The Journal of Supercomputing (2024) 80:1668–1693
https://doi.org/10.1007/s11227-023-05535-2

Using a privacy-enhanced authentication process to secure IoT-based smart grid infrastructures

Samad Rostampour[1,4] · Nasour Bagheri[2,3] · Behnam Ghavami[4] · Ygal Bendavid[5] · Saru Kumari[6] · Honorio Martin[7] · Carmen Camara[8]

Accepted: 2 July 2023 / Published online: 20 July 2023

Abstract
Over the last decade, technological advances in smart grids have permitted the modernization of legacy electricity networks. As Internet of Things (IoT)-based smart grids are becoming an efficient response to managing changing electric demand, the heterogeneous network of equipment required to make these Cyber-Physical Systems a reality poses some security threats. This paper proposes a novel mutual authentication and key agreement scheme to ensure communications security and protect users' privacy in smart grid applications. In the proposed scheme (named EPSG), an elliptic curve cryptography (ECC) module and a physical unclonable function (PUF) are used simultaneously to provide acceptable confidentiality and integrity levels. The security analysis demonstrates that the EPSG has a robust security posture regarding transferred messages on the communication channel and physical attacks. In addition, EPSG is resistant to modeling attacks as one of the main vulnerabilities of PUF modules. Furthermore, by implementing the EPSG on an Arduino UNO microcontroller, a comparative performance evaluation (e.g., Time 156 ms, Communication cost 1408 bits, and Energy consumption 13.728 mJ) demonstrates the efficiency of the proposed EPSG.

Keywords IoT · Smart grid · Authentication · Key agreement · ECC · PUF

1 Introduction

The requirements for "sensing" and "responding" to electric demand along the transmission lines between utility providers and their customers have forced utilities (including sub-sectors of electrical, gas, and water) to convert "classic" distribution networks into bidirectional communication smart grids (SGs) and digitize a part of their business model [1]. These asset-intensive companies are turning to an industry 5.0 vision [2] and adopting Industrial Internet of Things (IIoT) technologies to

Extended author information available on the last page of the article connect and manage their assets throughout their life cycle. Smart meters (SM) are an excellent example to illustrate this transition from pure product to product-as-a-service (PaaS) in which case IoT devices are used as key data acquisition technologies in advanced metering infrastructure used for improving network efficiency and resilience while addressing energy sustainability concerns. Smart meters are tracked and traced from the production line to the installation, enabling connectivity with end-users, opening up new business model opportunities, and new business value creation. As SGs are becoming an efficient response for managing changing electric demand, ensuring an increased efficiency without compromising the data quality, the heterogeneous network of equipment required to make these systems a reality poses some threats to security [3].

Despite their many benefits, smart meters are vulnerable to various threats due to their reliance on decentralized communication systems. Attacks can lead to corrupted billing data (e.g., the well-known case of the Puerto Rican Electric Power Authority where estimated loss reaches hundreds of millions of dollars per year) as well as a risk of failing to provide reliable communication and to keep personal data confidential [1, 4]. In fact, from a market perspective, repeatedly reported attacks on smart meters have shown how security-related incidents can impact more seriously those critical infrastructures (e.g., power outage, blackout, geopolitics sabotage) [5, 6]. To address such concerns, secure and private communication infrastructure is highly required. As a result, an efficient authenticated protocol is critical for ensuring security in smart metering infrastructure. Note that the authentication process should be light enough to be implemented on resource-constrained smart meters over a smart grid.

Many prior authentication schemes face the challenge of not providing enough privacy to smart meters and being vulnerable to various attacks or high-computational overhead (Sect. 2). As a result, a more efficient authentication protocol is required for the practical use of smart metering infrastructure. This paper introduces a novel lightweight authenticated key agreement protocol for smart meters over a smart grid. The proposed scheme uses an Elliptic Curve Cryptography module and an SRAM physical unclonable function (PUF) to provide acceptable confidentiality and integrity levels. The proposed protocol offers various features, such as multi-factor mutual authentication and secret key establishment using two message trips. In addition, given that one of the main vulnerabilities of PUF-based solutions is machine learning (ML)-based modeling attacks, besides higher performance and security level, by combining ECC and PUF methods, the proposed protocol is resistant to such attacks.

The key contributions of this research are stated below:

- We have designed a novel lightweight ECC-based protocol for smart grid infrastructures. This protocol utilizes a PUF as a root of trust to prevent the cloning or tampering of Smart Meters. The combination of ECC and PUF with a low complexity provides the novelty of this article.
- A formal security evaluation of the proposed scheme using the widely accepted RoR model and its detailed ad hoc security analysis are presented. These security evaluations confirm an acceptable security level against different attacks especially modeling attacks as a root weak point of PUF modules.

Table 1 Comparison of related works

Reference	Communication cost	Time-consuming	Method	Approved attack
[7]	↑	↑	ECC	✓
[9]	↓	↑	ECC + MAC	✓
[11]	↑	↑	ECC	✓
[12]	↓	↑	ECC	✓
[13]	↑	↓	ECC	✓
[14]	↓	↓	ECC	✓
[15]	↑	↑	ECC	✓
[17]	↓	↓	PUF	✓
[19]	↑	↓	PUF	×
[20]	↓	↑	AES + IBE(ECC) + PUF	✓

- Experimental validation based on an Arduino UNO microcontroller reveals that the proposed scheme provides lower computation and communication overheads and energy consumption than the recent state-of-the-art of similar schemes.

The rest of the paper is organized as follows. The related works are presented in Sect. 2. In Sect. 3, we provide the required preliminaries (through this paper, we use the list of notations represented in Table 2). A detailed description of the proposed ECC-PUF-based protocol is provided in Sect. 4. The security analysis, including formal and heuristically evaluations against different attacks, is presented in Sect. 5. Performance analysis and comparison with the related works are given in Sect. 6. Finally, some conclusions are drawn in Sect. 7.

2 Related works

In recent years, much research has been conducted to present a secure and reliable smart metering communication infrastructure. For instance, to provide an acceptable integrity and security level in smart energy networks, Kumar et al. designed a lightweight authentication and key agreement scheme named LAKA [7]. In order to protect message confidentiality LAKA uses an ECC module and hash functions, and to provide message integrity, it utilizes a message authentication code (MAC) function. Therefore, the combination of various functions increases the complexity of the SM. In addition, as pointed out by Baghestani et al. [8], this protocol suffers from traceability attacks.

In Table 1, each protocol is evaluated based on communication overhead, time-consuming overhead, encryption method, and vulnerability. The downarrow (↓) and the uparrow (↑) show each feature's low and high amounts. In addition, the check-mark symbol (✓) indicates an approved attack already published, and the times symbol (×) means that the attack has not been published yet.

Table 2 List of used notations

Symbol	Description
E	Elliptic curve cryptography
$\mathbb{G}$	A finite prime group
P	Generator point of a large group G
q	A large prime number
SM	Smart meter
SP	Service provider
S/TA	A trusted server/authority
ID	The unique identifier
sk	ECC-based private key
PK	ECC-based public key
r	Random number
$Auth$	Authentication token
$H(\cdot)$	One-way hash functions
TS	Timestamp
$a \cdot P$	multiplying a point P by scalar a
$\|\|$	Concatenation
ΔT	An acceptable threshold for time
$A \overset{?}{=} B$	Comparison of $\mathcal{A}$ and B
SK	The shared session key
$\|X\|$	Cardinality of the set X
T_F	Computational run-time of the component F
$\perp$	Undefined symbol
Adv	Adversary advantage
PW_j	Password
ID_j	Identity

ECCAuth is another newly developed authentication protocol by Kumar et al. in smart grid applications [9] based on ECC cryptography. The authors stated that ECCAuth establishes a secure connection between an SG device and a Utility Center (UC) to transfer data confidentiality and protect user privacy. However, the communication cost of the ECCAuth is acceptable, and the authentication process and the computation operations are time-consuming. Moreover, the ECCAuth is vulnerable, and Yu et al. revealed ECCAuth's security flaws against masquerade, stolen device, and session key disclosure attacks [10]. They suggested a new lightweight protocol using hash and XOR functions to address the reported weaknesses. Another ECC-based authentication protocol was suggested by Wu et al. [11]. To protect the confidentiality of transferred messages, the SM uses an ECC module and encrypts data. However, their security analysis shows how the protocol's robustness in terms of functionality does not have an acceptable level. For example, the communication and computation overhead is high, and the SM needs lots of time to complete an authentication process. Recently Garg et al. [12] also designed another lightweight ECC-based authentication scheme for smart metering. The authors claimed that the proposed protocol is robust against various attacks and provides an acceptable computational cost. Our evaluation shows that their protocol is vulnerable to impersonation and traceability attacks. The computational cost was not calculated correctly, and the authors underestimated their results. References [13–15] are other ECC-based protocols for smart meters that have tried to present a secure infrastructure but in terms of efficiency have some issues that will be discussed in greater detail in Sect. 6.

