Peer-to-Peer Networking and Applications
https://doi.org/10.1007/s12083-022-01305-8

An authentication and key agreement scheme for smart grid

Masoumeh Safkhani[1] · Saru Kumari[2] · Mohammad Shojafar[3] · Sachin Kumar[4]

Received: 12 November 2020 / Accepted: 17 February 2022

Abstract
The Internet of Things (IoT) plays a crucial role in the new generation of smart cities, in which developing Internet of Energy (IoE) in the energy sector is a necessity also. Several schemes have been proposed so far and in this paper we analyze the security of a recently proposed authentication and key agreement framework for smart grid named PALK. Our security analysis demonstrates that an attacker can extract the user permanent identifier and password, which are enough to do any other attacks. To remedy the weaknesses and amend PALK, we propose an improved protocol based on Physical Unclonable Function(PUF) to provide desired security at a reasonable cost. We also prove the semantic security of constructed scheme by using the widely-accepted real and synthetic model, under the computationally hard Diffie-Hellman assumption. Computational and communication cost analysis of the improved protocol versus PALK, based on identical parameter sets on our experimental results on an Arduino UNO R3 board having microcontroller ATmega328P, shows 46% and 23% enhancements, respectively. We also provide, the energy consumption of the proposed protocol and each session of the protocol consumes almost 24 mJ energy. It shows that it is an appropriate choice for constrained environments, such as IoE.

Keywords Internet of energy · Smart grid · Authentication · Elliptic curve cryptography · Physical unclonable function

1 Introduction

The Internet of Things (IoT)-based technologies are enabling the development of a new generation of smart cities. By utilising sustainable information and communication technologies, a smart city can improve the quality of our lives, education, and health, as well as the performance of urban services. Many other concepts, such as smart grids, smart transportation, and smart buildings, can be included in the broad concept of a smart city. In a future city, infrastructure, services, and technologies are integrated to create a city that is designed for the needs of its residents, who want new urban energy services ranging from the indoor environment to private and public transportation and a dedicated healthcare system. Energy, water, transportation, health, and safety will be coordinated in future cities to provide a clean, economical, and safe living environment. To that end, Smart Grid (SG), as the energy infrastructure of a smart city, will undoubtedly be an important urban infrastructure to support the realisation of a sustainable future city [1]. On the other hand, the global warming and energy crisis have raised serious concerns about climate change, energy costs, and limited non-renewable energy resources, such as fossil fuels. Furthermore, the use of fossil fuels is the primary source of greenhouse gases, which is the primary cause of global warming. As a result, renewable energy sources such as solar cells and wind turbines, which are more environmentally friendly than fossil fuels, should be increasingly integrated into the power grid. However, this integration is a difficult task that necessitates massive information transmission in order to monitor and stabilise system states in real-time.

✉ Saru Kumari
Saryusiirohi@gmail.com

Masoumeh Safkhani
Safkhani@sru.ac.ir

Mohammad Shojafar
M.Shojafar@surrey.ac.uk

Sachin Kumar
imsachingupta@rediffmail.com

[1] Computer Engineering Department, Shahid Rajaee Teacher Training University and School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Tehran 16788-15811, Iran

[2] Department of Mathematics, Chaudhary Charan Singh University, Meerut 250004, India

[3] ICS/5GIC, University of Surrey, Guildford GU27XH, UK

[4] Department of Computer Science and Engineering, Ajay Kumar Garg Engineering College, Ghaziabad 201009, India

Published online: 10 March 2022

The smart grid can include anything from massive high-voltage transformers to gas pipelines to corrosion-resistant paint applied to outdoor equipment. Smart meters are the most prominent example of a smart grid. Smart meters are now widely used throughout the world. As a result, it makes a large amount of data available. But it's not just meters that have gone smart. The entire energy backbone and distribution network is already in use in many countries, and it immediately begins sending large amounts of operational data. However, in other cases, such as with smart meters, a large amount of data is available. The presence of data, as well as the work that can be done to secure it, can ensure the accuracy of the data and commands sent in these smart grids. If smart grid security is not taken into account, the massive amount of data generated by the smart grid itself can be used to deceive the network and do illegal things. As a result, authentication and key agreement (AKA) protocols are just as important in this area as they are in others.

