A Provably Secure Anonymous Authentication Protocol for Consumer and Service Provider Information Transmissions in Smart Grids

Zahraa Abdullah Ali 1, Zaid Ameen Abduljabbar 1,*, Hamid Ali Abed AL-Asadi 1, Vincent Omollo Nyangaresi 2,3, Iman Qays Abduljaleel 1 and Abdulla J. Y. Aldarwish 1,4

1 Department of Computer Science, College of Education for Pure Sciences, University of Basrah, Basrah 61004, Iraq; iman.abduljaleel@uobasrah.edu.iq (I.Q.A.)
2 Department of Computer Science and Software Engineering, Jaramogi Oginga Odinga University of Science and Technology, Bando 40601, Kenya
3 Department of Applied Electronics, Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences, Chennai 600124, India
4 Department of Computer Science, College of Computer Science and Information Technology, University of Basrah, Basrah 61004, Iraq
* Correspondence: zaid.ameen@uobasrah.edu.iq

Abstract: Smart grids integrate information technology, decision support systems, communication networks, and sensing technologies. All these components cooperate to facilitate dynamic power adjustments based on received client consumption reports. Although this brings forth energy efficiency, the transmission of sensitive data over the public internet exposes these networks to numerous attacks. To this end, numerous security solutions have been presented recently. Most of these techniques deploy conventional cryptographic systems such as public key infrastructure, blockchains, and physically unclonable functions that have either performance or security issues. In this paper, a fairly efficient authentication scheme is developed and analyzed. Its formal security analysis is carried out using the Burrows–Abadi–Needham (BAN) logic, which shows that the session key negotiated is provably secure. We also execute a semantic security analysis of this protocol to demonstrate that it can resist typical smart grid attacks such as privileged insider, guessing, eavesdropping, and ephemeral secret leakages. Moreover, it has the lowest amount of computation costs and relatively lower communication overheads as well as storage costs.

Keywords: attacks; authentication; BAN; protocol; security; smart grids; privacy

1. Introduction

Smart grid (SG) networks incorporate information technology and energy grid so as to manage energy consumptions efficiently. This is normally accomplished by offering bi-directional communication for data exchanges between consumers and power producers [1]. In addition, an SG integrates intelligent sensing, contemporary communication networks, and novel systems that support decision making in conventional grid systems. These technologies enable the effectual distribution of power from the generating stations to the consumer terminals. As explained in [2], SG bi-directional communication is achieved through Advanced Metering Infrastructure (AMI). A typical AMI comprises concentrators, smart meters, and measurement data management systems. On the other hand, a typical SG is made up of control, sensing, and communication systems and actuators [3]. Whereas smart meters (SMs) perform sensing and communication, actuation and control are executed by service providers (SPs). Therefore, SMs are located at consumer premises, where they accurately measure power consumption and transmit these data over to the SP servers. Through effective real-time processing and analyses of consumer data, the generation and
distribution of power is dynamically fine-tuned in accordance with user demands. This helps in enhancing the reliability of the power grid system [4].

In spite of the benefits discussed above, the public internet is utilized for the data exchange between the SMs and the SPs [5]. As such, the SG is exposed to security and privacy threats such as eavesdropping, forgery, denial of service (DoS), tampering, and ephemeral secret leakage (EPSL) [6,7]. In addition, the misuse of consumer power consumption reports can lead to privacy leaks. By sending forged and inaccurate data, the SG network can incur additional loads [8]. All these challenges can disrupt the communication process, leading to the degradation of the SG system’s performance [9]. As such, security violations and privacy leakages are major issues during smart grid design [10]. This can be attained by perfect data encryption, mutual authentication, as well as session key establishment. In addition, Authenticated Key Exchange (AKE) is crucial for the protection of transmitted data against tampering and interception [6].

The above concerns necessitate the designing of robust, privacy-preserving, secure, and lightweight protocols to safeguard the data exchanged among legitimate SG participants. Since an SG comprises numerous SMs, each SM must be authenticated prior to information exchange. This will help curb threats exemplified by impersonation, SM capture, Man-in-the-Middle (MitM), packet replays, de-synchronization, and privileged insider [7]. Upon an effectual mutual authentication process, a common session key should be created between the SM and the SPs to encipher the exchanged data. In addition, data integrity should be upheld, while preventing non-repudiation and side-channeling through a power analysis [11]. Another major concern in an SG network is the limited capabilities of smart meters in terms of communication, energy, and computation. This puts some limitations on the implementation of conventional cryptographic techniques in SG networks. Therefore, ideal SG security approaches should strive to be lightweight in addition to fulfilling numerous security requirements.

1.1. Motivation

It has been shown that a myriad of protocols have been introduced in the smart grid network to preserve its security posture. However, these solutions are based on conventional cryptographic systems such blockchain, public key infrastructure, PUF, and bilinear pairings. All these techniques have many security, performance, or privacy issues and, hence, are not suitable for resource-incapacitated SG devices such as SMs. Attacks such as de-synchronization, impersonation, privacy leaks, replays, and DoS must be prevented, as they adversely interfere with the reliability of smart grids. As such, there is a need for an effective, efficient, and robust security scheme for SGs.

1.2. Threat Model

In this section, we model attacks against our scheme using the most popular Dolev–Yao (DY) and Canetti–Krawczyk models. In these threat models, attacker Ā is capable of the following actions, compromising the private keys belonging to smart meters and service providers:

- Modifying and deleting the contents of intercepted messages;
- Generating and forwarding bogus messages to unsuspecting entities;
- Physically capturing and compromising network entities such as smart meters;
- Retrieving sensitive security tokens stored in the smart meter’s memory;
- Deploying extracted smart meter memory content to execute attacks;
- Intercepting derived session keys and other session state parameters.

1.3. Security Requirements

In the face of numerous security threats and privacy leaks, an ideal authentication scheme for smart grid networks should fulfill the following requirements:

**Mutual authentication:** The identities of all the communicating parties should be reciprocally verified prior to exchanging any network data.
Key agreement: To preserve confidentiality and the integrity of the communication process, a session key should be set up to encrypt all exchanged messages.

Anonymity and untraceability: An attacker should be incapable of discerning the real identity of the communicating entities based on any captured network messages. Additionally, the attacker should be incapable of tracing the communicating parties using these intercepted messages.

Key security: The captured current session key should not facilitate the derivation of past and subsequent session keys.

Formal verification: The derived session key should be mathematically sound.

Resilience against: To offer sufficient security, an ideal authentication protocol needs to withstand attacks such as EPSL, de-synchronization, DoS, eavesdropping, privileged insider, guessing, spoofing, Known Session-Secret Temporary Information (KSSTI), ephemeral secret leakage, physical capture, impersonation, replay, MitM, and forgery.

1.4. Contributions
To address the security, performance, and privacy challenges discussed above, we make the following contributions in our paper.

- We deploy shared keys and pseudo-identities to encipher the communication channel so as to enhance security and privacy preservation.
- To protect against MitM and replay attacks, each entity computes the session keys for traffic protection.
- We deploy BAN logic for the revelation of the probably secure nature of the negotiated session key.

An extensive comparative analysis shows that our protocol withstands the largest number of attacks. In addition, it incurs the lowest computation overheads and relatively lower storage and communication overheads.

The rest of this work is structured as follows: Section 2 discusses the related works in this domain, while our scheme is described in Section 3. On the other hand, Section 4 discusses the security analysis of this protocol, while Section 5 describes its evaluation in terms of performance. Finally, Section 6 presents the conclusions and gives some future research scopes.

2. Related Work
Smart grid security, privacy, and performance have attracted a lot of attention, leading to the introduction of many schemes. For instance, researchers in [10] have presented an identity-based technique, while the authors in [12,13] have developed elliptic curve cryptography (ECC)-based schemes. However, extensive ECC multiplication operations render the schemes in [12,13] inefficient [14]. Therefore, they are not ideal for deployment in computation-limited smart grid components. On the other hand, PUF-based schemes are developed in [15–18]. Although the protocol in [15] withstands modeling attacks, protocols based on PUF have stability issues [19]. In addition, the scheme in [18] offers smart meter physical security but is still vulnerable to EPSL attacks and cannot provide backward key secrecy [17]. To offer smart meter anonymity, a secure scheme is presented in [20]. However, this scheme fails to mutually authenticate the network entities and is prone to DoS attacks [21]. Although the scheme in [22] is anonymity-preserving, it cannot withstand ephemeral secret and session key leakage attacks [23]. In addition, its bilinear pairing operations result in extensive computation overheads [24], similar to the protocols in [23,25].

To reduce the computation overheads associated with bilinear pairings, a scheme based on elliptic curve cryptography is developed in [26]. However, this technique cannot offer anonymity [1] and is defenseless against ephemeral secret leakage attacks [27]. Additionally, it incurs high computation overheads during the generation of security tokens at the Trusted Authority (TA) [1]. On the same breadth, the technique introduced in [28] fails to offer untraceability and identity protection [29]. To deal with these challenges, an anonymous
authentication protocol is introduced in [30]. Although identity protection is assured, this technique incurs high computation costs [6]. To offer efficiency in smart grids, lightweight authentication schemes are developed in [1,6,29,31–34]. However, the schemes in [6,31,32] have not been evaluated against de-synchronization attacks. Similarly, the protocol in [29] has not been evaluated against spoofing and guessing attacks. Although the schemes in [1,33] are resilient against de-synchronization attacks, they have not been evaluated against spoofing attacks. On the other hand, the scheme in [34] cannot withstand desynchronization attacks [29].