PUF-based schemes are another interesting technique applied in smart metering/grid applications because physical properties can provide valuable information to detect tampering [16]. There are several numbers of PUF-based protocols oriented to SG/SM. A good precedent is the PUF-based privacy-aware authenticated key agreement scheme presented by Gope and Sikdar to improve the security level of smart meters [17]. Given that cyber-attacks on meters, such as tampering data, can affect decision-making processes related to demand and supply management, the authors designed a solution to increase channel confidentiality among meters and UCs and ensure physical security. Although the authors tried to present a secure solution with minimum requirements, Baeken et al. proved the vulnerabilities of this method [18]. Thus, this protocol could not meet all security requirements. Furthermore, the Gope-Sikdar protocol, in addition to PUF, uses a hash function as the primary source of security, indicating that it is a symmetric protocol that is vulnerable to key compromise impersonation attacks. Moreover, in their protocol, the secret key is XORed with a temporal value before being transferred over the public channel (i.e., $np * = n_p \oplus K$), and K is later used as the primary source of authentication. Given K, this protocol allows for any other desired attacks, such as impersonation, de-synchronization, and so on. As a result, this protocol is vulnerable to the Known session-specific temporary information attack.

Mustapa et al. [19] devised a ring oscillator PUF-based (ROPUF) scheme to secure information exchange in advanced metering infrastructures. The main objective of this solution is to design a secure and resistant authentication protocol for advanced metering infrastructure. In ROPUF architecture, a secure channel between the SM and UC to exchange the parity bits and the challenges has been considered. Nevertheless, as also has been pointed out by [20], the authors have not mentioned how this channel works and how the SM communicates in this channel. Definitely, this channel burdens overhead on the SM to provide cipher texts, and this load has not been considered in the computation cost of the protocol. On the other hand, the proposed protocol was vulnerable to impersonation and tracing attacks if we assume that the data are exchanged via a public channel. Additionally, because the proposed protocol does not contain any cryptographic primitives, advanced attacks like insider attacks are also applicable against it.

Recently, Harishma et al. proposed a mutual authenticated key-exchange scheme for smart meters [20]. Using the advanced encryption standard (AES) method, SHA-2, identity-based encryption (which could use ECC encryption), and PUF functions, the authors designed a scheme between the SM and UC to secure communication and protect data confidentiality. They tested the protocols on commercial-off-the-shelf products and presented the experimental results. However, this scheme employs identity-based encryption (IBE) to manage credentials, for which the most efficient schemes currently available are based on bilinear pairings on elliptic curves, such as the Weil or Tate pairings and previously published non-pairing-based schemes are generally inefficient in encryption, decryption, key generation, ciphertext size, or key size [21, 22]. Given that the scheme is designed for constrained devices and each authentication process needs to run various encryption methods, the complexity of the protocol is high, and it is time-consuming, for instance, the reported time for authenticated key-exchange protocol on the smart meter setup is 525 and 360 milliseconds, respectively, for the meter and server. Further to that, [23], the protocol was shown to be vulnerable to a spoofing attack by spoofing the server and fooling the meter to compromise the authentication claim as well as the key-establishment claim of the authentication protocol. Furthermore, the meter does not contribute to the protocol's freshness during the authentication phase of the proposed protocol. As a result, using [24], it is possible to impersonate the server.

As shown in Table 1, in order to provide confidentiality plus the integrity of transferred messages in smart grids, many protocols have been published. Unfortunately, a secure, lightweight, and energy-friendly protocol is still needed.

3 Preliminaries

3.1 System model

As shown in Fig. 1, we consider an IoT-based smart grid system model including edge devices, gateways/service providers, and servers. Among the different participants in this complex network, we focus on the edge layer of the SG and consider two clients participating in sharing a session key (i.e., a smart meter (SM) and a service provider (SP)- two red boxes).

Due to the different roles of these entities in an SG, we assume that the trusted authority (TA) assigns each registered SM_i to the specific cluster which is connected to a parent SP_j (in line with previous works, i.e., [11]). Each SP provides services for several SM, e.g., it collects their data and transmits it to a server over a secure channel, while any registered SM_i can only communicate with its parent SP_j.

An overview of the connections between the protocol entities, i.e., TA, SMs, and SP, is depicted in Fig. 1. Therefore, this research aims to provide a secure authentication process between a smart meter and a service provider via communication layers to protect confidential data and preserve users' privacy.

3.2 Physical unclonable function

The foundation of trust (i.e., the root of trust) is based on the device identity and the root key (the main secret on which all trust and security in a system are built). Therefore, how this key is provisioned and handled during the lifecycle of the device is critical for the security of the entire system. In this context, physical unclonable functions (PUFs) have proven as a scalable solution for key provisioning from silicon to cloud (e.g., IoT [25, 26]). PUFs exploit intrinsic manufacturing variability of

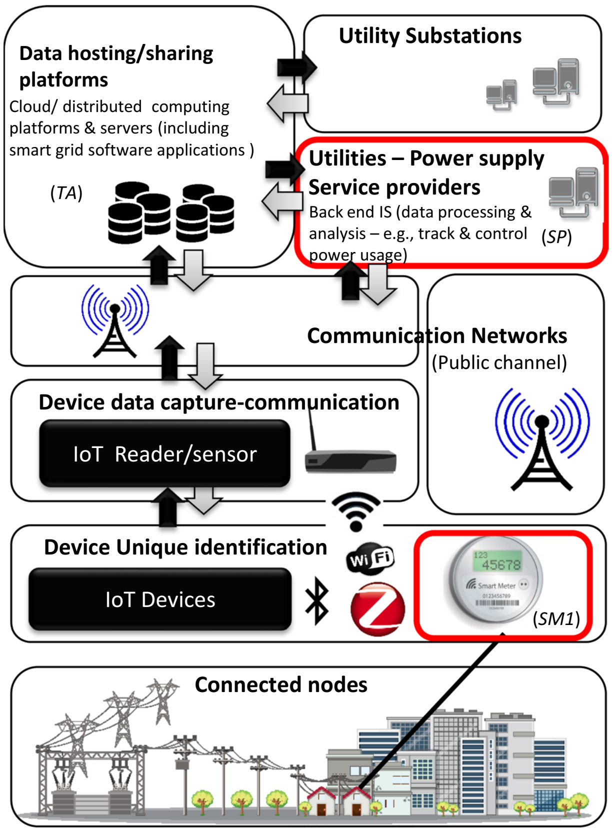

Data hosting/sharing platforms
Cloud/ distributed computing platforms & servers (including smart grid software applications)
(TA)
Utility Substations
Utilities – Power supply Service providers
Back end IS (data processing & analysis – e.g., track & control power usage)
(SP)
Communication Networks
(Public channel)
Device data capture-communication
IoT Reader/sensor
Device Unique identification
IoT Devices
(SM1)
Connected nodes

Fig. 1 Architecture of an IoT-based smart grid

the silicon fabrication process to generate a signature unique to every single physical device [25]. The ability of PUFs to generate unique challenge-response pairs (CRPs) for each device can be leveraged as an authentication mechanism to detect tamper, impersonation, or substitution of such devices and play the role of hardware-based root of trust of devices.

3.3 Adversary model

Through our analysis, we consider a probabilistic polynomial time (PPT) active adversary with full access to the transferred messages over the public channels among the protocol's parties. Thus, the adversary can eavesdrop on the transferred messages, manipulate them, store and replay them later or try to impersonate any of the protocol's parties. In addition, the adversary has access to public parameters of the protocol, such as the public keys of the participants. This adversary model follows "Dolev-Yao (DY)" adversary model [27]. On the other hand, given that in reality, the adversary can access the smart meters and read their memory; in the case of forward secrecy, we assume that the attacker can compromise a target smart meter or a service provider in offline mode and reveal the stored information in the non-volatile memory, including the secret key. However, $\mathcal{A}$ has no access to the internal values in an active session. Therefore, the adversary is restricted to access the temporary values of a legitimate session.