Interestingly, IoT as a medium that can connect anything at any time could provide the required backbone to transmit smart grid's sensing massive information. The combination of smart grid and IoT is called the Internet of Energy (IoE) [2, 3]. The IoE adopts large number of distributed power generation equipment, energy storage facilities and IoT based technologies to facilitate energy sharing and promote the use of electrical grids and maintain their safety [4]. Besides, to achieve environment sustainability and manage energy resources and consumers such as power plants, smart grids, smart buildings, smart vehicles and other smart objects efficiently, IoE developing in the energy sector is a necessity. Hence, IoE, as the integration of IoT and smart grid, has attracted the researchers' attention significantly. Today, the concept of smart city makes all this possible, where an extensive number of sensors and sensor networks are available that could be used to implement IoE efficiently, as an important part of the smart management of the city [5]. IEEE describes the smart grid (SG) as [6] "encompasses the integration of power, communications, and information technologies for an improved electric power infrastructure that serves end-use applications and loads". SG brings new information and communication technologies into the traditional power grids. Hence, SG should be considered as a network that includes a variety of operations, services and energy measures including smart meters, smart appliances, renewable energy resources and other energy resources such as power plants and also transmission and distribution networks [7].

Integrating advanced communications and information technologies into traditional power grids provides many opportunities such as live monitoring of the energy consumption of any consumer, better resource allocations, two-way energy flow, prediction and preventing outages, integration of micro energy generators such as solar power into the electricity grid and many others. However, this massive integrated network also requires massive bi-directional communication between all entities of SG, while in a power grid the communication was generally limited to the energy providers such as major power plants and transmission and distribution networks. However, in an SG, any node, e.g. a smart meter or an electrical vehicle, could be the source to send or receive data to/from another node of the grid. It means that SG has been merged with IoT more and more to perform IoE and this attracted the researchers attention [8–12]. IoE, as a web-enabled usage of SG, provides many benefits for the energy generators/sources, distribution operators, consumers and stakeholders because of its flexibility and intelligence-based monitoring, management and control, and data processing. Therefore, using IoE, it is possible to make the energy chain cost-efficient, safe, reliable, and flexible. A high-level structure of an IoE ecosystem is depicted in Fig. 1.

The massive flow of information in IoE has many advantages for the management of energy consumption and production and reduces many energy losses, which at a higher level can impact many aspects of a smart city's inhabitants' lives by providing more green energy by better utilizing renewable energy resources and reducing fossil fuel usage, which leads to lower environmental emissions. However, any public or distributed communication network will always have security and privacy concerns. In other words, the use of communication technologies in smart grids makes them vulnerable to cyber-attacks [13, 14]. The majority of these attacks are designed to compromise the SG's confidentiality, integrity, and availability of information. To protect the SG from these attacks, a number of security protocols are in place. One of the most recent challenges that must be addressed to ensure that these protocols are free of flaws is their security. Attempts to obtain and eliminate all of these protocols' vulnerabilities continue to be a research focus for researchers. As a result, we concentrated our efforts on this critical issue.

1.1 Challenges and motivation

To ensure network security, an authentication mechanism must be used. Following authentication and the establishment of a secret key, protocol participants can use the secret key and accompanying cryptographic techniques to protect their own confidential information in an unprotected communication channel. One authentication mechanism can achieve authentication, and a key agreement protocol can achieve the goal of establishing a secret key. There are protocols that can achieve both of these goals at the same time. These protocols are known as authentication and key agreement (AKA) protocols.