To address the anonymity issues in some of the protocols above, a password-based security technique is introduced in [35]. However, this protocol has incorrect login and authentication phases [36]. Although the scheme in [37,38] overcomes this challenge, it is defenseless against de-synchronization threats. In addition, it fails to provide formal security verification and revocability. On the other hand, the usage of some fixed messages in each session in [39,40] renders said session vulnerable to traceability attacks. The protocol in [41] solves this issue by updating this message for each session. However, the service provider needs to buffer previous data for each SM so as to withstand de-synchronization attacks. Consequently, it incurs heavy storage costs especially in networks with massive SMs.

To enhance security in wireless networks, quantum computing technology has been adopted. For instance, based on quantum information engineering, a technique for local energy distribution to numerous remote nodes is presented in [42], while a verification scheme applicable in a quantum channel is developed in [43]. On the other hand, a blind quantum-based protocol is presented in [44], while a zero-knowledge proof is developed in [45]. However, comparative performance analyses have not been carried out in [42–45]. As explained in [46], blockchain technology can ensure privacy and security devoid of an authorized third party. As such, a blockchain-based protocol is presented in [47]. Although blockchain technology provides traceability, improved security, and immutability, it raises serious issues regarding transparency and privacy [48]. In addition, the blockchain-based protocol in [47] lacks evaluation against threats such as privileged insider and physical capture. To avert the misuse and malicious manipulation of battery equipment and data, a robust security scheme is presented in [49]. Although this technique protects against counterfeiting and possible software backdoors, its comparative security and performance evaluations are missing.

Based on the above discussions, it is clear that many schemes have been developed to address security and privacy issues in the smart grid environment. However, most of them still have challenges in terms of privacy, performance, or security. There is, therefore, a need for the development of novel protocols that can help alleviate these challenges.

3. The Proposed Protocol

The network model of our protocol comprises a utility service provider (USP), a trusted control server (TC), and a smart meter (SM), as evidenced in Figure 1. The TCS executes system initialization and generates the secret values for the SM and the USP during the registration phase.

The SM measures electricity usage on the client end and transmits power consumption reports to the USP over public channels. At the USP, these reports are processed and analyzed to facilitate decision making, which may include dynamic power adjustments. Table 1 describes the symbols used throughout this paper.
Figure 1. Proposed network model.

Table 1. Notations.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCS</td>
<td>Trusted control server</td>
</tr>
<tr>
<td>SMᵢ</td>
<td>iᵗʰ smart meter</td>
</tr>
<tr>
<td>USP</td>
<td>Utility service provider</td>
</tr>
<tr>
<td>K_TCS</td>
<td>Master key of the TCS</td>
</tr>
<tr>
<td>ID_TCS</td>
<td>Unique identifier of the TCS</td>
</tr>
<tr>
<td>ID_SM</td>
<td>Unique identifier of the SMᵢ</td>
</tr>
<tr>
<td>K_SM</td>
<td>SMᵢ’ s private key</td>
</tr>
<tr>
<td>Rᵢ</td>
<td>Random nonce i</td>
</tr>
<tr>
<td>PID_SM</td>
<td>Shared key between TCS and SMᵢ</td>
</tr>
<tr>
<td>K_TSM</td>
<td>Unique identity of the USP</td>
</tr>
<tr>
<td>K_USP</td>
<td>USP’s private key</td>
</tr>
<tr>
<td>PID_USP</td>
<td>USP’ s pseudo-identity</td>
</tr>
<tr>
<td>K_UT</td>
<td>Shared key between USP and TCS</td>
</tr>
<tr>
<td>SK_SU</td>
<td>Session key between SMᵢ and USP</td>
</tr>
<tr>
<td>h(.)</td>
<td>Hashing function</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>⊕</td>
<td>XOR operation</td>
</tr>
</tbody>
</table>

Our scheme executes five major steps, which encompass system setup, entity registration, mutual authentication, key negotiation, and parameter refresh phases. Algorithm 1 summarizes this protocol, and the sub-sections that follow give the details of these phases.

Algorithm 1 Secure and efficient authentication

Begin

#************************System setup phase ************************#
(1) Generate K_TCS, ID_TCS, ID_SM & K_SM

#************************Registration phase ************************#
(2) Generate R₁ & derive PID_SM, then Reg⁻¹ TCS
(3) Generate R₂ & compute K_TSM
(4) Store{PID_SM, K_TSM, R₁}, publish PID_SM, then Reg⁻² SMᵢ
(5) Calculate A₁, A₂ & store {A₁, A₂, PID_SM}
(6) Generate R₃, select ID_USP & K_USP, then Reg⁻³ TCS
(7) Compute K_UT & A₃
(8) Store {PID_USP, A₃, K_UT}, then Reg⁻⁴ USP
(9) Calculate A₄, A₅, B₁, B₂ & B₃
(10) Store {A₅, B₁, B₂, B₃}
Algorithm 1 Cont.

# Authentication and key negotiation phase #

(11) Input \(\{ID_{USP}, K_{USP}\}\), then compute \(R_3, A_4 & B_1^*\)

(12) If \(B_1^* = B_1\) then:

(13) Terminate session

(14) Else:

(15) Generate \(R_4\), derive \(A_3, K_{UT}, B_4, B_5\) & \(C_1\), then \(\text{Auth}^{-1} \rightarrow \text{TCS}\)

(16) Retrive \(A_3, K_{UT}\) & derive \((R_4^* \| \text{PID}_{SM}^*), C_1^*\)

(17) If \(C_1^* = C_1\) then:

(18) Abort session

(19) Else:

(20) Generate \(R_5\) & Fetch \(K_{TSM}, R_1\)

(21) Derive \(C_2, C_3, C_4 & C_5\), then \(\text{Auth}^{-2} \rightarrow \text{SM}_i\)

(22) Calculate \(R_1, K_{TSM}, C_2^* & C_5\)

(23) If \(C_5^* = C_5\) then:

(24) Stop session

(25) Else:

(26) Generate \(R_6\), derive \((h(ID_{USP} \| R_3) \| h(ID_{TCS} \| R_5)), SK_{SU}, D_1 & D_2, \text{Auth}^{-3} \rightarrow \text{TCS}\)

(27) Derive \(h(ID_{SM} \| R_6) & D_2^*\)

(28) If \(D_2^* = D_2\) then:

(29) Abort session

(30) Else:

(31) Derive \(SK_{SU}, \text{PID}_{USP}^*, A_3^*, D_3 & D_4\)

(32) Store \(\{\text{PID}_{USP}, A_3\}\) with \(\{\text{PID}_{USP}^*, A_3^*\}\), then \(\text{Auth}^{-4} \rightarrow \text{USP}\)

(33) Calculate \(\text{PID}_{USP}^*, (h(ID_{TCS} \| R_3) \| h(ID_{SM} \| R_6) \| \text{PID}_{USP}^*) \& D_4^*\)

(34) If \(D_4^* = D_4\) then:

(35) Stop session

(36) Else:

(37) Compute \(SK_{SU}, A_3^*, B_2^* & B_3^*\)

(38) Substitute \(\{B_2, B_3, \text{PID}_{USP}\}\) with \(\{B_2^*, B_3^*; \text{PID}_{USP}^*\}\)

(39) Derive \(D_3\)

(40) If \(D_3^* = D_3\) then:

(41) Terminate session

(42) Delete \(\{\text{PID}_{USP}, A_3\}\) from database

(43) End; End;

(44) End; End;

(45) End;
3.1. System Setup

In this phase, the TCS selects its master key as $K_{TCS}$. This is followed by the generation of its unique identity $ID_{TCS}$, the smart meter’s unique identity $ID_{SM}$, as well as the private key of the smart meter, $K_{SM}$, as shown in Figure 2.

![Figure 2. System initialization and registration.](image)

3.2. Registration

In this particular phase, the smart meters are registered at the TCS before they are deployed in the actual field. In addition, the USP is also registered at the TCS prior to exchanging data with the smart meters. The following sub-sections describe this phase in more detail.

3.2.1. Smart Meter Registration

The subsequent three procedures are executed to register the smart meter $SM_i$ to the TCS. To accomplish this, secure communication channels are deployed.

**Step 1:** The $SM_i$ chooses a random nonce $R_1$ to derive its pseudo-identity $PID_{SM} = h(ID_{SM} \| R_1)$. It then composes registration message $Reg-1 = \{PID_{SM}, R_1\}$ that is forwarded to the TCS over secure communication media, as shown in Figure 2.

**Step 2:** When it receives message $Reg-1$, the TCS selects a random nonce $R_2$ and computes $K_{TSM} = h(PID_{SM} \| R_1 \| R_2)$. Next, the TCS stores $\{PID_{SM}, K_{TSM}, R_1\}$ in its repository. Next, registration message $Reg-2 = \{K_{TSM}\}$ is constructed and forwarded to the $SM_i$, as evidenced in Figure 2. Afterwards, the TCS publishes $PID_{SM}$.
Step 3: Upon receiving the message Reg-2, the smart meter SM$_i$ derives $A_1 = R_1 \oplus h(ID_{SM} \| K_{SM})$ and $A_2 = K_{TSM} \oplus h(R_1 \| K_{SM})$. Thereafter, it stores $\{A_1, A_2, PID_{SM}\}$ in its memory.

3.2.2. Utility Service Provider Registration

To register to the TCS, the USP needs to execute the following three procedures.

**Step 1:** The USP chooses its real identity $ID_{USP}$ and secret key $K_{USP}$. Next, it generates a random nonce $R_3$ that is used to calculate its pseudo-identity $PID_{USP} = h(ID_{USP} \| R_3)$. Thereafter, it constructs registration message $Reg-3 = \{PID_{USP}\}$, which is transmitted to the TCS, as depicted in Figure 2.