3.4 Semantic security in the real-or-random model

To share a session key SK in a 3-party authenticated key agreement scheme, instances use their long-term secrets. A protocol's instance could be either a client $N \in \mathcal{N}$ or a trusted server $S \in \mathcal{S}$, and a client N could be either honest or malicious. Any client N holds a long-term key sk_N and S holds a vector $sk_S = \langle sk_S[N] \rangle_{N \in \mathcal{N}}$, includes an entry for each client N, where $sk_S[U]$ is a transformation of sk_N. An adversary $\mathcal{A}$ has access to sk_N if N is a malicious client. N_i and N_j are called *partner* if they share a same session data. To distinguish the target protocol $\mathcal{P}$ from an ideal one, $\mathcal{A}$ can run the following query types [28]:

- Execute: models a passive adversary $\mathcal{A}$, which eavesdrops on the channel and obtains the read access to the transferred messages.
- Send (S): models an active adversary that may intercept a message and then either modify it, create a new one, or simply forward it to an instance.
- Reveal (N_i): for which the output could be the session key held by the instance N_i.
- Test (N_i): If no session key for instance N_i is defined or if a Reveal query was asked to either N_i or to its partner, then returns the undefined symbol $\perp$. Otherwise, returns the session key for instance N_i if $b = 1$, or a random key with the same size if $b = 0$.

b is a predefined bit at the beginning of the experiment and should be guessed by A using the above queries to win the game. Ending the game, A releases the choice b_0 as the guessed value of b, and this game defines the semantic security in the real-or-random (RoR). The adversary's advantage to win this game, $Adv^{RoR}_{\mathcal{D},\mathcal{P}}(t, R)$, is defined as follows:

$$Adv^{RoR}_{\mathcal{D},\mathcal{P}}(t, R) = \big((Pr(A \rightarrow b_0 = 1 : b = 1) - (Pr(A \rightarrow b_0 = 1 : b = 0))\big)$$

$\mathcal{P}$ offers RoR semantic security if:

$$Adv^{RoR}_{\mathcal{D},\mathcal{P}(t,R)} < \varepsilon(\cdot)$$

and $\varepsilon(\cdot)$ being some negligible function. The maximum advantage is taken over all A with time complexity at most t and using resources at most R.

4 Proposed protocol

We assume that in an SG, a smart meter SM_i aims to communicate with a service provider SP_j to transfer the collected data or extract the processed data securely. However, various types of encryption models [29] are utilized in IoT, the ECC (as one of the main candidates for IoT encryption) has been chosen in this study because it needs low data requirement that causes low-power and fast processor resources. Therefore, the proposed protocol uses the ECC and the PUF to provide the desired security level in an SG named EPSG, which stands for ECC and PUF-based protocol for SG. The EPSG includes three phases, i.e., initialization, registration, and login and key agreement phases. We also consider a password revocation phase to allow the legitimate user/administrator to update his password. The used notations are listed in Table 2.

4.1 Initialization phase

In this phase of the EPSG, each device is equipped with a $PUF(\cdot)$. Besides, the TA releases the system parameters $\{E_q(c,d), q, P, h(\cdot)\}$ and its public key $PK_{TA} = sk_{TA} \cdot P$ and keeps sk_{TA} secret.

4.2 Registration phase

The registration phase of EPSG occurs over a secure channel. In this phase, all entities are registered to the TA, but they receive different information from the TA depending on their role (Fig. 2).

4.2.1 Registration phase of service provider

As it is depicted in Fig. 3, the process of the registration phase for a service provider, e.g., SP_j, is as follows:

1. The SP_j's user/administrator securely chooses its password PW_j and identity ID_j. These paramaters are used to run a smart meter as username and password. The administrator inters SP_j and ID_j through the available terminal and also SP_j generates a random number $m_j \in Z_q^*$, computes $PID_j = PUF(h(PW_j \| ID_j))$, and sends $PID'_j = m_j \cdot PID_j$ and ID_j to the TA.
2. Upon receiving the message, the TA generates an integer $s'_j \in Z_q^*$, computes $sk'_j = s'_j + PID'_j \cdot sk_{TA}$ and $PK'_j = sk'_j \cdot P$ and sends $\{sk'_j, PK'_j\}$ to the SP_j.
3. SP_j stores $(sk_j = m_j^{-1} \cdot sk'j, PK_j = m_j^{-1} \cdot PK'_j)$ as its secret key and public key, respectively, where $sk_j = m_j^{-1} \cdot s'_j + PID_j \cdot sk_{TA}$. It also stores

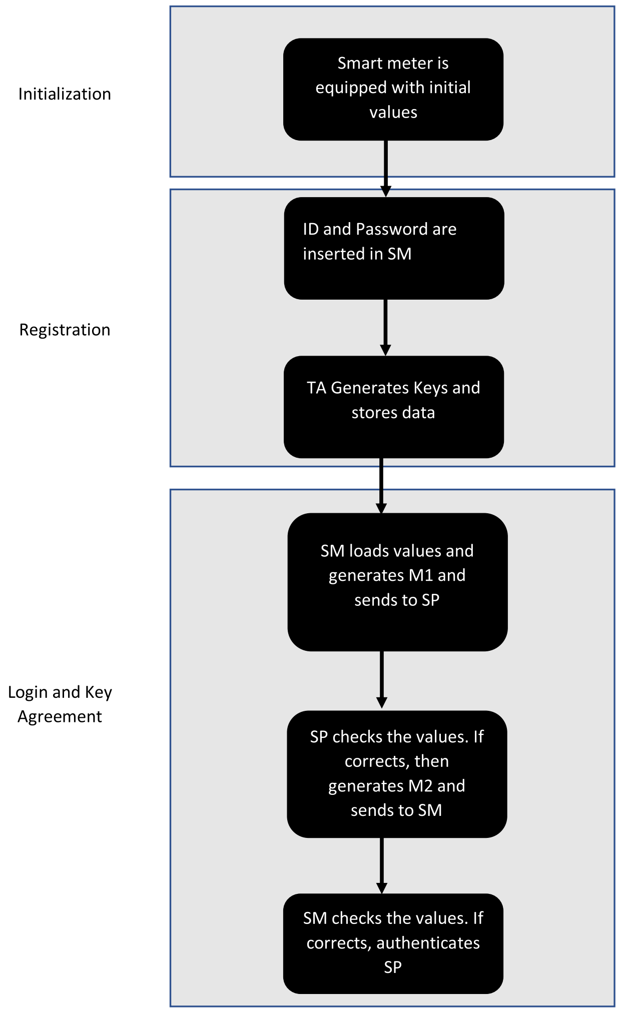

Initialization
Smart meter is equipped with initial values
Registration
ID and Password are inserted in SM
TA Generates Keys and stores data
Login and Key Agreement
SM loads values and generates M1 and sends to SP
SP checks the values. If corrects, then generates M2 and sends to SM
SM checks the values. If corrects, authenticates SP

Fig. 2 Flowchart of the proposed solution

$X_j = h(PUF(h(PW_j \| ID_j)) \| sk_j \| PK_j)$. All values are stored in the non-volatile memory. Moreover, the SP_j sends PK_j to the TA.

4. The TA stores (ID_j, PK_j) in its database.

4.2.2 Registration phase of smart meter

The process of the registration phase for an SM_i is as follows:

1. The SM_i user/administrator securely chooses its password PW_i and identity ID_i and inters them through the available terminal and also SM_i generates a random

SP_j	$Message$	TA
PW_j and ID_j is inserted, generates $m_j \in Z_q^*$, computes $PID_j = PUF(h(PW_j \| ID_j))$ and $PID'_j = m_j.PID_j$		
	$\xrightarrow{PID'_j, ID_j}$	Generates $s'_j \in Z_q^*$, computes $sk'_j = s'_j + PID'_j.sk_{TA}$, $PK'_j = s'_j.P$
Computes $(sk_j = m_j^{-1}.sk'_j, PK_j = m_j^{-1}.PK'_j)$ and $X_j = h(PUF(h(PW_j \| ID_j)) \| sk_j)$ and stores X_j, (sk_j, PK_j) and PK_{TA}	$\xleftarrow{\{(sk'_j, PK'_j), PK_{TA}\}}$	
	$\xrightarrow{PK_j}$	Stores (ID_j, PK_j)

Fig. 3 Registration phase between SP_j and TA over secure channel

number $m_i \in Z_q^*$, computes $PID_i = PUF(h(PW_i \| ID_i))$ and sends $PID'_i = m_i \cdot PID_i$ and ID_i to the TA.

2. Once received the message, the TA generates an integer $s'_i \in Z_q^*$, computes $sk'_i = s'_i + PID'_i \cdot sk_{TA}$ and $PK'_i = sk'_i \cdot P$ and sends $\{(sk'_i, PK'_i), PK_j\}$ to the SM_i.
3. The SM_i computes $(sk_i = m_i^{-1} \cdot sk'_i, PK_i = m^{-1} \cdot PK'_i)$ as its secret key and public key, respectively, and sends PK_i to the TA, where $sk_i = m_i^{-1} \cdot s'_i + PID_i \cdot sk_{TA}$. The smart meter SM_i also stores (PK_j) as the data related to the authorized SP_j that the SM_i can communicate with. In addition, the SM_i stores $X_i = h(PUF(h(PW_i \| ID_i)) \| sk_i \| PK_j)$. All values are stored in the non-volatile memory.
4. The TA stores (ID_i, PK_i) in its database and sends them through a secure channel to the target SP_j; then, SP_j stores them in its database securely.