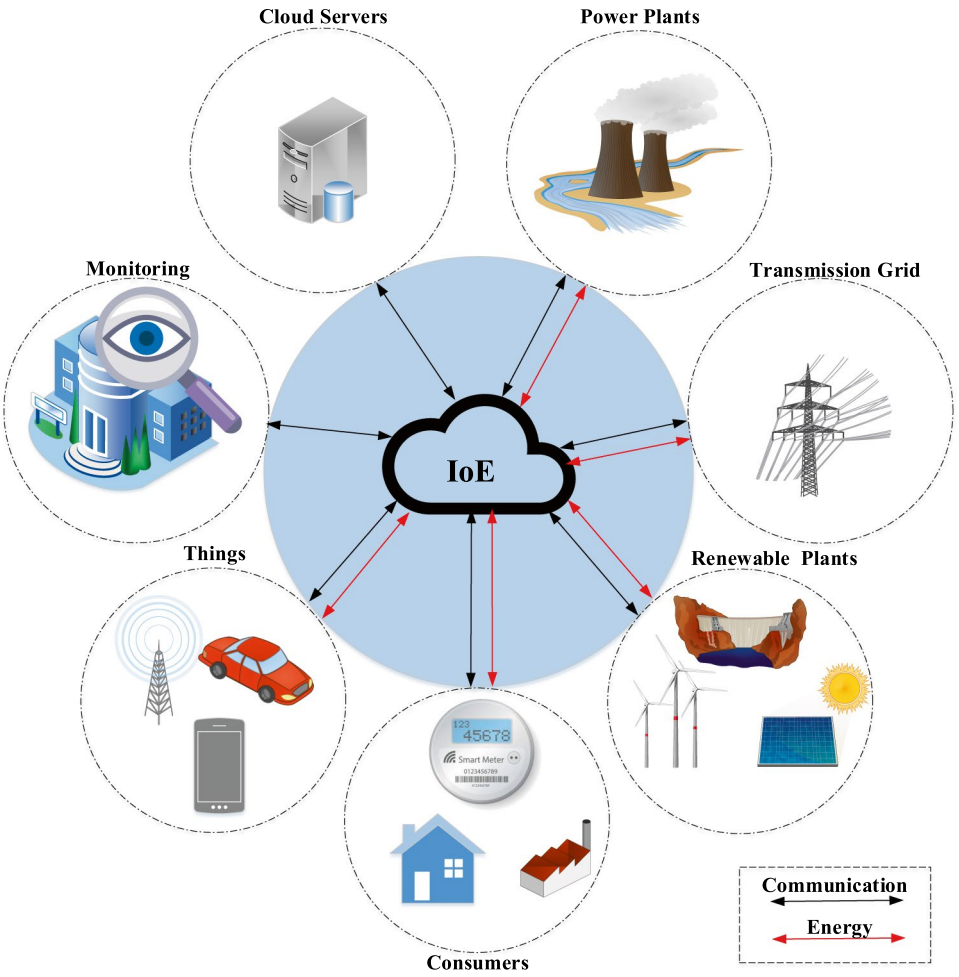

Fig. 1 An IoE ecosystem (Thanks to **conceptdraw** https://www.conceptdraw.com for the icons)

Many recent studies have attempted to solve the SG infrastructures' security and privacy concerns. A survey of security and privacy challenges in SG, by Kumar et al., can be found in [15]. Authentication and key management in many domains, which is the foundation for much proper access to the services delivered over SG, is a significant security concern in SG. A recent survey of key management systems for SG infrastructure is conducted by Ghosal et al. [16]. Among the several techniques to managing key agreements between different entities, the authentication-based technique is more popular due to its better level of trustworthiness. In this study, we focus on studies that use Elliptic Curve Cryptography (ECC) as the core of security to share an authenticated key between two clients, such as a smart meter and its corresponding gateway or service provider, among various approaches to designing authentication-based key agreement protocols. ECC offers a design benefit over other public-key cryptosystems like RSA in that it requires a smaller key size, which is beneficial in constrained environments [17] such as internet of things (IoT) and many edge devices that are used in an SG infrastructure. Several schemes have been developed in this context in recent years, e.g. [18–20]. Among them, a scheme named PALK [19] could be the most recent proposal which has been designed by Khan et al.. It is an ECC-based authentication and key agreement protocol that enables two users to share a session key through the use of a trusted authority. In addition, symmetric encryption is used to reduce the computational cost of the protocol's instances. Based on the designer's computation and communication costs analysis and comparison to the state of the art, PALK outperforms related works. As a result, if it also provides the necessary security, it may be a viable option for restricted devices in an IoE system. These characteristics prompted us to conduct a security analysis in order to identify its benefits and drawbacks in this work.