**Step 2:** After receiving registration message $Reg-3$, the TCS calculates shared key $K_{UT} = h(PID_{USP} \| K_{TCS} \| R_2)$ and $A_3 = h(PID_{USP} \| K_{UT})$. Next, it stores $\{PID_{USP}, A_3, K_{UT}\}$ in its database. Finally, registration message $Reg-4 = \{K_{UT}, A_3\}$ is composed and sent to the USP.

**Step 3:** Upon receiving message $Reg-4$, the USP derives $A_4 = h(K_{USP} \| R_3), A_5 = R_3 \oplus h(ID_{USP} \| K_{USP})$, $B_1 = h(ID_{USP} \| K_{USP} \| R_3 \| A_4), B_2 = A_3 \oplus h(R_3 \| A_4)$, and $B_3 = K_{UT} \oplus h(A_3 \| A_4)$. Next, it stores $\{A_5, B_1, B_2, B_3\}$ in its database.

3.3. Authentication and Key Setup

To securely exchange power consumption reports and adjustment commands, the USP and SM$_i$ must first mutually validate one another. This is followed by the establishment of a session key for message protection over the public internet. The subsequent nine steps are utilized to accomplish these two processes.

**Step 1:** The USP operator supplies parameter set $\{ID_{USP}, K_{USP}\}$, after which values $R_3 = A_3 \oplus h(ID_{USP} \| K_{USP}), A_4 = h(K_{USP} \| R_3)$, and $B_1^* = h(ID_{USP} \| K_{USP} \| R_3 \| A_4)$ are computed. Next, it confirms if $B_1^* \neq B_1$ in a manner such that the communication session is aborted if these two parameters are not identical. Otherwise, the USP randomly generates nonce $R_4$, which is used to derive $A_3 = B_2 \oplus h(R_3 \| A_4), K_{UT} = B_3 \oplus h(A_3 \| A_4), B_4 = h(PID_{USP} \| A_3 \| K_{UT} \oplus (R_4 \| PID_{SM}), B_5 = h(ID_{USP} \| R_4) \oplus h(K_{UT} \| R_4)$, and $C_1 = h(PID_{USP} \| A_3 \| R_4 \| PID_{SM} \| K_{UT})$. At the end, message $Auth-1 = \{PID_{USP}, B_4, B_5, C_1\}$ is constructed and transmitted to the TCS, as shown in Figure 3.

**Step 2:** After receiving message $Auth-1$, TCS retrieves $A_3$ and $K_{UT}$ corresponding to $PID_{USP}$ and derives $(R_4^* \| PID_{SM^*}) = B_4 \oplus h(PID_{USP} \| A_3 \| K_{UT})$ as well as $C_1^* = h(PID_{USP} \| A_3 \| R_4^* \| PID_{SM^*} \| K_{UT})$. Thereafter, the TCS validates if $C_1^* \neq C_1$ such that the communication session is halted when this check flops. If not, the TCS fetches $K_{TSM}$ and $R_1$ corresponding to $PID_{SM}$.

**Step 3:** The TCS randomly generates number $R_5$, which is used to calculate $C_2 = h(R_4 \| R_5), C_3 = h(PID_{SM} \| K_{TSM} \| R_1) \oplus C_2, h(ID_{USP} \| R_4) = B_5 \oplus h(K_{UT} \| R_4), C_4 = h(ID_{USP} \| R_4 \| ID_{TCS} \| R_5) \oplus h(K_{TSM} \| R_1), and C_5 = h(PID_{USP} \| C_2 \| K_{TSM})$. Finally, message $Auth-2 = \{PID_{USP}, C_3, C_4, C_5\}$ is composed and passed over to the SM$_i$.

**Step 4:** After receiving $Auth-2$, SM$_i$ computes $R_1 = A_1 \oplus h(ID_{SM} \| K_{SM}), K_{TSM} = A_2 \oplus h(R_1 \| K_{SM}), C_2^* = h(PID_{SM} \| K_{TSM} \| R_1) \oplus C_3, and C_5 = h(PID_{USP} \| C_2^* \| K_{TSM})$. Next, it confirms whether $C_5^* \neq C_5$ such that the communication session is abandoned upon validation flop. Otherwise, it chooses a random nonce $R_6$ and calculates $(h(ID_{USP} \| R_4) \| h(ID_{TCS} \| R_5) \| h(ID_{SM} \| R_6), D_1 = h(PID_{SM} \| K_{SM} \| R_1) \oplus h(ID_{SM} \| R_6), and D_2 = h(PID_{USP} \| PID_{SM} \| C_2^* \| h(ID_{SM} \| R_6) \| K_{TSM}). Next, message $Auth-3 = \{D_1, D_2\}$ is constructed and forwarded to the TCS.
Figure 3. Authentication and key negotiation.

Step 6: Upon receiving message Auth-3, the TCS calculates $h(\text{ID}_{\text{SM}} \| R_6) = D_1 \oplus h(\text{ID}_{\text{SM}} \| \text{K}_{\text{TSM}} \| R_1)$ and $D_2^* = h(\text{ID}_{\text{USP}} \| \text{ID}_{\text{SM}} \| C_2 \| h(\text{ID}_{\text{SM}} \| R_6) \| \text{K}_{\text{TSM}})$. Next, it checks if $D_2^* = D_2$ so that the authentication process is terminated upon verification.
failure. Otherwise, it computes session key \( SK_{SU} = h(h(ID_{USP} \| R_4) \| h(ID_{TCS} \| R_5) \| h(ID_{SM} \| R_6)) \), new pseudo-identity \( PID_{USP}^* = h(ID_{USP} \| R_4), A_3^* = h(_PID_{USP}^* \| K_{UT}), D_3 = h(A_3 \| R_4) \oplus (h(ID_{TCS} \| R_5) \| h(ID_{SM} \| R_6) \| PID_{USP}^*), \) and \( D_4 = h(_PID_{USP} \| R_4) \| h(ID_{TCS} \| R_5) \| h(ID_{SM} \| R_6) \| PID_{USP}^* \| K_{UT}) \). The TCS stores \{PID_{USP}*, A_3^*\} in its database. At the end, authentication message Auth-4 = \{D_3, D_4\} is composed and sent over to the USP.

**Step 7:** Upon receiving Auth-4, the USP derives \( PID_{USP}^* = h(_PID_{USP} \| R_4), (h(ID_{TCS} \| R_5) \| h(ID_{SM} \| R_6) \| PID_{USP}^*), D_3 = h(A_3 \| R_4) \oplus (h(ID_{TCS} \| R_5) \| h(ID_{SM} \| R_6) \| PID_{USP}^*), \) and \( D_4 = h(_PID_{USP} \| R_4) \| h(ID_{TCS} \| R_5) \| h(ID_{SM} \| R_6) \| PID_{USP}^* \| K_{UT}) \). It then confirms if \( D_4^* \neq D_4 \) such that the authentication is aborted when the verification fails. Otherwise, it derives session key \( SK_{SU} = h(h(ID_{USP} \| R_4) \| h(ID_{TCS} \| R_5) \| h(ID_{SM} \| R_6)) \).

**Step 8:** The USP derives parameters \( A_3^* = h(_PID_{USP}^* \| K_{UT}), B_3^* = A_3^* \oplus h(R_3 \| A_4)) \), and \( B_4^* = K_{UT} \oplus h(A_3^* \| A_4)) \). Next, it replaces \{B_2, B_3, PID_{USP}\} with \{B_2^*, B_3^*, PID_{USP}^*\} in its database. Finally, it derives \( D_5 = h(SK_{SU} \| PID_{USP}^*) \) and transmits it towards the TCS for the subsequent session.

**Step 9:** After receiving \( D_5 \), the TCS recomputes \( D_5^* = h(SK_{SU} \| PID_{USP}^*) \). Next, it confirms if \( D_5^* \neq D_5 \) such that it terminates the session when this validation fails. Otherwise, it deletes parameter set \{PID_{USP}, A_3\} from its database.

### 3.4. Parameter Update

In this phase, the USP’s private key \( K_{USP} \) is updated using the following two steps.

**Step 1:** The operator supplies their unique identity \( ID_{USP} \) as well old secret key \( K_{USP}^{\text{Old}} \). This is followed by the derivation of parameter \( R_3 = A_3 \oplus h(ID_{USP} \| K_{USP}^{\text{Old}})), A_4 = h(K_{USP}^{\text{Old}} \| R_3), \) and \( B_1^* = h(ID_{USP} \| K_{USP}^{\text{Old}} \| R_3 \| A_4)) \). The USP checks if \( B_1^* \neq B_1 \) such that this authentication is halted when this check fails. Otherwise, the operator is prompted to input the new secret key \( K_{USP}^{\text{New}} \).

**Step 2:** The USP derives \( A_3 = B_2 \oplus h(R_3 \| A_4)), K_{UT} = B_3 \oplus h(A_3 \| A_4)), A_4^{\text{New}} = h(K_{USP}^{\text{New}} \| R_3), A_5^{\text{New}} = R_3 \oplus h(ID_{USP} \| K_{USP}^{\text{New}})), B_1^{\text{New}} = h(ID_{USP} \| K_{USP}^{\text{New}} \| R_3 \| A_4^{\text{New}})), B_2^{\text{New}} = A_3 \oplus h(R_3 \| A_4^{\text{New}})), \) and \( B_3^{\text{New}} = K_{UT} \oplus h(A_3 \| A_4^{\text{New}})) \). Lastly, it replaces parameter set \{A_3, B_1, B_2, B_3\} with its refreshed equivalents \{A_3^{\text{New}}, B_1^{\text{New}}, B_2^{\text{New}}, B_3^{\text{New}}\}.

### 4. Security Analysis

In most of the authentication protocols, both formal and informal security analyses are carried out. As such, we present these analyses in this section and provide further details in the sub-sections that follow.