4.3 Login and key agreement phase

On startup of each device, we assume that the user/administrator inputs the username and password, and they are verified, and $h(PW_{i/j} \| ID_{i/j})$ is stored in a volatile memory accordingly. For example, on the power-on phase before initiating a login and key agreement process between SM_i and SP_j, we assume that the SP_j is turned on by a user/administrator. The user/administrator of the SP_j enters ID_j and PW_j. The SP_j checks $h(PUF(h(PW_j \| ID_j)) \| sk_j \| PK_j) \overset{?}{=} X_j$ and, if the verification passes, $h(PW_j \| ID_j)$ is stored in a volatile memory. Similarly, the SM_i verifies the inserted ID_i and PW_i based on its X_i, and if they are valid, it stores $h(PW_i \| ID_i)$. The login and key agreement phase of EPSG between SM_i and SP_j proceeded as follows, also depicted in Fig. 4:

SM_i	$Message$	SP_j
The stored values are loaded, checks $h(PUF(h(PW_i\|\|ID_i))\|\|sk_i\|\|PID_j\|\|PK_j) \overset{?}{=} X_i$, generates TS_i and $r_i \in Z_q^*$, computes $B_1 = r_i \cdot sk_i \cdot PK_j$, $B_2 = r_i \cdot PK_i$, $B_3 = h^2(B_1) \oplus PK_i$ and $B_4 = h(B_1\|\|B_2\|\|B_3\|\|TS_i)$		
	$\xrightarrow{M_1=\{B_2,B_3,B_4,TS_i\}}$	Checks TS_i, computes $B_1^* = sk_j \cdot B_2$, extracts $PK_i^* = h^2(B_1^*) \oplus B_3$, verifies PK_i^* is legitimate and also $B_4 \overset{?}{=} h(B_1^*\|\|B_2\|\|B_3\|\|TS_i)$ to authenticate SM_i, generates TS_j and $r_j \in Z_q^*$, computes $C_1 = r_j \cdot PK_j$, $C_2 = r_j \cdot B_1^*$, and $C_3 = h(C_1\|\|C_2\|\|TS_j)$
Computes $C_2^* = r_i \cdot sk_i \cdot C_1$, verifies whether $C_3 \overset{?}{=} h(C_1\|\|C_2^*\|\|TS_j)$ to authenticate SP_j	$\xleftarrow{M_2=\{C_1,C_3,TS_j\}}$	
		$SK_{ij} = h(C_2\|\|PK_i\|\|PK_j\|\|TS_j)$
$SK_{ij} = h(C_2\|\|PK_i\|\|PK_j\|\|TS_j)$		

Fig. 4 Login and key agreement phase of EPSG, over an insecure channel

1. The SM_i loads the stored parameters $h(PW_i\|ID_i), sk_i, PK_j$. Next, it generates its current timestamp TS_i and a random value $r_i \in Z_q^*$, calculates $B_1 = r_i \cdot sk_i \cdot PK_j$, $B_2 = r_i \cdot PK_i$, $B_3 = h^2(B_1) \oplus PK_i$ and $B_4 = h(B_1\|B_2\|B_3\|TS_i)$, and sends $M_1 = \{B_2, B_3, B_4, TS_i\}$ to the SP_j.
2. Once received M_1, the SP_j checks the timestamp TS_i, if it is invalid, the session is aborted. Next, it computes $B_1^* = sk_j \cdot B_2$ computes $B_1^* = sk_j \cdot B_2$, extracts $PK_i^* = h^2(B_1^*) \oplus B_3$, verifies PK_i^* is legitimate and also $B_4 \overset{?}{=} h(B_1^*\|B_2\|B_3\|TS_i)$ to authenticate the SM_i. Assuming that the SM_i is authenticated, the SP_j generates its current timestamp TS_j and a random value $r_j \in Z_q^*$, calculates $C_1 = r_j \cdot PK_j$, $C_2 = r_j \cdot B_1^*$ and $C_3 = h(C_1\|C_2\|TS_j)$, and sends $M_2 = \{C_1, C_3, TS_j\}$ to the SM_i.
3. The SM_i calculates $C_2^* = r_i \cdot sk_i \cdot C_1$ and checks whether $C_3 \overset{?}{=} h(C_1\|C_2^*\|TS_j)$ to authenticate the SP_j.
4. The shared session key is computed as $SK_{ij} = h(C_2\|PK_i\|PK_j\|TS_j)$.

4.4 Password revocation phase

The user/administrator may desire to change his/her password after a while for security reasons. To enable the legitimate user/administrator of the SP_j to update its password, the considered steps are as follows:

1. The user/administrator of the SP_j enters ID_j and PW_j and the password revocation command $PassRevoc$.
2. The SP_j checks $h(PUF(h(PW_j \| ID_j)) \| sk_j \| PK_j) \overset{?}{=} X_j$ and, if the verification passed requests for new password PW_j^{new} two times to make sure, they are matching and the new password has been entered correctly. Then, it computes $h(PUF(h(PW_j^{new} \| ID_j) \| sk_j \| PK_j))$ and replaces the current X_j in its memory for the later authentications.
3. The SP_j informs the user/administrator that the password has been changed successfully.

Similarly, to enable the legitimate user/administrator of the smart meter SM_i to update its password, the procedure is as follows:

1. The user/administrator of SM_i enters ID_i and PW_i and the password revocation command $PassRevoc$.
2. The SM_i checks $h(PUF(h(PW_i \| ID_i)) \| sk_i \| PID_i \| PK_j) \overset{?}{=} X_i$ and if the verification passed requests for new password PW_i^{new} two times to make sure that they are matching and the new password has been entered correctly, then, SM_i replaces X_i by $h(PUF(h(PW_i^{new} \| ID_i) \| sk_i \| PID_i \| PK_j))$ and stores it in its memory for the later authentications.

If it is required to change the identity or the secret key, then, SM_i/SP_j should be re-registered. Because it has been registered in the TA based on the current $(ID_i, PK_i)/(ID_j, PK_j)$.

5 Security analysis of EPSG

5.1 Informal security analysis

In this section, we informally prove that the EPSG provides the desired security level against attacks in the context.

5.1.1 Mutual authentication

In the proposed protocol, the SP_j receives $M_1 = \{B_2, B_3, B_4, TS_i\}$, calculates $B_1^* = sk_j \cdot B_2$ and checks whether $B_4 \overset{?}{=} h(B_1^* \| B_2 \| B_3 \| TS_i)$ to authenticate the SM_i, where $B_1 = r_i \cdot sk_i \cdot PK_j$, $B_2 = r_i \cdot PK_i$, $sk_i = s_i + sk_{TA} \cdot PID_i$ and $PK_i = sk_i \cdot P$. Following the calculations below, the SP_j successfully authenticates the legitimate SM_i:

$$\begin{aligned} sk_j \cdot B_2 =& sk_j \cdot r_i \cdot PK_i \\ =& r_i \cdot sk_i \cdot PK_j \\ =& B_1 \end{aligned}$$

On the other hand, the SM_i receives $M_2 = \{C_1, C_3, TS_j\}$, calculates $C_2^* = r_i \cdot sk_i \cdot C_1$, and verifies whether $C_3 \overset{?}{=} h(C_1 \| C_2^* \| ID_i \| (PK_i \oplus PK_j))$ to authenticate the SP_j. Following a similar argument, the SM_i also successfully authenticates the SP_j. Therefore, the proposed protocol provides a mutual authentication process.

5.1.2 Session key agreement

The session key is calculated as $SK_{ij} = h(C_2 \| PK_i \| PK_j \| TS_j)$. Both the SP_j and the SM_i have access to PK_i, PK_j, and TS_j. In addition, based on the argument provided in Sect. 5.1.1, the SM_i can successfully calculate C_2.

5.1.3 Replay attack

The adversary needs to store what is being exchanged on a channel to conduct a replay attack. Then, the attacker sends these values to the smart meter or the SP as a valid response without knowing the content of the message itself to execute the replay attack finally. In the proposed protocol, two parameters exist to avoid the replay attack. The first one is the timestamp. Any session is refreshed by the timestamps TS_i and TS_j, verified by the SM_i and the SP_j. In the first step, each device checks the TS and continues the process. Thus, using old messages are not applicable. In addition, utilizing a random number in the messages' content allows the receiver to be sure that the received message is based on the last generated random number. Therefore, the proposed protocol is robust against replay attacks.

5.1.4 Impersonation attack

To impersonate the SM_i, the adversary should either do a replay attack (which is not feasible) or generate a valid M_1 that consists of B_2, B_3, and B_4. Given that B_1, as a secret and important value, is not sent on the public channel and embedded in other parameters as the hash function's output, the attacker cannot create it. In addition, he has no access to sk_i and PID_i. Therefore, there is not enough information for impersonating a smart meter. The same argument can be deduced for the impersonation of the SP_j. Thus, it can be claimed that the EPSG is not vulnerable to impersonation attacks.

