$E_r + 3E_{pm} + 4E_h = 7.2262$ ms	N/A	1024 bits
[24]	$E_r + E_e + E_{sed} + 4E_h = 4.4028$ ms	$E_r + E_e + 3E_h = 4.3959$ ms	$2E_e + E_{sed} + 3E_h = 7.7115$ ms	640 bits
[12]	$E_r + 3E_e + E_{sed} + E_{ped} + 8E_h + E_{sig} + E_{ver} = 20.0045$ ms	$E_r + 3E_e + E_{sed} + 7E_h + E_{sig} + E_{ver} = 16.1522$ ms	N/A	1152 bits
[13]	$E_r + E_{pm} + E_{bp} = 8.576$ ms	$E_r + E_{pm} + E_{bp} = 8.576$ ms	N/A	512 bits

10991131, 2019, 16, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/dac.4137 by Istanbul Gelisim Universitesi, Wiley Online Library on [03/01/2025]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

that our protocol is cost-effective as compared with correlated protocols. Thus, our protocol is proved to be more suitable candidate for resource constrained environment of smart grid.

8.2.3 | Communication cost

The communication cost is observed by determining the number of bits transmitted between the participants during protocol execution. Table 4 presents the communication cost comparison. It shows that our protocol has lower communication overhead as compared with protocols presented in Mohammadali et al[21] and Nicanfar et al.[12] However, communication overhead of protocols presented in Xia and Wang[24] and Wan et al[13] is lower than our protocol. It is due to the fact that our protocol has aided security features (see Table 2). Moreover, it is important to reveal that we have determined communication overhead of each protocol under following assumptions: 128 bits are assumed for the participant's identity, public/private keys, and shared private keys, and for hash values and time stamps, we assume them to be 64 bits.

9 | CONCLUSION

Designing secure cryptographic protocols remains challenging, as evidenced by the number of protocols whose security was invalidated subsequent to their publication. However, such break-and-fix cycle is essential as it allows the cryptographic protocol designer community to learn from past experience. Also, in this paper, we revealed the security weaknesses in the identity-based key establishment protocol of Mohammadali et al.[21] We then demonstrated how the security weaknesses can be fixed. We then proved the security of the fixed protocol in the random oracle model, as well as evaluating its security features (with two additional features: anonymity and untraceability) and performance.

ORCID

Khalid Mahmood https://orcid.org/0000-0001-5046-7766
Shehzad Ashraf Chaudhry https://orcid.org/0000-0002-9321-6956
Saru Kumari https://orcid.org/0000-0003-4929-5383

REFERENCES

1. Jin J, Gubbi J, Marusic S, Palaniswami M. An information framework for creating a smart city through internet of things. *IEEE Internet Things J*. 2014;1(2):112-121.
2. McHenry MP. Technical and governance considerations for advanced metering infrastructure/smart meters: Technology, security, uncertainty, costs, benefits, and risks. *Energy Policy*. 2013;59:834-842.
3. Yang Z, Ping S, Sun H, Aghvami AH. Crb-rpl: A receiver-based routing protocol for communications in cognitive radio enabled smart grid. *IEEE Trans Veh Technol*. 2017;66(7):5985-5994.
4. Aalamifar F, Lampe L. Cost-efficient qos-aware data acquisition point placement for advanced metering infrastructure. *IEEE Trans Commun*. 2018;66(12):6260-6274.
5. Gungor VehbiC, Sahin Dilan, Kocak Taskin, et al. Smart grid technologies: Communication technologies and standards. *IEEE Trans Ind Inf*. 2011;7(4):529-539.
6. Liu N, Chen J, Zhu L, Zhang J, He Y. A key management scheme for secure communications of advanced metering infrastructure in smart grid. *IEEE Trans Ind Electron*. 2013;60(10):4746-4756.
7. Kamto J, Qian L, Fuller J, Attia J. Light-weight key distribution and management for advanced metering infrastructure. In: IEEE GLOBECOM Workshops (GC Wkshps) IEEE; 2011:1216-1220.
8. Chen J, Liu N, Liu W, Luo H. A key management scheme for secure communications of advanced metering infrastructure. In: International Conference on Applied Informatics and Communication. Springer; 2011:430-438.
9. Wu D, Zhou C. Fault-tolerant and scalable key management for smart grid. *IEEE Trans Smart Grid*. 2011;2(2):375-381.
10. Nicanfar H, Jokar P, Leung VC. Smart grid authentication and key management for unicast and multicast communications. In: IEEE PES Innovative Smart Grid Technologies Asia (ISGT) IEEE; 2011:1-8.
11. Nicanfar H, Leung VC. Multilayer consensus ECC-based password authenticated key-exchange (MCEPAK) protocol for smart grid system. *IEEE Trans Smart Grid*. 2013;4(1):253-264.
12. Nicanfar H, Jokar P, Beznosov K, Leung VC. Efficient authentication and key management mechanisms for smart grid communications. *IEEE Syst J*. 2014;8(2):629-640.
13. Wan Z, Wang G, Yang Y, Shi S. Skm: Scalable key management for advanced metering infrastructure in smart grids. *IEEE Trans Ind Electron*. 2014;61(12):7055-7066.
14. Benmalek M, Challal Y, Bouabdallah A. Efficient and scalable multi-group key management for advanced metering infrastructure in smart grid. In: IEEE CIT; 2015.