#### 4.1. Formal Security Analysis

To accomplish this analysis, BAN logic is deployed to show that USP and SMi authenticate each other based on fresh and reliable data. Essentially, this involves the verification of the origin, freshness, and legitimacy of the exchanged messages. The notations in Table 2 are used throughout this formal analysis.

**Table 2. BAN logic notations.**

<table>
<thead>
<tr>
<th>Notation</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>( R )</td>
<td>Secret key</td>
</tr>
<tr>
<td>( A_1 \equiv X )</td>
<td>Entity A believes statement X</td>
</tr>
<tr>
<td>( A_1 \sim X )</td>
<td>Entity A once said statement X</td>
</tr>
<tr>
<td>( \langle X \rangle_M )</td>
<td>X is combined with M</td>
</tr>
<tr>
<td>( A \oslash X )</td>
<td>Entity A sees statement X</td>
</tr>
<tr>
<td>( A \Rightarrow X )</td>
<td>Entity A has jurisdiction over X</td>
</tr>
</tbody>
</table>
Table 2. Cont.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td># (X)</td>
<td>Message X is fresh</td>
</tr>
<tr>
<td>(X_R)</td>
<td>Message X is hashed using key (R)</td>
</tr>
<tr>
<td>((X, M))</td>
<td>(X) or (M) is part of formula ((X, M))</td>
</tr>
<tr>
<td>(A \leftrightarrow B)</td>
<td>Entities (A) and (B) share secret key (R)</td>
</tr>
<tr>
<td>(</td>
<td>X</td>
</tr>
<tr>
<td>(A \equiv B)</td>
<td>(R) is only known to (A) and (B)</td>
</tr>
</tbody>
</table>

The BAN logic postulates are described using a number of rules that are detailed in Table 3 below.

Table 3. BAN logic rules.

<table>
<thead>
<tr>
<th>Rule</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>(A \equiv A \land A \equiv (X, X))</td>
<td>Message Meaning Rule (MMR)</td>
</tr>
<tr>
<td>(A \equiv (X, A) \equiv (X, X))</td>
<td>Nonce Verification Rule (NVR)</td>
</tr>
<tr>
<td>(A \equiv (X, M) \equiv (X, X))</td>
<td>Believe Rule (BR)</td>
</tr>
<tr>
<td>(A \equiv (X, A) \equiv (X, X))</td>
<td>Jurisdiction Rule (JR)</td>
</tr>
<tr>
<td>(A \equiv (X, A) \equiv (X, X))</td>
<td>Freshness rule (FR)</td>
</tr>
</tbody>
</table>

Next, we lay bare that our protocol offers protected mutual validation between the \(SM_i\) and the USP. In our protocol, four messages are exchanged during the processes of entity verification and session key setup. These particular messages are idealized as follows:

**Auth-1.** USP \(\rightarrow\) TCS: \([PID_{USP}, B_4, B_5, C_1]\)  
Idealized form: \((PID_{USP}, A_3, R_4)_K_{UT}\)  
**Auth-2.** TCS \(\rightarrow\) SM_i: \([PID_{USP}, C_3, C_4, C_5]\)  
Idealized form: \((PID_{USP}, h (ID_{USP} || R_4), h (ID_{TCS} || R_5), PID_{SM}, R_1)_K_{TSM}\)  
**Auth-3.** SM_i \(\rightarrow\) TCS: \([D_1, D_2]\)  
Idealized form: \((PID_{USP}, PID_{SM}, h (ID_{USP} || R_4), h (ID_{SM} || R_6))_K_{TSM}\)  
**Auth-4.** TCS \(\rightarrow\) USP: \([D_3, D_4]\)  
Idealized form: \((A_3, h (ID_{USP} || R_4), h (ID_{TCS} || R_5), h (ID_{SM} || R_6))_K_{UT}\)  

Using the BAN logic analytic procedures, our scheme should uphold the four security goals (GLs) below.

- **GL_1:** USP \(\equiv (USP \leftrightarrow SM)\)  
- **GL_2:** USP \(\equiv SM \equiv (USP \leftrightarrow SM)\)  
- **GL_3:** SM \(\equiv (USP \leftrightarrow SM)\)  
- **GL_4:** SM \(\equiv USP \equiv (USP \leftrightarrow SM)\)  

To ensure that the BAN logic analysis of our scheme is successfully executed, a number of initial state assumptions (AS) are made as follows.

- **AS_1:** TCS \(\equiv (USP \leftrightarrow TCS)\)  
- **AS_2:** TCS \(\equiv \# (R_4)\)  
- **AS_3:** SM \(\equiv (TCS \leftrightarrow SM)\)  
- **AS_4:** SM \(\equiv \# (R_5)\)  
- **AS_5:** TCS \(\equiv (TCS \leftrightarrow SM)\)  
- **AS_6:** TCS \(\equiv \# (R_5)\)  
- **AS_7:** USP \(\equiv (USP \leftrightarrow TCS)\)  
- **AS_8:** USP \(\equiv \# (R_5)\)
AS9: USP ∥ TCS ⇒ (USP $h(ID_{TCS} || R_5) || h(ID_{SM} || R_6)$ SM)

AS10: SM ∥ TCS ⇒ (USP $h(ID_{USP} || R_4) || h(ID_{TCS} || R_5)$ SM)

AS11: USP ∥ SM ⇒ (USP $SK_{SU} \leftrightarrow$ SM)

AS12: SM ∥ USP ⇒ (USP $SK_{SU} \leftrightarrow$ SM)

Based on message Auth-1, we obtain B1L1.

B1L1: TCS ↯ (PID_{USP}, A_3, R_4)_{K_{UT}}

Deploying B1L1 and AS1 with MMR, B1 _2 is obtained.

B1L2: TCS ∥ USP ↯ (PID_{USP}, A_3, R_4)_{K_{UT}}

Applying FR to B1 _2 and AS2 yields B1L3.

B1L3: TCS ∥ # (PID_{USP}, A_3, R_4)_{K_{UT}}

Using NVR on both B1L2 and B1L3, we obtain B1L4.

B1L4: TCS ∥ USP ∥ (PID_{USP}, A_3, R_4)_{K_{UT}}

From message Auth-2, we can obtain B1L5.

B1L5: SM ↯ (PID_{USP}, h(ID_{USP} || R_4), h(ID_{TCS} || R_5), PID_{SM}, R_1)_{K_{TSM}}

The application of MMR on both B1L5 and AS3 results in B1L6.

B1L6: SM ∥ TCS ∥ (PID_{USP}, h(ID_{USP} || R_4), h(ID_{TCS} || R_5), PID_{SM}, R_1)_{K_{TSM}}

To obtain B1L7, FR is used on B1L6 and AS4.

B1L7: SM ∥ # (PID_{USP}, h(ID_{USP} || R_4), h(ID_{TCS} || R_5), PID_{SM}, R_1)_{K_{TSM}}

On the other hand, NVR is applied to both B1L4 and B1L7 to obtain B1L8.

B1L8: SM ∥ TCS ∥ # (PID_{USP}, h(ID_{USP} || R_4), h(ID_{TCS} || R_5), PID_{SM}, R_1)_{K_{TSM}}

Based on message Auth-3, we can obtain B1L9.

B1L9: TCS ↯ (PID_{USP}, PID_{SM}, h(ID_{USP} || R_4), h(ID_{SM} || R_6))_{K_{TSM}}

Applying MMR on B1L9 and AS5 yields B1L10.

B1L10: TCS ∥ SM ∥ # (PID_{USP}, PID_{SM}, h(ID_{USP} || R_4), h(ID_{SM} || R_6))_{K_{TSM}}

Using FR on B1L10 and AS6 results in B1L11.

B1L11: TCS ∥ # (PID_{USP}, PID_{SM}, h(ID_{USP} || R_4), h(ID_{SM} || R_6))_{K_{TSM}}

On the other hand, NVR is used on both B1L10 and B1L11 to obtain B1L12.

B1L12: TCS ∥ SM ∥ # (PID_{USP}, PID_{SM}, h(ID_{USP} || R_4), h(ID_{SM} || R_6))_{K_{TSM}}

From message Auth-4, we can obtain B1L13.

B1L13: USP ↯ (A_3, h(ID_{USP} || R_4), h(ID_{TCS} || R_5), h(ID_{SM} || R_6))_{K_{UT}}

The application of MMR on B1L13 and AS7 yields B1L14.

B1L14: USP ∥ TCS ∥ (A_3, h(ID_{USP} || R_4), h(ID_{TCS} || R_5), h(ID_{SM} || R_6))_{K_{UT}}

To obtain B1L15, FR is used in both B1L14 and AS8.

B1L15: USP ∥ # (A_3, h(ID_{USP} || R_4), h(ID_{TCS} || R_5), h(ID_{SM} || R_6))_{K_{UT}}

However, using NVR on B1L14 and B1L15 yields B1L16.

B1L16: USP ∥ TCS ∥ # (A_3, h(ID_{USP} || R_4), h(ID_{TCS} || R_5), h(ID_{SM} || R_6))_{K_{UT}}

Since the session key is $SK_{SU} = h(h(ID_{USP} || R_4) || h(ID_{TCS} || R_5) || h(ID_{SM} || R_6))$,

B1L17 can be obtained from B1L12, B1L16, and AS9.

B1L17: USP ∥ SM ∥ (USP $SK_{SU} \leftrightarrow$ SM); hence, GL2 is obtained.

From B1L4, B1L8, and AS10, we obtain B1L18.

B1L18: SM ∥ USP ∥ (USP $SK_{SU} \leftrightarrow$ SM); thus, GL4 is obtained.

Based on B1L17 and AS11, we can obtain B1L19.

B1L19: USP ∥ (USP $SK_{SU} \leftrightarrow$ SM), achieving GL1.

Using B1L18 and AS12, we can obtain B1L20.