5.1.5 Traceability and anonymity

By implementing a traceability attack, an adversary can detect a member of the system in different situations. For achieving this goal, he needs a permanent and instant value in the transferred messages. In the login and key agreement phase, firstly, the SM_i generates a random number (i.e., r_i) and uses it in the $B1$. Then, the SP_j also generates r_j randomly. Therefore excluding TS_i and TS_j, the rest of the exchanged parameters are encrypted values or the output of the one-way hash function, and fresh nonce values randomize them. As a result, it is not feasible to trace the SM_i or the SP_j or compromise their anonymity.

5.1.6 Secret disclosure attack

This type of attack occurs when sensitive or confidential information such as encryption keys or identification numbers cannot be protected against unauthorized users. Any sensitive data in the EPSG are transferred in the form of an encrypted value or as the output of the one-way hash function. Hence, A cannot perform the secret disclosure attack, excluding that, for example, he can break with ECDLP or EC-CDHP.

5.1.7 Forward secrecy with compromised edge device

Given that smart meters are distributed over the field, it could be possible for the adversary to compromise a smart meter SM_i and read the stored data in the non-volatile memory, i.e., $X_i = h(PUF(h(PW_i \| ID_i)) \| sk_i \| PK_j)$, sk_i, PK_i and PK_j. However, the SM_i is equipped with a PUF function, and the adversary cannot clone it. In addition, each session key is randomized by ephemeral values contributed by SM_i and SP_j. Hence, even long-term key leakages do not compromise the previous session keys.

5.1.8 Insider attack

An insider attacker in the TA—with access to its memory and monitoring the transferred messages over the secure channel, in the registration phase—can access to sk'_i, sk'_j, PK'_i, PK'_j, PK_i, PK_j, PID_i, and PID_j. None of this information helps the insider adversary to have access to the password or trace the SM_i or the SP_j. The reason is the fact that SP_j (resp. SM_i) computes $PID_j = PUF(h(PW_j \| ID_j))$ (resp. $PID_i = PUF(h(PW_i \| ID_i)))$ and sends $PID'_j = m_j \cdot PID_j$ and ID_j (resp. $PID'_i = m_i \cdot PID_i$ and ID_i) to the TA. Hence, the password is protected by $PUF(\cdot)$ and m_j (resp. m_i) which is not known by the adversary. Hence, the insider has no advantage to recover the devices' user/administrator password, and the proposed protocol is also secure against offline password guessing.

5.1.9 Password guessing

Among the transferred messages, the values of B_2, B_3, and B_4 are randomized by r_i which is a fresh nonce contributed by the SM_i and C_1 and C_3 are randomized by r_i and r_j, which r_j is a fresh nonce contributed by the SP_j. These messages are produced by the one-way hash function or the ECC multiplication. Hence, it is not feasible for the adversary to guess the password of the SM_i or the SP_j. In addition, we have already proved that an insider adversary has no chance to guess the passwords.

5.1.10 Known session-specific temporary information attack

In a known session-specific temporary information attack (KSTIA), it is assumed the ephemeral values are leaked, and the target is to compromise the session key or the long-term keys. In EPSG, the ephemeral values used in each session are m_i and m_j and the session key is computed as $SK_{ij} = h(C_2 \| PK_i \| PK_j \| TS_j)$, where $C_2 = r_i \cdot r_j \cdot sk_i \cdot PK_j$. Since the session key is a factor of both ephemeral and long-term secret values, the adversary cannot compute the session key just given the ephemeral secret values. It shows that EPSG is secure against KSTIA.

5.1.11 Key compromise impersonation resilience

In a key-exchange protocol with two parties $\mathcal{X}$ and $\mathcal{Y}$, in a key compromise impersonation attack (KCI), it is assumed the adversary knows all secret parameters of $\mathcal{X}$; then, it should not be possible to impersonate $\mathcal{Y}$ toward $\mathcal{X}$ if the protocol is KCI resilience. The main target of the KCI attack is to play the role of a man-in-the-middle attack. In the proposed protocol, $sk_i \cdot sk_j \cdot P$ is directly used in the computation of $B_4 = h(B_1 \| B_2 \| B_3 \| TS_i)$ and $C_3 = h(C_1 \| C_2 \| TS_j)$. More precisely, $B_1 = r_i \cdot sk_i \cdot PK_j$, $B_2 = r_i \cdot PK_i$, $C_1 = r_j \cdot PK_j$, $C_2 = r_j \cdot B_1^*$ and B_2 and C_1 are sent over the public channel. Hence, assuming that the adversary has access to the SM_i's secret parameters, for impersonating SP_j, the adversary should be able to recompute $B_1 = r_i \cdot sk_i \cdot sk_j \cdot P$, given $r_i \cdot sk_i \cdot P$ which is not feasible without the knowledge of sk_j. On the other hand, to impersonate SM_i toward SP_j and share the session key, the adversary should compute $C_2 = r_i \cdot r_j \cdot sk_i \cdot PK_j$ given $C_1 = r_j \cdot PK_j$, which requires the knowledge of sk_i. It shows that EPSG provides KCI resilience.

5.1.12 Modeling attacks against the PUF

Modeling attacks based on Machine Learning (ML) techniques [30] or approximation techniques [31] have emerged as one of the most important threats against PUF technology. In this kind of attack, the adversary collects a subset of CRPs (typically hundreds of them) and uses the aforementioned techniques to build a mathematical model to predict PUF responses. In this respect, several countermeasures at the PUF architecture level have been presented in the literature [32, 33]. In addition, the integration of countermeasures at different levels (including the protocol level) is highly recommended. In this vein, lockdown and obfuscation mechanisms are popular techniques that could be easily integrable into the proposed protocol. On the one hand, modeling attacks need hundreds/thousands of CRPs (less than 10% of the total CRPs [34]) in order to perform a successful attack. Lockdown mechanisms like the one presented in [34] try to limit the number of CRPs that an attacker can observe. Another example of a lockdown countermeasure integrated into a protocol is presented in [35]. On the other hand, randomization methods are used to preserve the mappings between challenges and responses reducing the correlation between them.

Table 3 Security comparison of EPSG with other schemes

Attacks	[11]	[12]	[13]	[14]	[15]	[17]	EPSG
Impersonation	✓	✓	✓	✓	×	✓	✓
Replay	✓	✓	✓	✓	✓	✓	✓
Traceability	×	×	✓	✓	×	×	✓
Secret conflict of interest	✓	✓	✓	✓	×	×	✓
Relay	×	✓	×	×	✓	✓	✓

Resistant: ✓

Non-resistant: ×

References [36] or [37] are good examples of these obfuscation mechanisms applied at different layers.

Nevertheless, the way the PUF is used in this authentication protocol makes it robust against modeling attacks. As stated before, several CRPs are required to carry out this kind of attack. In a realistic scenario, an attacker has access to the SM/SP only during the login and key agreement phase. If we assume that the attacker knows PK_j (public) and somehow he/she obtained sk_j, then, the attacker could try different pairs of PW_j and ID_j. In this scenario, the attacker will have access to PUF material only in the case that the correct pair is introduced.

Thus, as shown in Table 3, in terms of informal security analysis, the EPSG, in comparison with other protocols, has a better security level.

5.2 Formal security analysis in semantic security model

In any security protocol, several messages are exchanged among entities. On the other hand, the adversary passively or actively tries to maximize its advantage to attack the protocol and contradict a security requirement. For example, if the transferred messages over a session could be linked to a protocol participant or the transferred messages over two sessions, in which a specific client is participating, could be linked to each other, then the protocol suffers from traceability. Thus, a promising approach to ensure different security aspects could prove the indistinguishability of transferred messages over protocol from random values, excluding the messages that do not reflect any participant-connected information, e.g., timestamps. Such property could directly satisfy security against various attacks, including traceability, impersonation, secret disclosure, etc. The RoR model is an approach to show that the transferred messages over different sessions are not distinguishable from random values, considering a probabilistic polynomial time (PPT) for the adversary. Therefore, in this subsection, we formally evaluate the security of the EPSG in the RoR, by determining the adversary's advantage in distinguishing the EPSG from the random world RW.

Theorem 1 *Let q_{exe}, q_{send} and q_{test}, respectively, represent the number of queries to* Execute, Send *and* Test *oracles on the EPSG/RW, then*:

$$\begin{aligned} &Adv^{RoR}_{\mathcal{D},EPSG}(t, q_{exe};q_{test};q_{send}) \\ &\quad - Adv^{RoR}_{\mathcal{D},RW}(t, q_{exe};q_{test};q_{send}) \\ &\quad \leq 5 \cdot q \cdot \varepsilon_{ECC} + 7 \cdot q \cdot \varepsilon_h + q \cdot \varepsilon_{PUF} \end{aligned}$$

where ε_{ECC} denotes the maximum advantage of solving the ECDLP or the EC-CDHP by the adversary on each query, ε_H denotes the maximum advantage of contradicting collision resistance property of $h(\cdot)$, and ε_{PUF} denotes the maximum advantage of distinguishing the output of $PUF(\cdot)$ from a random sequence and $q = q_{exe} + q_{test} + q_{send}$.