1.2 Our contribution

The main contribution of this paper contains the folds below:

- We present efficient passive and active attacks against PALK [19], a recently proposed elliptic curve based key agreement scheme for SG. The proposed attacks rule out any possible application for this protocol.

- We show that the analyzed protocol does not meet the expected functionality, due to the inappropriate storage of the information.
- We propose an amended version of PALK which uses PUF and show its security heuristically against the common attacks in this context such as traceability attacks, desynchronization attacks, secret disclosure attacks and data confidentiality compromising attacks. In addition, we formally evaluate its security using the widely-accepted real and synthetic model.

1.3 Organization

The rest of the paper is organized as follows: We review the latest related work in the field of smart grid security in Sect. 2. In Sect. 3, required preliminaries, including notations and description of PALK are provided. In Sect. 4, we analyze the security and functionality of this protocol, where we show its critical pitfalls. We propose an improved protocol in Sect. 5. The improved protocol's analysis is described in Sect. 6. In Sect. 7, we compare the improved protocol's security and computational and communication costs with those of some related protocols, include PALK and iPALK. Finally, the paper is concluded with concluding remarks in Sect. 8.

2 Related work

In 2011, Fouada et al. [21] and Wu and Zhou [22] independently proposed key agreement schemes based on the Diffie-Hellman key exchange protocol and Elliptic Curve Cryptography (ECC). Both of these schemes required a public key infrastructure (PKI) to implement. Later in [23], the man in the middle attack was presented against the protocol of [22]. In 2012, Sule et al. [24] presented a similar protocol to [21] scheme for smart grid communications, which do not retain the features of message authentication, user anonymity, insider and DoS attacks resistance.

Xia and Wang [23] also presented a scheme based on trusted third party and lightweight directory access protocol (LDAP) to eliminate the high cost of PKI implementation and maintenance. Later in [25], Xia and Wang's scheme vulnerability against authentication attacks and anonymous key sharing attacks was shown. The MCEPAK protocol was introduced in [26] based on ECC, x.1035 standard and the Diffie-Hellmann algorithm, which requires that passwords be pre-set between the home area networks and different devices. In 2015, a key distribution scheme was proposed that also addresses the anonymity of smart meters [27]. Security and efficiency of this scheme were independently improved by Odelu et al. [28] and He et al. [29].

In all three [27, 28, 30], schemes have been proposed in which the private keys of smart meters and service providers are produced based on their identities and by a trusted anchor. It is worth mentioning that there is no need for a trusted anchor participation in the authentication process. Mahmoud et al. [31] proposed an RSA-based protocol that is computationally better than Fouda et al. [21] and Sule et al. [24] schemes, however, data privacy, internal attack resistance and session key protection were not maintained in this scheme [19]. In 2017, Mahmoud et al. [32] introduced another protocol for smart grid communications that is better than their own previous scheme [31] because it does not have many security pitfalls of the previous scheme.

Chen et al. [33] proposed an anonymous authentication protocol and key establishment employed in smart grid, which was shown in [34] to be vulnerable to DoS attacks. In [18], in order to solve the challenges of key distribution schemes in smart grids, and specifically to solve the PKI maintenance problem, an anonymous ECC-based self-certificated scheme is proposed, the security of which is also proved in the random oracle model and through ProVerif tool. The same authors in [35], presented an authentication scheme for smart grid which in [34] is shown that it has no security against anonymity and confidentiality contradiction attacks. In 2019, Lee et al. [36] proposed another scheme for anonymous authentication on the smart grid. In the same year, Wu et al. [20] presented another elliptic curve cryptography based AKA scheme for SG. They proved their scheme security using ProVerif tool under the Dolev-Yao model [37]. They also simulated their proposed scheme using NS-3, to show its practicality in communication.