B1L20: SM ∥ (USP $SK_{SU} \leftrightarrow$ SM), attaining GL3.

The effectual attainment of all the formulated security objectives implies that the USP, TCS, and SM have executed secure mutual authentication and can now proceed to exchange data.
4.2. Informal Security Analysis

In this sub-section, both the Dolev–Yao (DY) and Canetti–Krawczyk (CK) threat models are deployed to show the robustness of our protocol against typical smart grid attacks. Essentially, we make some assumptions about the attacker’s capabilities and then show how our protocol counters the attacker’s capabilities in both the DY and CK models. These attack capabilities are well articulated in [50].

Theorem 1. Our scheme offers anonymity and untraceability.

Proof. Let us assume that an adversary Ā has eavesdropped on Auth-1 = \{PID_{USP}, B_4, B_5, C_1\}, Auth-2 = \{PID_{USP}, C_3, C_4, C_5\}, Auth-3 = \{D_1, D_2\}, and Auth-4 = \{D_3, D_4\}. Here, \(B_4 = h(PID_{USP} \| A_3 \| K_{UT} \oplus R_4 \| PID_{SM})\), \(B_5 = h(PID_{USP} \| R_4 \| K_{UT} \| R_4)\), \(C_1 = h(PID_{USP} \| A_3 \| R_4 \| PID_{SM} \| R_3)\), \(C_3 = h(PID_{SM} \| K_{TSM} \| R_1 \oplus C_2\), \(C_4 = h(\text{ID}_{USP} \| R_4) \| h(\text{ID}_{TCS} \| R_3) \| h(\text{ID}_{SM} \| R_6))\), \(D_2 = h(PID_{SM} \| K_{UT} \| R_5)\). D_1 = h(PID_{SM} \| K_{UT} \| R_5)\). \(D_3 = h(A_3 \| R_4)\). Consequently, Ā is unable to derive these parameters without knowledge of USP’s real identity (ID_{USP}). The goal is to obtain the real identities of the USP, TCS, and SM_i that can facilitate the tracking of these entities. Evidently, these identities are encapsulated in other parameters (such as nonces R_1, R_4, R_5, and R_6) before being hashed. Towards the end of each session, secret parameter PID_{USP} is updated as PID_{USP}^* = h(PID_{USP} \| R_4). As such, all the messages are dynamic for each session. □

Theorem 2. Spoofing and impersonation attacks are thwarted.

Proof. The main objective of these attacks is to spoof exchanged messages so as to masquerade oneself as a legitimate network entity. The following three cases demonstrate the resilience of our scheme against these threats. □

Case 1: Suppose that Ā wants to impersonate the USP through the interception of message Auth-1 = \{PID_{USP}, B_4, B_5, C_1\} sent from the USP towards the TCS over public channels. Here, \(B_4 = h(PID_{USP} \| A_3 \| K_{UT} \oplus R_4 \| PID_{SM})\), \(B_5 = h(PID_{USP} \| R_4 \| K_{UT} \| R_4)\), and \(C_1 = h(PID_{USP} \| A_3 \| R_4 \| PID_{SM} \| K_{UT})\). However, Ā is unable to derive these parameters without knowledge of USP’s real identity (ID_{USP}). The shared key between USP and TCS (K_{UT}), and random nonce R_4, among other values.

Case 2: Let us assume that Ā has intercepted messages Auth-2 = \{PID_{USP}, C_3, C_4, C_5\} and Auth-4 = \{D_3, D_4\} transmitted from the TCS towards the SM_i, respectively. Here, \(C_3 = h(PID_{SM} \| K_{TSM} \| R_1)\), \(C_4 = h(\text{ID}_{USP} \| R_1) \| h(\text{ID}_{TCS} \| R_3)\), \(C_5 = h(PID_{USP} \| K_{TSM} \| R_1)\), \(D_1 = h(PID_{SM} \| K_{TSM} \| R_1)\), \(D_2 = h(PID_{SM} \| K_{TSM} \| K_{UT})\). Afterwards, an attempt is made to construct bogus messages \(PID_{USP}^b, C_3^b, C_4^b, C_5^b\) and \(D_3^b, D_4^b\). However, without TCS’ real identity (ID_{TCS}), random nonces (R_1, R_4, R_5, and R_6), and shared key K_{TSM}, among other parameters, the derivation of these messages flops.

Case 3: Suppose that Ā has captured message Auth-3 = \{D_1, D_2\} sent from SM_i towards TCS over public channels. Here, \(D_1 = h(PID_{SM} \| K_{TSM} \| R_1)\), \(D_2 = h(PID_{SM} \| K_{TSM} \| C_4^b \| h(\text{ID}_{SM} \| R_6))\). Similar to Case 2 above, Ā cannot construct valid message Auth-3 without knowledge of SM_i’s real identity (ID_{SM}), shared key (K_{TSM}), and random nonces (R_1 and R_6).

Theorem 3. Strong mutual entity verification is executed.

Proof. In the proposed approach, all the network parties mutually authenticate one another. For instance, upon receiving message Auth-1 = \{PID_{USP}, B_4, B_5, C_1\} from the USP, the TCS computes \(C_1 = h(PID_{USP} \| A_3 \| R_4^* \| PID_{SM}^* \| K_{UT})\) and validates USP by
checking if $C_1' \neq C_1$. Conversely, upon receiving $Auth-2 = [PID_{USP}, C_3, C_4, C_5]$ from the TCS, the SM$_i$ computes $C_5 = h(PID_{USP} \| C_2' \| K_{TSM})$ and verifies the TCS by confirming whether $C_5' \neq C_i$. Similarly, the TCS receives message $Auth-3 = [D_2]$ from SM$_i$, derives $D_2' = h(PID_{USP} \| PID_{SM} \| C_2 \| h(ID_{SM} \| R_6) \| K_{TSM})$ and authenticates SM$_i$ by checking if $D_2' \neq D_2$. In contrast, the USP obtains message $Auth-4 = [D_3, D_4]$ from the TCS, computes $D_4' = h(PID_{USP} \| C_1 \| h(ID_{TCS} \| R_4) \| h(ID_{SM} \| R_6) \| PID_{USP} \| K_{UT})$, and validates the TCS by confirming if $D_4' \neq D_4$. □

**Theorem 4.** The communicating entities negotiate session keys.

**Proof.** In our protocol, the TCS, SM$_i$, and USP autonomously calculate the session key $SK_{SU} = h(h(ID_{USP} \| R_4) \| h(ID_{TCS} \| R_3) \| h(ID_{SM} \| R_6))$. For instance, after receiving message $Auth-2$ from the TCS, the SM$_i$ computes the session key as $SK_{SU} = h(h(ID_{USP} \| R_4) \| h(ID_{TCS} \| R_3) \| h(ID_{SM} \| R_6))$, together with parameters $D_1$ and $D_2$. However, upon receiving $Auth-3$ from the SM$_i$, the TCS computes the session key as $SK_{SU} = h(h(ID_{USP} \| R_4) \| h(ID_{TCS} \| R_3) \| h(ID_{SM} \| R_6))$, together with values $PID_{USP}'$, $A_3'$, $D_3$, and $D_4$. Similarly, the USP receives message $Auth-4$ from the TCS and computes the session key as $SK_{SU} = h(h(ID_{USP} \| R_4) \| h(ID_{TCS} \| R_3) \| h(ID_{SM} \| R_6))$, together with values $A_3'$, $B_3'$, and $B_5'$. □

**Theorem 5.** Our scheme can withstand forgery and eavesdropping attacks.

**Proof.** Let us assume that adversary $\hat{A}$ wants to forge session key $SK_{SU} = h(h(ID_{USP} \| R_4) \| h(ID_{TCS} \| R_3) \| h(ID_{SM} \| R_6))$. Evidently, $\hat{A}$ must have access to identities $ID_{USP}, ID_{TCS},$ and $ID_{SM}$. In addition, random nonces $R_4, R_5,$ and $R_6$ must be obtained by $\hat{A}$. However, these identities and nonces cannot be obtained by eavesdropping messages $Auth-1 = [PID_{USP}, B_4, B_5, C_1], Auth-2 = [PID_{USP}, C_3, C_4, C_5], Auth-3 = [D_1, D_2], and Auth-4 = [D_3, D_4]$ exchanged over public channels. Let us assume that $\hat{A}$ has captured long-term secret keys $K_{TCS}, K_{UT}, K_{TSM},$ and $K_{SM}$. However, none of these keys is incorporated in the negotiated session key $SK_{SU}$. As such, the session keys derived in our protocol are secured. □

**Theorem 6.** MitM and replay attacks are thwarted.

**Proof.** Suppose that $\hat{A}$ has the ability of intercepting and modifying authentication messages $Auth-1 = [PID_{USP}, B_4, B_5, C_1], Auth-2 = [PID_{USP}, C_3, C_4, C_5], Auth-3 = [D_1, D_2],$ and $Auth-4 = [D_3, D_4]$ exchanged over insecure public channels. Here, $B_4 = h(PID_{USP} \| A_3 \| K_{UT}) \oplus (R_4 \| PID_{SM}), B_5 = h(ID_{USP} \| (R_4) \oplus h(K_{UT} \| R_3)), C_1 = h(PID_{USP} \| A_3 \| R_4 \| PID_{SM} \| K_{UT}), C_2 = h(PID_{SM} \| K_{TSM} \| R_3), C_3 = h(ID_{USP} \| R_4) \| h(ID_{TCS} \| R_5) \| h(K_{TSM} \| R_1), C_4 = h(PID_{USP} \| C_2 \| K_{TSM}), D_1 = h(PID_{SM} \| K_{TSM} \| R_1) \oplus h(ID_{SM} \| R_6), D_2 = h(PID_{USP} \| PID_{SM} \| C_2 \| h(ID_{SM} \| R_6) \| K_{TSM}), D_3 = h(A_3 \| R_4) \oplus (h(ID_{TCS} \| R_3) \| h(ID_{SM} \| R_6) \| PID_{USP}),$ and $D_4 = h(PID_{USP} \| R_4 \| h(ID_{TCS} \| R_3) \| h(ID_{SM} \| R_6) \| PID_{USP} \| K_{UT})$. It is clear that all these messages incorporate random nonces such as $R_1, R_4, R_5,$ and $R_6$. In addition, any successful modification of these messages requires knowledge of identities ($ID_{USP}, ID_{TCS}, ID_{SM}$) and shared keys ($K_{UT}, K_{TSM}$), all of which are unavailable to $\hat{A}$. □