Proof Let clients the SM_i and the N_j are communicating through S to share a session key and let A be an adversary against the semantic security of the EPSG in the RoR model. A game-based approach is used to prove the above theorem by defining a series of games G, starting from the random world RW and ending in the real-world EPSG. For each game G_n, we define an event $Adv^{RoR-G_n}_{\mathcal{D},\mathcal{P}}(t, R)$ corresponding to the adversary's advantage to correctly guess the hidden bit b involved in the Test queries. In each game, a simulator tries to behave as similarly to the real work as it can.

Game G_0. It defines RW and $Adv^{RoR-G0}_{\mathcal{D},RW}(t, R) = 0$

Game G_1. Compared to G_0, in this game, any instance follows the structure of the transferred messages in the EPSG. However, all sensitive messages are selected completely randomly. It is clear $Adv^{RoR-G0}_{\mathcal{D},RW}(t, R) - Adv^{RoR-G1}_{\mathcal{D},RW}(t, R) = 0$.

Game G_2. This game is identical to G_1 with the exception that the timestamps are not random anymore and follow the expected structure. Given that the session key and also other messages that are connected with the instances are still generated randomly, and the simulator can easily adapt timestamps, this modification has no impact on the adversary's advantage to guess the hidden bit and $Adv^{RoR-G_2}_{\mathcal{D},RW}(t, R) - Adv^{RoR-G_1}_{\mathcal{D},RW}(t, R) = 0$.

Game G_3. In this game, B_4 and C_3 are calculated using $h(\cdot)$, but the input values are still random with specific lengths. Hence, the adversary's advantage comes from collisions on $h(\cdot)$ and for $q = q_{exe} + q_{send} + q_{test}$:

$$Adv^{RoR-G_3}_{\mathcal{D},RW}(t, R) \leq Adv^{RoR-G_2}_{\mathcal{D},RW}(t, R) + 2 \cdot q \cdot \varepsilon_h \tag{1}$$

Game G_4. In this game, set $B_3 = R_1 \oplus PK_i$ where R_1 is a random sequence. It is clear this modification does not affect the adversary's advantage and:

$$Adv^{RoR-G_4}_{\mathcal{D},RW}(t, R) = Adv^{RoR-G_3}_{\mathcal{D},RW}(t, R) \tag{2}$$

Game G_5. In this game, set $R_1 = h^2(B_1)$ and $B_1 = R_2$. The adversary can win this game if he can distinguish $h(\cdot)$ from R_1. Hence:

$$Adv^{RoR-G_5}_{\mathcal{D},RW}(t, R) = Adv^{RoR-G_4}_{\mathcal{D},RW}(t, R) + q \cdot \varepsilon_h \tag{3}$$

Game G_6. In this game, B_2 and C_1 are, respectively, calculated as $B_2 = r_i \cdot PK_i$ and $C_1 = r_j \cdot PK_j$. Given that r_i and r_j are session-dependent random values, the adversary wins this game if he can solve the ECDLP or the EC-CDHP. Hence, his advantage is determined as follows:

$$Adv_{\mathcal{D},RW}^{RoR-G_6}(t,R) \leq Adv_{\mathcal{D},RW}^{RoR-G_5}(t,R) + 2 \cdot q \cdot \varepsilon_{ECC} \tag{4}$$

Game G_7. This game is identical to G_6, exclude that $B_1 = r_i \cdot sk_i \cdot PK_j$. Given that, B_1 is masked by $h(\cdot)$, the adversary's advantage in this game at most comes from solving the ECDLP or the EC-CDHP. The adversary wins this game if he can solve the ECDLP or the EC-CDHP:

$$Adv_{\mathcal{D},RW}^{RoR-G_7}(t,R) \leq Adv_{\mathcal{D},RW}^{RoR-G_6}(t,R) + q \cdot \varepsilon_{ECC} \tag{5}$$

Game G_8. This game is identical to G_7, exclude that $C_2 = r_j \cdot B_1^*$. Given that, C_2 is also masked by $h(\cdot)$; we have already counted the adversary's advantage regarding B_1:

$$Adv_{\mathcal{D},RW}^{RoR-G_8}(t,R) \leq Adv_{\mathcal{D},RW}^{RoR-G_7}(t,R) + q \cdot \varepsilon_{ECC} \tag{6}$$

Game G_9. This game is identical to G_8, exclude that X_i and X_j are calculated as $X_i = h(PUF(h(PW_i \| ID_i)) \| sk_i \| PK_j)$ $X_j = h(PUF(h(PW_j \| ID_j)) \| sk_j \| PK_j)$. The adversary's advantage compared to G_8 comes from distinguishing the output of $h(\cdot)$ or $PUF(\cdot)$ from a random number:

$$Adv_{\mathcal{D},RW}^{RoR-G_7}(t,R) \leq Adv_{\mathcal{D},RW}^{RoR-G_6}(t,R) + 3.q \cdot \varepsilon_h + q \cdot \varepsilon_{PUF} \tag{7}$$

Game G_{10}. This game is identical to G_9 with the exception the session key is calculated using the hash function as $SK_{ij} = h(C_2 \| PK_i \| PK_j \| TS_j)$, where $C_2 = r_i \cdot r_j \cdot PK_j$. Given that r_i and r_j are nonce values and TS_j is a timestamp, therefore:

$$Adv_{\mathcal{D},RW}^{RoR-G_{10}}(t,R) \leq Adv_{\mathcal{D},RW}^{RoR-G_9}(t,R) + q \cdot \varepsilon_h + q \cdot \varepsilon_{ECC} \tag{8}$$

On the other hand, G_{10} represents the implementation of the EPSG. Hence:

$$\begin{aligned} & Adv_{\mathcal{D},EPSG}^{RoR}(t,R) - Adv_{\mathcal{D},RW}^{RoR}(t,R) \\ & \leq Adv_{\mathcal{D},RW}^{RoR-G_{10}}(t,R) - Adv_{\mathcal{D},RW}^{RoR-G_0}(t,R) \\ & \leq 5 \cdot q \cdot \varepsilon_{ECC} + 7 \cdot q \cdot \varepsilon_h + q \cdot \varepsilon_{PUF} \end{aligned}$$

which completes the proof. □

Fig. 5 Arduino UNO and sensors circuit

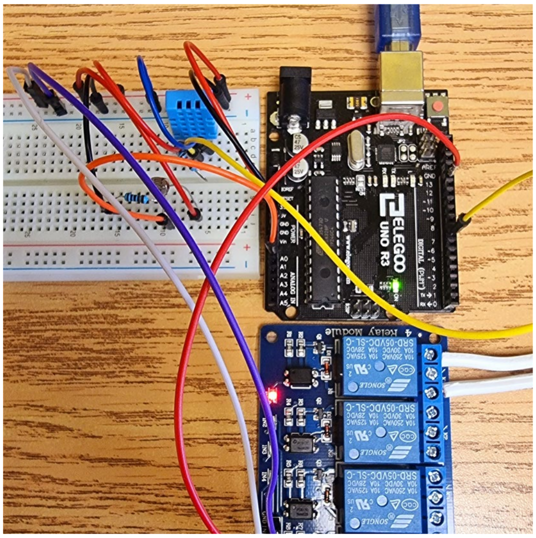
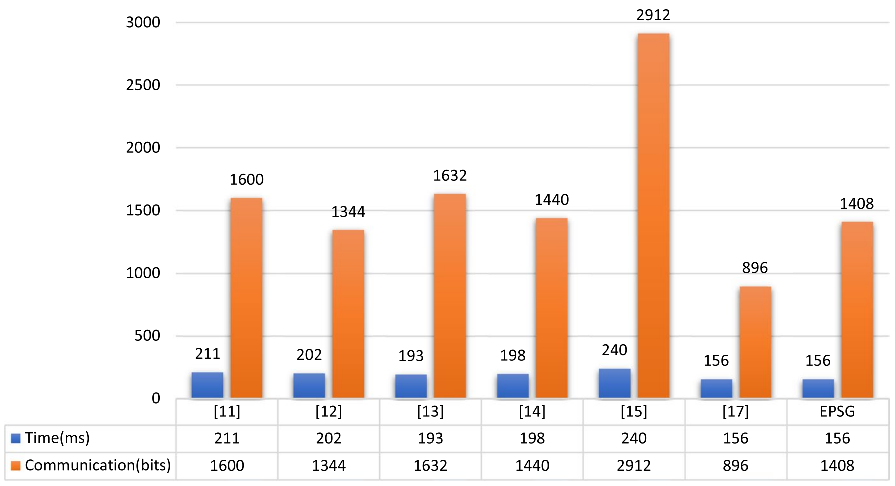