10991131, 2019, 16, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/dac.4137 by Istanbul Gelisim Universitesi, Wiley Online Library on [03/01/2025]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

15. Mahmood K, Chaudhry SA, Naqvi H, Shon T, Ahmad HF. A lightweight message authentication scheme for smart grid communications in power sector. *Comput Electr Eng*. 2016;52:114-124.
16. Mahmood K, Chaudhry SA, Naqvi H, et al. An elliptic curve cryptography based lightweight authentication scheme for smart grid communication. *Futur Gener Comput Syst*. 2018;81:557-565.
17. Chen CM, Xiang B, Liu Y, Wang KH. A secure authentication protocol for internet of vehicles. *IEEE Access*. 2019;7:12047-12057.
18. Kumari S, Chaudhary P, Chen CM, Khan MK. Questioning key compromise attack on Ostad-Sharif et al.'s authentication and session key generation scheme for healthcare applications. *IEEE Access*. 2019.
19. Kumari S, Li X, Wu F, et al. Design of a provably secure biometrics-based multi-cloud-server authentication scheme. *Futur Gener Comput Syst*. 2017;68:320-330.
20. Kumari S, Khan MK, Atiquzzaman M. User authentication schemes for wireless sensor networks: a review. *Ad Hoc Networks*. 2015;27:159-194.
21. Mohammadali A, Haghighi MS, Tadayon MH, Nodooshan AM. A novel identity-based key establishment method for advanced metering infrastructure in smart grid. *IEEE Trans Smart Grid*. 2017:1-1. https://doi.org/10.1109/TSG.2016.2620939
22. Brusilovsky A, Faynberg I, Patel S, Zeltsan Z. Password-authenticated key exchange (pak) protocol. *STUDY GROUP*. 2006;17.
23. Wu TD, et al. The secure remote password protocol. *NDSS*. 1998;98:97-111.
24. Xia J, Wang Y. Secure key distribution for the smart grid. *IEEE Trans Smart Grid*. 2012;3(3):1437-1443.
25. Canetti R, Krawczyk H. Analysis of key-exchange protocols and their use for building secure channels. In: International Conference on the Theory and Applications of Cryptographic Techniques. Springer; 2001:453-474.
26. Pointcheval D, Stern J. Security arguments for digital signatures and blind signatures. *J Cryptol*. 2000;13(3):361-396.
27. Kilinc HH, Yanik T. A survey of sip authentication and key agreement schemes. *IEEE Commun Surv Tutorials*. 2014;16(2):1005-1023.
28. Lynn B, Mitchell J, Boyen X. Efficient implementation of pairing-based cryptosystems. *J Cryptol*. 2004;17:2004.

How to cite this article: Mahmood K, Arshad J, Chaudhry SA, Kumari S. An enhanced anonymous identity-based key agreement protocol for smart grid advanced metering infrastructure. *Int J Commun Syst*. 2019;32:e4137. https://doi.org/10.1002/dac.4137

10991131, 2019, 16, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/dac.4137 by Istanbul Gelisim Universitesi, Wiley Online Library on [03/01/2025]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License