**Theorem 7.** Privileged insider attacks are effectively prevented.

**Proof.** Let us assume that some privileged insider $\hat{A}$ has accessed USP’s pseudo-identity ($PID_{USP}$) during the registration phase. In addition, $\hat{A}$ has access to $\{A_5, B_1, B_2, B_3\}$ stored in the USP’s database. With all these parameters, $\hat{A}$ makes some attempts in deriving session key $SK_{SU} = h(h(ID_{USP} \| R_4) \| h(ID_{TCS} \| R_3) \| h(ID_{SM} \| R_6)).$ However, $\hat{A}$ does not know real identities ($ID_{USP}, ID_{TCS}, ID_{SM}$) and random nonces ($R_4, R_5, R_6$). Therefore, this attack will fail. □
Theorem 8. The proposed scheme can resist de-synchronization and backdoor-based DoS attacks.

Proof. The objective of these threats is to alter and block exchanged messages so as to interfere with future mutual verification processes among the USP, TCS, and SM. This can be occasioned by some SG and SM firmware-containing backdoors. Suppose that \( \hat{\Delta} \) wants to de-synchronize the next authentication session by modifying Auth-1, Auth-2, and Auth-3. However, Theorem 6 demonstrates the difficulty in modifying these messages.

Theorem 11. The proposed protocol can withstand physical attacks.

Proof. The assumption made here is that adversary \( \hat{\Delta} \) has physically obtained the SM, upon which the stored values \( \{A_1, A_2, \text{PID}_{USP}\} \) in its memory are extracted via a power analysis. Here, \( A_1 = R_1 \oplus h (\text{ID}_{SM} || K_{SM}), A_2 = K_{TSM} \oplus h (R_1 || K_{SM}), \) and \( \text{PID}_{SM} = h (\text{ID}_{SM} || R_1). \) The next objective is to ascertain SM’s identity (IDSM), shared key (K_{TSM}), and SM’s private key (K_{SM}). However, these values are masked with random nonces before being hashed. Since reversing the one-way hashing function is computationally cumbersome, our scheme is robust against physical attacks. \( \square \)
5. Performance Evaluations

Storage, computation, supported security, and privacy features, as well as communication complexities are most often utilized as metrics to evaluate authentication protocols. As such, we deploy such metrics in our comparative performance evaluations as detailed below.

5.1. Computation Overheads

During the mutual verification and key setup phase, our scheme executes only one-way hashing ($T_H$) operations. Specifically, $7T_H$ and $16T_H$ operations are executed on the smart meter and utility service provider sides, respectively. The time complexities of the diverse cryptographic functions in the smart meter are computed on a 1 GB RAM, 1.2 GHz CPU, Quad-core Raspberry Pi-3, while the USP cryptographic primitives are computed on an 8 GB RAM, Core i7-6700 laptop equipped with a 3.40 GHz CPU. Under these two environments, the execution durations are presented in Table 4.

Table 4. Execution durations.

<table>
<thead>
<tr>
<th>Scheme</th>
<th>SM Costs (ms)</th>
<th>USP Costs (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bilinear pairing operations, $T_{BP}$</td>
<td>95.72100</td>
<td>9.52800</td>
</tr>
<tr>
<td>ECC point addition, $T_{ECA}$</td>
<td>0.13400</td>
<td>0.00700</td>
</tr>
<tr>
<td>One-way hash function, $T_H$</td>
<td>0.34500</td>
<td>0.03900</td>
</tr>
<tr>
<td>ECC point multiplication, $T_{PM}$</td>
<td>2.70000</td>
<td>0.70500</td>
</tr>
<tr>
<td>Symmetric encryption, $T_{SE}$</td>
<td>0.41000</td>
<td>0.00460</td>
</tr>
<tr>
<td>Symmetric decryption, $T_{SD}$</td>
<td>0.41000</td>
<td>0.00460</td>
</tr>
<tr>
<td>Esch256 one-way hash function, $T_{HE}$</td>
<td>0.33000</td>
<td>0.03200</td>
</tr>
<tr>
<td>Physically unclonable function, $T_{PUF}$</td>
<td>0.00049</td>
<td>-</td>
</tr>
<tr>
<td>Counter-mode encryption with authentication tag, $T_{COP}$</td>
<td>0.34900</td>
<td>0.04100</td>
</tr>
<tr>
<td>Bio-metric key generation and reproduction, $T_{REP}$</td>
<td>2.70000</td>
<td>0.70500</td>
</tr>
<tr>
<td>Modular exponential, $T_E$</td>
<td>30.7920</td>
<td>0.31200</td>
</tr>
<tr>
<td>Scalar multiplication, $T_{SM}$</td>
<td>2.70000</td>
<td>0.70500</td>
</tr>
</tbody>
</table>

Using the execution durations in Table 4 as a basis, the total computation complexity of our scheme is 2.805 ms. Table 5 details the derivation and comparison of the computation complexities of other peer approaches.

Table 5. Computation complexities.

<table>
<thead>
<tr>
<th>Scheme</th>
<th>SM Total (ms)</th>
<th>USP Total (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baghestani et al. [1]</td>
<td>$17T_H + 4T_{PM} = 8.964$</td>
<td>$17T_H + 4T_{PM} = 8.964$</td>
</tr>
<tr>
<td>Xia et al. [6]</td>
<td>$10T_H + 8T_{PM} = 63.285$</td>
<td>$10T_H + 8T_{PM} = 63.285$</td>
</tr>
<tr>
<td>Mohammadali et al. [10]</td>
<td>$7T_H + 5T_{PM} = 8.706$</td>
<td>$7T_H + 5T_{PM} = 8.706$</td>
</tr>
<tr>
<td>Kumar et al. [13]</td>
<td>$17T_H + 4T_{PM} = 8.769$</td>
<td>$17T_H + 4T_{PM} = 8.769$</td>
</tr>
<tr>
<td>Tsai &amp; Lo [22]</td>
<td>$11T_H + 4T_{PM} = 8.769$</td>
<td>$11T_H + 4T_{PM} = 8.769$</td>
</tr>
<tr>
<td>Tanveer &amp; Alasmari [29]</td>
<td>$2T_{HE} + 7T_{PM} + 2T_E + 10T_{HE} ≈ 237.381$</td>
<td>$7T_{HE} + 2T_{PM} + 2T_E + 10T_{HE} ≈ 237.381$</td>
</tr>
<tr>
<td>Chaudhry et al. [31]</td>
<td>$2T_{HE} + 7T_{PM} + 2T_E + 10T_{HE} ≈ 237.381$</td>
<td>$2T_{HE} + 7T_{PM} + 2T_E + 10T_{HE} ≈ 237.381$</td>
</tr>
<tr>
<td>Taqi &amp; Jalili [32]</td>
<td>$2T_{HE} + 7T_{PM} + 2T_E + 10T_{HE} ≈ 237.381$</td>
<td>$2T_{HE} + 7T_{PM} + 2T_E + 10T_{HE} ≈ 237.381$</td>
</tr>
<tr>
<td>Chen et al. [33]</td>
<td>$7T_{HE} + 2T_{PM} + 2T_E + 10T_{HE} ≈ 237.381$</td>
<td>$7T_{HE} + 2T_{PM} + 2T_E + 10T_{HE} ≈ 237.381$</td>
</tr>
<tr>
<td>Park et al. [37]</td>
<td>$16T_H + 2T_{SM} + 2T_E + 2T_{SM} = 3.1898$</td>
<td>$16T_H + 2T_{SM} + 2T_E + 2T_{SM} = 3.1898$</td>
</tr>
<tr>
<td>Proposed</td>
<td>$16T_H + 2T_{SM} + 2T_E + 2T_{SM} = 3.1898$</td>
<td>$16T_H + 2T_{SM} + 2T_E + 2T_{SM} = 3.1898$</td>
</tr>
</tbody>
</table>

As demonstrated in Figure 4, the technique in [22] has the longest execution time of 237.381 ms. This can be explained by the computationally extensive bilinear pairings in [22]. This is followed by the protocols in [6], [31], [32], [1], [13], [47], [10], [29], and [33] respectively. Conversely, our protocol incurs the least computation complexities.
Figure 4. Computational complexities [1,6,10,13,22,29,31–33,47].

Even though the approach in [33] has a relatively lower execution time, it cannot withstand guessing, KSSTI, eavesdropping, ephemeral secret leakage, spoofing, and physical capture attacks. In the SG environment, the majority of components does not have a high computation power; hence, our protocol is the most suitable for deployment.

5.2. Communication Overheads

In our scheme, messages Auth-1, Auth-2, Auth-3, and Auth-4 are exchanged during the verification and key setup phase. The specific details of these messages are as follows.