Table 4 Cost comparison of the related protocols and EPSG

Protocol	Computations	Time	Communications	Energy (mJ)
[11]	$11 \times T_h + 6 \times T_{ECC} + 2 \times T_{2ECC}$	211 ms	1600 bits	18.568
[12]	$8 \times T_h + 6 \times T_{ECC} + 2 \times T_{2ECC}$	202 ms	1344 bits	17.776
[13]	$5 \times T_h + 6 \times T_{ECC} + 2 \times T_{2ECC}$	193 ms	1632 bits	16.984
[14]	$10 \times T_h + 8 \times T_{ECC}$	198 ms	1440 bits	17.424
[15]	$19 \times T_h + 4 \times T_{Es} + 8 \times T_{ECC}$	240 ms	2912 bits	21.12
[17]	$11 \times T_h + T_{PUF} + T_{FE.GEN} + T_{FE.REC}$	156 ms	896 bits	13.728
EPSG	$9 \times T_h + T_{PUF} + 6 \times T_{ECC}$	156 ms	1408 bits	13.728

6 Performance analysis of EPSG

In order to compare the EPSG with other similar protocols and summarize the evaluation process, we choose some related protocols from Sect. 2, and evaluation metrics selection is done according to [29, 38, 39]. In Table 4, the comparison is made based on computation cost, execution time, communication cost, and energy consumption. With the purpose of implementing a practical environment to achieve experimental results, as shown in Fig. 5, a smart home network is simulated (including a microcontroller, temperature sensor, humidity sensor, photoresistor sensor, and a relay for controlling AC power). We have used an Arduino UNO board to simulate the cryptographic operations on each smart meter client. This board integrates an ATmega328P microcontroller with 32-kB flash memory, 2-kB SRAM, and a clock speed of 16 MHz. It is worth noting that the PUF reliability results obtained in a similar microcontroller [40]. To use Arduino's SRAM to implement an SRAM



	[11]	[12]	[13]	[14]	[15]	[17]	EPSG
🟦 Time(ms)	211	202	193	198	240	156	156
🟧 Communication(bits)	1600	1344	1632	1440	2912	896	1408

Fig. 6 Time and communication cost comparison

PUF, we have assessed whether the power-up values of the SRAM of 20 microcontrollers acquired 100 times at room temperature exhibit the necessary quality. As a result, the mean bias of all devices (uniformity) is 48.38%, which is very close to the ideal 50%. Furthermore, the intra-distance between different acquisitions (reliability) is 97.58%. Finally, the inter-distance between different devices (uniqueness) is 38.62%, aligning with other results reported in the literature for similar microcontrollers. Using the mentioned platform, we achieved $T_{ECC} \approx 21$ ms, $T_{2ECC} \approx 26$ ms, $T_h \approx 3$ ms for SHA-256[1] and $T_{Es} = 3.7$ ms. We also considered the time of a PUF invocation (T_{PUFn}) equals to T_h. We establish this equivalence because we assume a key management module that can generate multiple keys from a single root key should be used. In this case, a secure key derivation function (KDF) that uses cryptographic primitives such as SHA-256 is used to ensure cryptographic separation between the different derived keys. In [17], fuzzy extractors and helper data among other algorithms are used in functions FE.GEN and FE.REC. According to the data provided in [17], $T_{FE.GEN}$ and $T_{FE.REC}$ can be approximated to $10 \times T_{PUF}$ and $30 \times T_{PUF}$, respectively.

Any client in the EPSG should support the ECC, the hash function, and the PUF (only SM). In terms of computational complexity, the SM_i performs five calls to the hash function (T_h) and a PUF invocation (T_{PUF}), while the SP_j does four calls to the hash function. Besides, the SM_i performs two ECC scalar-multiplications (T_{ECC}) and does a double-scaler-multiplication (T_{2ECC}), e.g., $u \cdot P + v \cdot Q$, and the SP_j does three ECC point-multiplications (T_{ECC}). Hence, in total, a login and key agreement phase of the EPSG costs $9 \times T_h + T_{PUF} + 6 \times T_{ECC}$. Based on this experiment, the execution time of a key agreement session in the EPSG is 156 ms, which is the

[1] SHA-256 may be replaced by SHA-3 if the performance requirements of the systems demand it or whether SHA-256 is considered insecure.

lowest among the compared protocols. As indicated in Table 4, when compared to other protocols, ESPG leverages a combination of PUF functions and the ECC method. This solution not only ensures message confidentiality through the ECC module, but also guarantees message integrity at a low computational cost by utilizing PUF functions.

For the communication overhead comparison, the bit lengths of a timestamp, an identifier, a random number, a hash value, and an ECC point are, respectively, considered as 32, 64, 128, 160, and 320 bits. It should be noted that we consider SHA-256 but truncate its output to 160-bit to avoid the recent security flaws of SHA-1 [41]. Based on this setting, the communication overhead of the EPSG is $320 + 320 + 64 + 160 + 32 = 896$ bits for M_1 and $320 + 160 + 32 = 512$ bits for M_2, and 1408 bits in total. However, the communication cost of [15] has been reported to be 1184 bits for the same parameter set; our independent analysis shows that there should be a typo that led to the underestimation of the communication cost in that scheme. The source of the mistake could be the considered bit length of the values computed using symmetric encryption, and their length should be at least as long as the length of the encrypted values. As illustrated in Fig. 6, the ESPG needs minimum communication overhead in comparison with other protocols.

Energy consumption can be upper-bounded as $Ec = V_{max}.I_{max}.T_c$, where Ec is the energy consumption, I_{max} is the maximum consumed current, V_{max} is the maximum working voltage, and Tc is the total computational time to share a session key. Following ATmega328P datasheet [42], the maximum working power, i.e., $(V.I)$, of ATmega328P is $14\,\text{mA} \times 5.5V = 77\,\text{mW}$ in active mode within 16MH. Following calculation and considering 15% extra energy for the used board, the energy comparison of EPSG is compared with other schemes in Table 4. These results show that the energy consumption of a session of EPSG is less than that of other schemes.

7 Conclusion

Over the last decade, advancements in smart grid technology have enabled the modernization of existing electricity networks. This research paper introduces a novel mutual authentication and key agreement scheme called EPSG. The unique aspect of this study is the integration of physical unclonable functions (PUFs) and elliptic curve cryptography (ECC) modules to provide simultaneous confidentiality and integrity for lightweight smart meters. To validate the functionality of EPSG for constrained IoT devices, we implemented the scheme on an Arduino UNO board and assessed computation, communication costs, and energy consumption. Additionally, a comprehensive security analysis confirmed the robustness of the proposed scheme against security attacks. However, during the simulation, we faced challenges in finding a smart meter capable of running a PUF function, despite achieving satisfactory results with real home appliances. Thus, for future research endeavors, it would be valuable for the authors to explore alternative platforms to evaluate the effectiveness of this solution or consider developing a new method in different layers of a smart grid, based on the foundation of EPSG.

Author contributions Samad Rostampour, Nasour Bagheri, and Saru Kumari contributed equally to the conception and design of the proposed scheme, named EPSG. They also conducted the security analysis and performance evaluation of the scheme and wrote the corresponding sections of the manuscript. Ygal Bendavid provided Carmen Cámara valuable insights into the design and implementation of the Physical Unclonable Function (PUF) module used in the proposed scheme. He also contributed to the writing and editing of the manuscript. Honorio Martín provided technical guidance on the implementation of the EPSG scheme on the Arduino UNO microcontroller. He also conducted the comparative performance evaluation and contributed to the writing and editing of the corresponding sections of the manuscript. Behnam Ghavami provided overall supervision and guidance throughout the project. He contributed to the conception and design of the proposed scheme, supervised the security analysis and performance evaluation, and critically revised the manuscript for important intellectual content.

Funding This work has been partially supported by the Madrid Government (Comunidad de Madrid-Spain) under the Multiannual Agreement with UC3M ("Fostering Young Doctors Research," PUCFA-CM-UC3M), and in the context of the VPRICIT (Research and Technological Innovation Regional Program). Nasour Bagheri was supported by Shahid Rajaee Teacher Training University under grant number 4968.

Availability of data and materials Not applicable.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval Not applicable.