In [19], a password-based anonymous key agreement framework for smart grid communications named PALK is introduced and claimed PALK provides strong anonymity, confidentiality, non-traceability, perfect forward secrecy and mutual authentication. In [38], Chaudhry showed that PALK has design flaws due to multiplication of two points over the elliptic curve. They also fixed these design flaws of the PALK which led to proposing iPALK. The security of iPALK is proved through BAN logic and ProVerif tool.

In this paper, assuming that PALK works properly and does not have design flaws, we show that PALK suffers from seriously passive and active attacks which eliminates its use. We also modify PALK so that in addition to fixing all security vulnerabilities, it is secure against all known active and passive attacks. Precisely, in this paper, we propose an improved protocol based on Physical Unclonable Function (PUF) to provide desired security at a reasonable cost to address the weaknesses of the PALK. We also demonstrate the semantic security of the constructed scheme by employing the widely accepted real and synthetic models under the computationally hard Diffie-Hellman assumption. Computational and communication cost analysis of the improved

protocol versus PALK, based on identical parameter sets on our experimental results on an Arduino UNO R3 board with microcontroller ATmega328P, reveals 46% and 23% improvements, respectively. We also provide the energy consumption of the proposed protocol, which consumes nearly 24 mJ of energy per session.

3 Preliminaries

In this section, we introduce required notations, a brief description of elliptic curve-based cryptography, and also represent PALK protocol. The notations used throughout this paper are listed in Table 1.

3.1 Elliptic curve cryptography

Elliptic Curve Cryptography (ECC) is a public-key cryptography approach based on a group $\mathbb{G}$, which is defined over an elliptic curve. Let q be a large prime number. An elliptic curve E_{F_q} over the finite field F_q is defined as the set of all $(x, y) \in F_q \times F_q$ such that $\lambda^2 = \mu^3 + a\mu + b$, where $a, b \in F_q$ and $4a^3 + 27b^2 \ mod \ q \ \neq 0$, along with a distinguished point at infinity which is denoted by $\mathcal{O}$. Then $\mathbb{G} = \{(\lambda, \mu) \in E_{F_q} \cup \mathcal{O}, +\}$ is a group. If there is an element $P \in \mathbb{G}$ that its different orders can generate all elements of the group, $\mathbb{G}$ is called a cyclic group and P is called a generator of the group. The order of an element $Q \in \mathbb{G}$ is denoted as the smallest positive number n such that $nQ = \mathcal{O}$. Assuming that n is enough large, given any natural scalar $a \in F_q$ and $P = \{(\lambda, \mu) \in E_{F_q}\}$ of order n, it is easy to calculate $y = a \times P$. However, given y, E_{F_q} and P, it is computationally infeasible to determine a, which is known as Elliptic Curve Discrete Logarithm Problem (ECDLP). Similarly, for $a, b \in F_q$, given $a \times P$, $b \times P$, E_{F_q} and P, it is computationally infeasible to determine $a \times b \times P$, which is known as Elliptic Curve Computational Diffie-Hellman Problem (EC-CDHP).

3.2 Semantic security in the real-or-random model

In a three-party password-authenticated key agreement scheme, the scheme's parties use their password to share a common session key SK, which is then used to build secure channels [39]. In such schemes, a protocol's party is either a client $U \in \mathcal{U}$ or a trusted server $S \in \mathcal{S}$. Any client U could be either honest or malicious and holds a long-lived key password pw_U. The server S holds a vector $pw_S = \left\langle pw_S[U] \right\rangle_{U \in \mathcal{U}}$, contains an entry for each client U. $pw_S[U]$ defines a transformation of pw_U. If U is a malicious client, pw_U is assumed to be known by the adversary. If two clients U_i and U_j share the same session identifications, we call them *partners*.