- Auth-1 = \{PID_{USP}, B_4, B_5, C_1\}
- Auth-2 = \{PID_{USP}, C_3, C_4, C_5\}
- Auth-3 = \{D_1, D_2\}
- Auth-4 = \{D_5, D_4\}

Here, \(PID_{USP} = h(ID_{USP} \| R_3), B_4 = h(PID_{USP} \| A_3 \| K_{UT}) \oplus (R_4 \| PID_{SM}), B_5 = h(ID_{USP} \| R_4) \oplus h(K_{UT} \| R_4), C_1 = h(PID_{USP} \| A_3 \| R_4 \| PID_{SM} \| K_{UT}), C_3 = h(PID_{SM} \| K_{SM} \| R_4) \oplus C_2, C_4 = (h(ID_{USP} \| R_4) \| h(ID_{TCS} \| R_3) \oplus h(K_{SM} \| R_4), C_5 = h(PID_{USP} \| C_2 \| K_{SM}), D_1 = h(PID_{SM} \| K_{SM} \| R_4) \oplus h(ID_{SM} \| R_4) \| C_1, D_2 = h(PID_{USP} \| PID_{SM} \| C_4 \| h(ID_{SM} \| R_4) \| K_{SM}), D_3 = h(A_3 \| R_4) \oplus (h(ID_{TCS} \| R_3) \| h(ID_{SM} \| R_4) \| PID_{USP}^*),\) and \(D_4 = h(PID_{USP} \| R_4) \| h(ID_{TCS} \| R_3) \| h(ID_{SM} \| R_4) \| PID_{USP}^* \| K_{UT}).\)

Using the values in [23,33,39], the hashing, symmetric encryption, point multiplication, timestamps, and symmetric decryption output lengths are 160 bits, 128 bits, 320 bits, 32 bits, and 128 bits, correspondingly. As such, Auth-1 = 160 + 160 + 160 = 480 bits; Auth-2 = 160 + 160 + 160 + 160 = 640 bits; Auth-3 = 160 + 160 = 320 bits; and Auth-4 = 160 + 160 = 320 bits. Consequently, the overall communication complexity of our technique is 1920 bits. Table 6 presents the comparative analysis of the incurred communication complexities of our protocol together with those of its peer approaches.

Table 6. Communication complexities.

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Messages Exchanged</th>
<th>Total (Bits)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bagheestani et al. [1]</td>
<td>SM $\Rightarrow$ USP $\oplus$ USP $\Rightarrow$ SM $\oplus$ USP $\Rightarrow$ USP</td>
<td>1696</td>
</tr>
<tr>
<td>Xia et al. [6]</td>
<td>SM $\Rightarrow$ USP $\oplus$ SM $\Rightarrow$ SM $\oplus$ SM $\Rightarrow$ USP $\Rightarrow$ USP</td>
<td>2816</td>
</tr>
<tr>
<td>Mohammadali et al. [10]</td>
<td>SM $\Rightarrow$ USP $\oplus$ USP $\Rightarrow$ SM $\oplus$ USP $\Rightarrow$ USP</td>
<td>1536</td>
</tr>
<tr>
<td>Kumar et al. [13]</td>
<td>SM $\Rightarrow$ USP $\oplus$ USP $\Rightarrow$ SM $\oplus$ USP $\Rightarrow$ USP</td>
<td>1376</td>
</tr>
</tbody>
</table>
In addition, it does not offer entity untraceability and anonymity. Finally, the scheme in [31] is not robust against spoofing, de-synchronization, DoS, privileged insider, guessing, KSSTI, eavesdropping, EPSL, physical capture, ephemeral secret leakage, spoofing, replay, and forgery. In the same breadth, the protocol in [47] is not analyzed against attacks such as de-synchronization, privileged insider, guessing, eavesdropping, physical capture, ephemerel secret leakage, spoofing, replay, and forgery. In addition, it does not offer anonymity. On its part, the approach in [13] fails to provide session key agreement and mutual authentication. In addition, it is not analyzed against de-synchronization, DoS, privileged insider, guessing, KSSTI, eavesdropping, spoofing, and forgery attacks. Concerning the protocol in [33], it is defenseless against guessing, KSSTI, eavesdropping, EPSL, spoofing, and physical capture attacks. Likewise, the protocol in [1] cannot withstand privileged insider, physical capture, guessing, KSSTI, eavesdropping, spoofing, and forgery attacks. Regarding the protocol in [10], it cannot protect against DoS, spoofing, privileged insider, guessing, KSSTI, eavesdropping, EPSL, physical capture, and forgery.

As evidenced in Figure 5, the technique in [6] exhibits the largest communication overheads of 2816 bits. This is followed by the protocols in [32], our proposed scheme, [33], [1], [10], [31], [13], [47], [22], and [29], in this order. Even though the technique in [29] incurs the lowest communication overheads, its design does not consider guessing, eavesdropping, and spoofing attacks. Similarly, the security scheme in [22] is defenseless against privileged insider, de-synchronization, DoS, guessing, spoofing, KSSTI, eavesdropping, EPSL, physical capture, and forgery attacks. In the same breadth, the protocol in [47] is not analyzed against attacks such as de-synchronization, privileged insider, DoS, guessing, eavesdropping, physical capture, ephemeral secret leakage, spoofing, replay, and forgery. In addition, it does not offer anonymity. On its part, the approach in [13] fails to provide session key agreement and mutual authentication. In addition, it is not analyzed against de-synchronization, DoS, privileged insider, guessing, KSSTI, eavesdropping, spoofing, and forgery attacks. Concerning the protocol in [33], it is defenseless against guessing, KSSTI, eavesdropping, EPSL, spoofing, and physical capture attacks. Likewise, the protocol in [1] cannot withstand privileged insider, physical capture, guessing, KSSTI, eavesdropping, spoofing, and forgery attacks. Regarding the protocol in [10], it cannot protect against DoS, spoofing, privileged insider, guessing, KSSTI, eavesdropping, EPSL, physical capture, and forgery.

**Table 6. Cont.**

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Messages Exchanged</th>
<th>Total (Bits)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tsai &amp; Lo [22]</td>
<td>SM 480 USP 480 SM 220 USP</td>
<td>1280</td>
</tr>
<tr>
<td>Tanveer &amp; Alasmary [29]</td>
<td>USP 544 TCS 662 SM</td>
<td>1206</td>
</tr>
<tr>
<td>Chaudhry et al. [31]</td>
<td>SM 768 USP 768 SM</td>
<td>1536</td>
</tr>
<tr>
<td>Chen et al. [33]</td>
<td>SM 864 USP 768 SM 160 USP 160 SM</td>
<td>1888</td>
</tr>
<tr>
<td>Park et al. [47]</td>
<td>SM 512 USP 672 SM 192 USP</td>
<td>1376</td>
</tr>
<tr>
<td>Proposed</td>
<td>USP 640 TCS 752 SM 320 TCS 320 USP</td>
<td>1920</td>
</tr>
</tbody>
</table>

As evidenced in Figure 5, the technique in [6] exhibits the largest communication overheads of 2816 bits. This is followed by the protocols in [32], our proposed scheme, [33], [1], [10], [31], [13], [47], [22], and [29], in this order. Even though the technique in [29] incurs the lowest communication overheads, its design does not consider guessing, eavesdropping, and spoofing attacks. Similarly, the security scheme in [22] is defenseless against privileged insider, de-synchronization, DoS, guessing, spoofing, KSSTI, eavesdropping, EPSL, physical capture, and forgery attacks. In the same breadth, the protocol in [47] is not analyzed against attacks such as de-synchronization, privileged insider, DoS, guessing, eavesdropping, physical capture, ephemeral secret leakage, spoofing, replay, and forgery. In addition, it does not offer anonymity. On its part, the approach in [13] fails to provide session key agreement and mutual authentication. In addition, it is not analyzed against de-synchronization, DoS, privileged insider, guessing, KSSTI, eavesdropping, spoofing, and forgery attacks. Concerning the protocol in [33], it is defenseless against guessing, KSSTI, eavesdropping, EPSL, spoofing, and physical capture attacks. Likewise, the protocol in [1] cannot withstand privileged insider, physical capture, guessing, KSSTI, eavesdropping, spoofing, and forgery attacks. Regarding the protocol in [10], it cannot protect against DoS, spoofing, privileged insider, guessing, KSSTI, eavesdropping, EPSL, physical capture, and forgery.

**Figure 5. Communication complexities [1,6,10,13,22,29,31–33,47].**

In addition, it cannot offer entity untraceability and anonymity. Finally, the scheme in [31] is not robust against spoofing, de-synchronization, DoS, privileged insider, guessing,
eavesdropping, ESPL, and forgery attacks. Evidently, our protocol provides a good balance between security and communication complexity.

5.3. Storage Overheads

In our scheme, value sets \{A_5, B_1, B_2, B_3\} and \{A_1, A_2, PID_{SM}\} are stored in the USP database and smart meter memory, respectively. Here, \(A_5 = B_1 = B_2 = B_3 = A_1 = A_2 = PID_{SM} = 160\) bits. Consequently, the cumulative storage complexity in our scheme is 1120 bits, or 140 bytes. Table 7 shows the derivation of the storage complexities of our scheme as well as those ones of its peers.

Table 7. Storage overheads.