References

1. Avancini DB et al (2019) Energy meters evolution in smart grids: a review. J Clean Prod 217:702–715
2. Breque M, De Nul L, Petridis A (2021) Industry 5.0 : towards a sustainable, human-centric and resilient European industry. European Commission, Directorate-General for Research and Innovation. Accessed 31 Sept 2022
3. Sakhnini J et al (2019) Security aspects of internet of things aided smart grids: a bibliometric survey. Internet of Things 100111
4. Siboni S, Sachidananda V, Meidan Y, Bohadana M, Mathov Y, Bhairav S, Shabtai A, Elovici Y (2019) Security testbed for internet-of-things devices. IEEE Trans Reliab 68(1):23–44. https://doi.org/10.1109/TR.2018.2864536
5. Macola IG (2021) The five worst cyberattacks against the power industry since 2014. In: European Network of Transmission System Operators for Electricity (ENTSO-E). Accessed 13 Mar 2021
6. Kang W, Pan Y, Srivastava G (2021) The reliability of IoT networks with characteristics of abnormal induced signals. IEEE Trans Reliab 70(2):808–818. https://doi.org/10.1109/TR.2020.3021376
7. Kumar P, Gurtov AV, Sain M, Martin AP, Ha PH (2019) Lightweight authentication and key agreement for smart metering in smart energy networks. IEEE Trans. Smart Grid 10(4):4349–4359
8. Baghestani SH, Moazami F, Tahavori M (2022) Lightweight authenticated key agreement for smart metering in smart grid. IEEE Syst J 16(3):4983–4991
9. Kumar N et al (2019) ECCAuth: a secure authentication protocol for demand response management in a smart grid system. IEEE Trans Ind Inf 15(12):6572–6582
10. Yu S et al (2020) Privacy-preserving lightweight authentication protocol for demand response management in smart grid environment. Appl Sci 10(5):1758
11. Wu F et al (2019) A lightweight and provably secure key agreement system for a smart grid with elliptic curve cryptography. IEEE Syst J 13(3):2830–2838

12. Garg S et al (2020) Secure and lightweight authentication scheme for smart metering infrastructure in smart grid. IEEE Trans Ind Inform 16(5):3548–3557
13. He D et al (2016) Lightweight anonymous key distribution scheme for smart grid using elliptic curve cryptography. IET Commun 10(14):1795–1802
14. Abbasinezhad-Mood D, Nikooghadam M (2018) An anonymous ECC-based self-certified key distribution scheme for the smart grid. IEEE Trans Ind Electron 65(10):7996–8004
15. Khan AA et al (2020) PALK: Password-based anonymous lightweight key agreement framework for smart grid author links open overlay panel. Int J Electr Power Energy Syst 121:106121
16. Rincón AER et al (2021) Securing smart meters through physical properties of their components. IEEE Trans Instrum Meas 70:1–11. https://doi.org/10.1109/TIM.2020.3041098
17. Gope P, Sikdar B (2018) Privacy-aware authenticated key agreement scheme for secure smart grid communication. IEEE Trans Smart Grid 10(4):3953–3962
18. Braeken A et al (2018) Efficient and provably secure key agreement for modern smart metering communications. Energies 11(10):2662
19. Mustapa M, Niamat MY, Nath APD, Alam M (2018) Hardware-oriented authentication for advanced metering infrastructure. IEEE Trans Smart Grid 9(2):1261–1270. https://doi.org/10.1109/TSG.2016.2582423
20. Harishma B et al (2022) Safe is the new smart: PUF-based authentication for load modification-resistant smart meters. IEEE Trans Dependable Secur Comput 19(1):663–680. https://doi.org/10.1109/TDSC.2020.2992801
21. Liu J, Ke L (2019) New efficient identity based encryption without pairings. J Ambient Intell Humaniz Comput 10(4):1561–1570
22. Salimi M (2021) A new efficient identity-based encryption without pairing. Cryptology ePrint Archive
23. Lounis K (2021) PUF security: reviewing the validity of spoofing attack against safe is the new smart. IACR Cryptol. ePrint Arch., 985
24. Safkhani M, Rostampour S, Bendavid Y, Sadeghi S, Bagheri N (2022) Improving RFID/IoT-based generalized ultra-lightweight mutual authentication protocols. J Inf Secur Appl 67:103194
25. Flexible key provisioning with SRAM PUF. In: WHITE PAPER: Intrinsic ID (2020). http://www.intrinsic-id.com/wp-content/uploads/2020/10/White-Paper-Flexible-Key-Provisioning-with-SRAM-PUF.pdf
26. Barbareschi M et al (2021) On the adoption of physically unclonable functions to secure IIoT devices. IEEE Trans Ind Inform 1–1. https://doi.org/10.1109/TII.2021.3059656
27. Dolev D, Yao A (1983) On the security of public key protocols. IEEE Trans Inf Theory 29(2):198–208
28. Abdalla M et al (2005) Password-based authenticated key exchange in the three-party setting. In: Vaudenay S (ed) PKC 2005. Lecture Notes in Computer Science, vol 3386. Springer, Berlin, pp 65–84
29. Qiu S, Wang D, Xu G, Kumari S (2020) Practical and provably secure three-factor authentication protocol based on extended chaotic-maps for mobile lightweight devices. IEEE Trans Dependable Secur Comput
30. Wisiol N, Thapaliya B, Mursi KT, Seifert J-P, Zhuang Y (2022) Neural network modeling attacks on arbiter-PUF-based designs. IEEE Trans Inf Forensics Secur 17:2719–2731. https://doi.org/10.1109/TIFS.2022.3189533
31. Shi J, Lu Y, Zhang J (2020) Approximation attacks on strong PUFs. IEEE Trans Comput Aided Des Integr Circuits Syst 39(10):2138–2151. https://doi.org/10.1109/TCAD.2019.2962115
32. Wang A, Tan W, Wen Y, Lao Y (2021) NoPUF: A novel PUF design framework toward modeling attack resistant PUFs. IEEE Trans Circuits Syst I Regul Pap 68(6):2508–2521. https://doi.org/10.1109/TCSI.2021.3067319
33. Elmitwalli E, Ni K, Köse S (2022) Machine learning attack resistant area-efficient reconfigurable ISING-PUF. IEEE Trans Very Large Scale Integr (VLSI) Syst 30(4):526–538. https://doi.org/10.1109/TVLSI.2022.3144236
34. Yu M-D, Hiller M, Delvaux J, Sowell R, Devadas S, Verbauwhede I (2016) A lockdown technique to prevent machine learning on PUFs for lightweight authentication. IEEE Trans Multi Scale Comput Syst 2(3):146–159. https://doi.org/10.1109/TMSCS.2016.2553027
35. Barbareschi M, De Benedictis A, Mazzocca N (2018) A PUF-based hardware mutual authentication protocol. J Parallel Distrib Comput 119:107–120. https://doi.org/10.1016/j.jpdc.2018.04.007
36. Chen S, Li B, Chen Z, Zhang Y, Wang C, Tao C (2022) Novel strong-PUF-based authentication protocols leveraging Shamir's secret sharing. IEEE Internet Things J 9(16):14408–14425. https://doi.org/10.1109/JIOT.2021.3065836

37. Rai VK, Tripathy S, Mathew J (2020) 2SPUF: Machine learning attack resistant SRAM PUF. In: 2020 Third ISEA Conference on Security and Privacy (ISEA-ISAP), pp 149–154 (2020). https://doi.org/10.1109/ISEA-ISAP49340.2020.235013
38. Wang D, Wang P (2016) Two birds with one stone: two-factor authentication with security beyond conventional bound. IEEE Trans Dependable Secur Comput 15(4):708–722
39. Bonneau J et al (2021) The quest to replace passwords: a framework for comparative evaluation of web authentication schemes. In: 2012 IEEE Symposium on Security and Privacy. IEEE, pp 553–567
40. Bonneau J et al (2020) Long-term continuous assessment of SRAM PUF and source of random numbers. In: DATE, pp 7–12. https://doi.org/10.23919/DATE48585.2020.9116353
41. Leurent G, Peyrin T (2019) From collisions to chosen-prefix collisions application to full SHA-1. In: Ishai Y, Rijmen V (eds) Advances in Cryptology—EUROCRYPT 2019. Lecture Notes in Computer Science, vol 11478. Springer, Berlin, pp 527–555
42. Atmel: 8-bit AVR Microcontroller with 32K Bytes In-System Programmable Flash. microchip. Available online: http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf. Accessed 10 June 2020

Authors and Affiliations

Samad Rostampour[1,4] · Nasour Bagheri[2,3] · Behnam Ghavami[4] · Ygal Bendavid[5] · Saru Kumari[6] · Honorio Martin[7] · Carmen Camara[8]

✉ Nasour Bagheri
Nbagheri@sru.ac.ir

✉ Behnam Ghavami
ghavami@uk.ac.ir

Samad Rostampour
rostamps@vaniercollege.qc.ca

Ygal Bendavid
Bendavid.ygal@uqam.ca

Saru Kumari
Saryusiirohi@gmail.com

Honorio Martin
hmartin@ing.uc3m.es

Carmen Camara
macamara@inf.uc3m.es

[1] Computer Science Department, Vanier College, Montreal, QC, Canada

[2] Electrical Engineering Department of Shahid Rajaee Teacher Training University (SRTTU), Tehran, Iran

[3] School of Computer Science (SCS), Institute for Research in Fundamental Sciences (IPM), Tehran, Iran

[4] Computer Engineering Department, Shahid Bahonar University, Kerman, Iran

[5] AOTI Department, UQAM University, Montreal, QC, Canada

[6] Mathematics Department, Chaudhary Charan Singh University, Meerut, India

[7] Electronic Technology Department, University Carlos III of Madrid, Madrid, Spain

[8] Computer Science Department, University Carlos III of Madrid, Madrid, Spain