To determine the adversary's ability to distinguish a real session key from a random one, we define b to be a bit chosen uniformly at random at the beginning of the experiment. In general terms, the adversary (A) controls all the public communications between all the participants and interacts passively or actively with them. Specifically, following [39], $\mathcal{A}$ can run the following queries:

– Execute(U_i, S, U_j) query. This query models a passive adversary $\mathcal{A}$ eavesdrops on the channel, and gets read access to the exchanged messages between U_i, S and U_j in the honest execution of the protocol. The output of this query consists of the messages that were exchanged during that session of the protocol.

Table 1 Used notations

Symbol	Description	Symbol	Description
E_{F_q}	Elliptic curve over the field F_q	$\mathbb{G}$	A prime group over E
q	A large prime number	F_q	The field over $\{0, 1, \ldots, q-1\}$
$a.P$	Multiplying P by scalar a	TA	The third authority
U_i	i-th user/client	r, s, m	Random numbers
T	Timestamp	ID	Identifier
P	Generator point of the group $\mathbb{G}$	sk	Secret key
PK	Public key	PW	Password
PID	Pseudo ID	$h(.)$	One-way hash function
$ES_K(.)$	Symmetric encryption using the key K	$DS_K(.)$	Symmetric decryption using the key K
$\oplus$	Bitwise XOR operation	$+$	Modular addition
$\|\|$	Concatenation	$A \stackrel{?}{=} B$	Determine whether A and B are equal
$\mathcal{A}$	Adversary	$T_{\mathcal{F}}$	Computational complexity of function $\mathcal{F}$
Z_q^*	The set of integers $\{1, \ldots, q-1\}$	$\mathcal{G}_n$	Series of games used in semantic security proof

- Execute(U_i, U_j) query. This query also models a passive adversary $\mathcal{A}$ eavesdrops on the channel, and gets read access to the exchanged messages between U_i and U_j in the honest execution of the protocol. The output of this query consists of the messages that were exchanged during that session of the protocol.
- serverSend(S, m) query. This query models an active adversary that may intercept a message and then either modify it, create a new one, or simply forward S would returns upon receipt of message m.
- clientSend(U_i, m) query. This query models an active adversary that may intercept a message and then either modify it, create a new one, or simply forward it to the target participant. The output of this query is the message that U_i would returns upon receipt of message m.
- Reveal(U_i) query. The output of this query is the session key held by the instance U_i, if a session key defined for U_i and Test query was not asked to either U_i or to its partner; Otherwise, it returns the undefined symbol $\perp$.
- Test(U_i) query. If no session key for instance U_i is defined or if a Reveal query was asked to either U_i or to its partner, then it returns the undefined symbol $\perp$. Otherwise, it returns the session key for instance U_i if $b = 1$ or a random of key of the same size if $b = 0$.

It is clear, Test(U_i) is meaningful if both U_i and its partner are honest.

Let's consider an execution of a password-authenticated key agreement protocol $\mathcal{P}$, influenced by an adversary $\mathcal{A}$, in which $\mathcal{A}$ is given access to the Execute, Send, and Test oracles, and outputs a guess bit b_0. The adversary wins the game defining the semantic security in the Real-or-Random (RoR) sense if $b_0 = b$, where b is the hidden bit which is used by the Test oracle. The adversary's advantage to win this game, $Adv^{RoR}_{\mathcal{D},\mathcal{P}}(t, R)$, is defined as follows:

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

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

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

and $\varepsilon(.)$ being some negligible function, where the maximum is taken over all $\mathcal{A}$ with time-complexity at most t and using resources at most R, which could be the number of queries to its oracles. If we aim to model malicious clients, then we can give the adversary access to Reveal oracle also.

This mathematical proof will be used in this paper to demonstrate the improved protocol's security.