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Stored Parameters</th>
<th>Total (Bits)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baghestani et al. [1]</td>
<td>SM: {H_1, H_2, n, E, P, F_p, SM_{aj}, x_i, y_i} USP: {SM_{IDp}, M_{kj}}</td>
<td>2432</td>
</tr>
<tr>
<td>Xia et al. [6]</td>
<td>SM: {x_5, R_2} USP: {c}</td>
<td>896</td>
</tr>
<tr>
<td>Mohammadali et al. [10]</td>
<td>SM: {SM, R_{M}, Y_{M}, P_{M}} USP: {y_{AHE}, \tau_{AHE}}</td>
<td>1600</td>
</tr>
<tr>
<td>Kumar et al. [13]</td>
<td>USP: {RID_i, TC_i, {RID_i</td>
<td>i = 1, 2, \ldots, l}, h(), E_p (a, b), G}</td>
</tr>
<tr>
<td>Tsai &amp; Lo [22]</td>
<td>USP: {G_1, G_2, P, e, H, H_1, H_2, H_3, H_4, q, P_{pub}, g}, K_i, H_1 (SID)<em>jP + P</em>{pub}</td>
<td>6112</td>
</tr>
<tr>
<td>Tanveer &amp; Alasmary [29]</td>
<td>SM: {CH_{SMi}, TID_{SMi}, RN_r, HD}</td>
<td>1056</td>
</tr>
<tr>
<td>Chaudhry et al. [31]</td>
<td>USP: {SID_i, B_i, RN} SM: {E, P, F_p, n, SM_{pr}, a_i, idST_i, ST_i, H(), SM_{IDp}, P_{idst}}</td>
<td>2176</td>
</tr>
<tr>
<td>Taqi &amp; Jalili [32]</td>
<td>SM: {a_i, A_i} USP: {a_j, A_j}</td>
<td>896</td>
</tr>
<tr>
<td>Chen et al. [33]</td>
<td>SM: {ID_i, N_1, X_i} USP: {S}</td>
<td>832</td>
</tr>
<tr>
<td>Park et al. [47]</td>
<td>USP: {PCUID_j, H, E(a, b), G, PID_{s=1,..l}}</td>
<td>2240</td>
</tr>
<tr>
<td>Proposed</td>
<td>SM: {A_1, A_2, PID_{SM}} USP: {A_5, B_1, B_2, B_3}</td>
<td>1120</td>
</tr>
</tbody>
</table>

The specific details of the various parameters stored in the related schemes are described in Table 8.

As revealed in Figure 6, the approach in [22] incurs the highest storage complexity of 6112 bits. This is followed by the protocols in [1], [47], [13], [31], [10], the proposed scheme, [29], [6], [32], and [33] respectively. The high storage cost in [22] is due to the numerous security tokens that have to be stored in the end devices.
Table 8. Details of stored parameters.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_S$, $SM_{ai}$, $SM_{pi}$, $SM$</td>
<td>$SM$'s private keys</td>
</tr>
<tr>
<td>$R_M$, $R_2$</td>
<td>$SM$'s public key</td>
</tr>
<tr>
<td>$x_C$, $K_j$</td>
<td>Keying parameter based on smart meter’s public key</td>
</tr>
<tr>
<td>$H_1$, $H_2$, $H$, $h$ ($\cdot$), $h_1$, $H_2$, $H_3$, $H_4$</td>
<td>USP’s private keys</td>
</tr>
<tr>
<td>$n$, $E$, $P$</td>
<td>One-way hash functions</td>
</tr>
<tr>
<td>$F_P$</td>
<td>Elliptic curve $E$ and a point $P$ of order $n$</td>
</tr>
<tr>
<td>$x_i$, $y_i$, $X_i$, $LS_{SM_i}$, $\sigma_j$, $ST_j$, $A_i$, $A_j$, $SID_i$, $B_i$, $y_M$, $YAHE$, $\mathcal{G}$</td>
<td>Finite field</td>
</tr>
<tr>
<td>$SM_{ID_j}$, $ID_i$, $SM_{ID_i}$</td>
<td>Derived intermediary parameters</td>
</tr>
<tr>
<td>$idST_j$</td>
<td>$SM$'s unique identity</td>
</tr>
<tr>
<td>$SID_i$</td>
<td>Unique identifier for $SM$</td>
</tr>
<tr>
<td>$M_k$</td>
<td>USP’s unique identity</td>
</tr>
<tr>
<td>$N_1$, $a_i$, $q_i$, $RN_i$, $r_M$, $r_{AHE}$</td>
<td>Master key</td>
</tr>
<tr>
<td>$S_i$, $PID_i$, $PID_{SM_i}$, $TID_i$, $RID_i$</td>
<td>Random numbers</td>
</tr>
<tr>
<td>$PC_{UID_i}$, $RID_i$</td>
<td>SM’s unique identification stored in the table</td>
</tr>
<tr>
<td>$TC_i$, $TC_j$</td>
<td>Pseudo-identities for $SM$</td>
</tr>
<tr>
<td>$E(a, b)$, $G$, $E_P$ ($a$, $b$)</td>
<td>Pseudo-identities for USP</td>
</tr>
<tr>
<td>$P$, $G_1$, $G_2$</td>
<td>SM’s temporal credential</td>
</tr>
<tr>
<td>$q$</td>
<td>USP’s temporal credential</td>
</tr>
<tr>
<td>$e$</td>
<td>Elliptic curve with base point $G$.</td>
</tr>
<tr>
<td>$P_{pub}$</td>
<td>Generator of $G_1$, cyclic additive group, and cyclic multiplicative group, respectively</td>
</tr>
<tr>
<td>$CH_{SM_i}$</td>
<td>Prime order of $G_1$ and $G_2$</td>
</tr>
<tr>
<td>HD</td>
<td>Pairing operation</td>
</tr>
</tbody>
</table>

Figure 6. Storage complexities [1,6,10,13,22,29,31–33,47].
Although the protocols in [6,29,32,33] have slightly lower storage complexities compared to our scheme, they are susceptible to numerous threats, as shown in Table 9. Since smart devices such as SMs in the grid system have limited storage, our scheme is ideal for implementation in this environment.

Table 9. Supported functionalities.

<table>
<thead>
<tr>
<th>Functionality</th>
<th>[10]</th>
<th>[13]</th>
<th>[22]</th>
<th>[29]</th>
<th>[6]</th>
<th>[1]</th>
<th>[31]</th>
<th>[32]</th>
<th>[33]</th>
<th>[47]</th>
<th>Proposed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Session key agreement</td>
<td>√</td>
<td>×</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>Anonymity and untraceability</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>×</td>
<td>√</td>
<td>√</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Key security</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Mutual authentication</td>
<td>√</td>
<td>×</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>×</td>
</tr>
<tr>
<td>Formal verification</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>×</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>Resilience against</td>
<td></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</td>
<td>√</td>
<td>×</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>√</td>
</tr>
<tr>
<td>Backdoor-based DoS</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>×</td>
<td>√</td>
</tr>
<tr>
<td>Privileged insider</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>×</td>
<td>√</td>
</tr>
<tr>
<td>Guessing</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>×</td>
<td>√</td>
</tr>
<tr>
<td>KSSTI</td>
<td>×</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>×</td>
<td>√</td>
</tr>
<tr>
<td>Eavesdropping</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>√</td>
</tr>
<tr>
<td>Eavesdropping</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>×</td>
<td>√</td>
</tr>
<tr>
<td>Eavesdropping</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>×</td>
<td>√</td>
</tr>
<tr>
<td>Physical capture</td>
<td>×</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</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>√</td>
</tr>
<tr>
<td>Replay</td>
<td>√</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</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>Forgery</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
</tbody>
</table>

√ Feature supported; × Feature not supported or not considered.

5.4. Supported Functionalities

The protocol developed in this paper offers a wide range of salient security and privacy features and is robust against several attacks. Table 9 provides a comparative evaluation of the security characteristics of our scheme as well as its resilience to attacks. As revealed in Table 9, the scheme in [6] supports only six features and, hence, is the least secure. This is followed by the protocol in [47], which supports seven features. In contrast, the schemes in [10,13,22] support eight features and, hence, have been rated third. This is followed by the protocols in [32], [31], [1], [33], and [29], which offer support for 9, 10, 11, 11, and 15 characteristics, correspondingly.

Conversely, our scheme supports all 18 security and privacy features. Using the 15 features provided in [29] as a basis, our scheme offers a 20% improvement in smart grid networks’ security posture.

6. Conclusions

The consumer consumption report and power adjustments data exchanged between SMs and SPs are exposed to many privacy and security threats. This is due to the utilization of insecure communication channels for the message communication procedures. Such attacks include ephemeral secret leakage, denial of service, eavesdropping, tampering, and forgery. To address this challenge, many security solutions have been developed recently. Nevertheless, the majority of these solutions has been shown to be inefficient or have some susceptibilities that render them inappropriate for smart meters. In this paper, a security protocol that is provably secure has been developed. It has also been demonstrated to be resilient against attacks such as privileged insider, de-synchronization, DoS, guessing, KSSTI, eavesdropping, EPSL, spoofing, physical capture, impersonation, replay, MitM, and forgery. In addition, it provides security functionalities such as anonymity, strong authentication, session key agreement, session key security, and untraceability. In terms of performance, it incurs the least computational costs and relatively lower storage and...
communication costs. Future work will feature the development of novel approaches that can further reduce the incurred storage and communication overheads.

Author Contributions: All the authors have contributed equally to this article. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors upon request.

Conflicts of Interest: The authors declare no conflicts of interest.

References


4. Numan, M.; Baig, M.F.; Yousif, M. Reliability evaluation of energy storage systems combined with other grid flexibility options: A review. J. Energy Storage 2023, 63, 107022. [CrossRef]


15. Gope, P.; Sikdar, B. A privacy-aware reconfigurable authenticated key exchange scheme for secure communication in smart grids. IEEE Trans. Smart Grid 2021, 12, 5335–5348. [CrossRef]


18. Gope, P.; Sikdar, B. Privacy-aware authenticated key agreement scheme for secure smart grid communication. IEEE Trans. Smart Grid 2018, 10, 3953–3962. [CrossRef]


25. Deng, L.; Gao, R. Certificateless two-party authenticated key agreement scheme for smart grid. *Inf. Sci.* 2021, 543, 143–156. [CrossRef]


42. Ikeda, K. Long-range quantum energy teleportation and distribution on a hyperbolic quantum network. *IET Quantum Commun.* 2023, 1–8. [CrossRef]


Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.