3.3 Physically unclonable functions

Physically Unclonable Functions (PUF) are used to generate cryptography secret keys that are generated on demand and cleared immediately after use in order to provide security in systems. The output of a PUF is determined by random physical factors (unpredictable and uncontrollable) that exist naturally or are introduced at random during the manufacturing process. As a result, it is nearly impossible to copy or simulate a PUF. PUF technology creates a digital fingerprint for its associated security IC by default, which can be used as a unique key / secret value to support security protocols and services such as encryption / decryption, authentication, and digital signature. It worth noting PUF implementation is another part of research for example we refer interested readers to researches [40–46]. It can be seen from related literature that the cost of a challenge-response call to a PUF depends on different parameters including the auxiliary circuits that are used to enhance its reliability. More precisely, the fact is the PUF itself is very fast, may be nano-sec to give a response, much more faster than a hash function. But, for the PUF there are also additional delays due to communication, and also majority voting to improve reliability. There is a pretty large communication delay from PC to FPGA hosting the PUF if one implements the protocol and the PUF in different platform. However, if this process is done on the hardware and there is no communication with the PC it should be very fast. On the other hand, the implementation cost of a reliable PUF has a decreasing trend and considering a hash call as a cost of challenge-response is reasonable currently [47].

3.4 PALK protocol

In this section, we provide a brief description of PALK [19], which is a mutual authentication and key agreement protocol between two users, through a trusted authority (TA), for SG. This protocol includes four phases denoted by initialization phase, registration phase, login and key agreement phase, and password change phase.

Through the initialization phase, TA chooses an elliptic curve E_{F_q} and a generator P over $\mathbb{G}$ and a hash function $h(.)$. It also selects $sk_{TA} \in F_q$ as its secret key and its public key i.e. PK_{TA} will be $sk_{TA} \times P$. Finally, TA discloses the public parameters of the system, i.e. $\{E_q(c, d), q, P, h(.)\}$ and its public key PK_{TA} and keeps sk_{TA} secret.

The next phase of PALK, following Fig. 2, is the registration phase which is run between TA and a user U_i as follows:

1. U_i chooses its password PW_i and its identifier ID_i, generates a random integer $r_i \in Z^*_q$, calculates $A_i = h(PW_i \| r_i \| ID_i), X_i = A_i.P, PWI_i = PW_i \oplus h(ID_i \| X_i)$

Fig. 2 Registration phase of PALK over the secure channel

U_i	$Message$	TA
$r_i \in Z_q^*$, $A_i = h(PW_i \| r_i \| ID_i)$, $X_i = A_i.P$, $PWI_i = PW_i \oplus h(ID_i \| X_i)$		
	$\xrightarrow{\{PWI_i, ID_i, X_i, TS_i\}}$	Checks TS_i, $r_{TA} \in Z_q^*$, $PW_i^* = PWI_i \oplus h(ID_i \| X_i)$, $B_i = h(PW_i^* \| r_{TA} \| ID_i)$, $Y_i = B_i.P$, $W_i = X_i + Y_i$, $S_i = h(ID_i \| W_i \| PWI_i)$, $S_i' = S_i \oplus h(W_i \| X_i)$
$S_i^* = S_i' \oplus h(W_i \| X_i)$, $sk_i = A_i + B_i + S_i^*$, $PK_i = sk_i.P$, $PK_i \stackrel{?}{=} W_i + S_i^*.P$, stores $\{S_i^*, W_i\}$	$\xleftarrow{\{W_i, B_i, S_i'\}}$	

and sends the tuple $\{PWI_i, ID_i, X_i, TS_i\}$ to TA, over a secure channel.

2. Once received the message, TA checks the timestamp TS_i, generates a random integer $r_{TA} \in Z_q^*$, calculates $PW_i^* = PWI_i \oplus h(ID_i \| X_i)$, $B_i = h(PW_i^* \| r_{TA} \| ID_i)$, $Y_i = B_i.P$, $W_i = X_i + Y_i$, $S_i = h(ID_i \| W_i \| PWI_i)$, $S_i' = S_i \oplus h(W_i \| X_i)$ and sends the tuple $\{W_i, B_i, S_i'\}$ to U_i, over a secure channel.
3. U_i computes $S_i^* = S_i' \oplus h(W_i \| X_i)$, $sk_i = A_i + B_i + S_i^*$ and $PK_i = sk_i.P$ and verifies whether $PK_i \stackrel{?}{=} W_i + S_i^*.P$ to store $\{S_i^*, W_i\}$ in its database.

In the login and key agreement phase of the protocol, between U_i and U_j, that they are communicating to establish a session key, the process is as follows, see Fig. 3:

1. U_i enters its identity ID_i and its password PW_i, calculates $PWI_i = PW_i \oplus h(ID_i \| X_i)$ and $R_i = h(ID_i \| W_i \| PWI_i)$ and checks whether $R_i \stackrel{?}{=} S_i$. Next, it selects a random integer $r_i \in Z_q^*$, computes $Z = r_i.P$, $ID_{i1} = ID_i \oplus h(X_i \| PWI_i \| W_i)$, $L_1 = h(r_i.P \| ID_i \| X_i)$ and $K_{i1} = h((TS_i \oplus r_i.P) \| r_i.P)$, $E_1 = ES_{K_{i1}}(ID_{i1} \| L_1 \| W_i \| PWI_i \| X_i)$, $C = r_i \oplus h(Z.sk_i.P \| TS_i)$ and sends $M_1 = \{E_1, TS_i, C, Z\}$ to U_j, via a public channel.
2. Once received M_1, U_j checks the timestamp TS_i, calculates $r_i = C \oplus h(Z.PK_i \| TS_i)$, $K_{j1} = h((TS_i \oplus r_i.P) \| r_i.P)$ and $DS_{K_{j1}}(E_1) = (ID_{i1} \| L_1 \| W_i \| PWI_i \| X_i)$. Next, it calculates $ID_i^* = ID_{i1} \oplus h(X_i \| PWI_i \| W_i)$ and $L_1^* = h(r_i.P \| ID_i^* \| X_i)$ and verifies whether $L_1^* \stackrel{?}{=} L_1$ to authenticate U_i. Assuming that U_i has been authenticated, U_j chooses a random integer $r_j \in Z_q^*$, calculates $L_2 = h(ID_i^* \| ID_j \| W_i \| W_j)$, $MAC_j = h(ID_i^* \| ID_j \| X_i \| X_j \| W_i \| W_j \| TS_j)$, $SK_{ji} = h(ID_i \| ID_j \| L_2 \| MAC_j \| W_i \| W_j \| r_j.r_i.P \| TS_j)$, $K_{j2} = h(L_1 \| ID_i \| TS_i \| r_i)$, $ID_{j1} = ID_j \oplus h(X_j \| ID_{i1} \| L_1)$, $E_2 = ES_{K_{j2}}(ID_{j1} \| W_j \| X_j \| MAC_j \| r_j.P \| L_2)$ and sends $M_2 = \{E_2, TS_j\}$ to U_i.
3. U_i receives M_2, checks TS_j, calculates $K_{i2} = h(L_1 \| ID_i \| TS_i \| r_i)$ and $DS_{K_{i2}}(E_2) = (ID_{j1} \| W_j \| X_j \| MAC_j \| r_j.P \| L_2)$. Then, it calculates $ID_j^* = ID_{j1} \oplus h(X_j \| ID_{i1} \| L_1)$, $L_2^* = h(ID_i \| ID_j^* \| W_i \| W_j)$ and verifies whether $L_2^* \stackrel{?}{=} L_2$ to authenticate U_j. Assuming that U_j has been authenticated, U_i calculates $MAC_i = h(ID_i \| ID_j^* \| X_i \| X_j \| W_i \| W_j \| TS_j)$ and also verifies whether $MAC_i \stackrel{?}{=} MAC_j$. Next, it sets the session key as $SK_{ij} = h(ID_i \| ID_j^* \| L_2^* \| MAC_j \| W_i \| W_j \| r_j.r_i.P \| TS_j)$.

PALK also supports changing the password for the legitimate users, as follows:

1. U_i enters its identity ID_i and its password PW_i, computes $PWI_i = PW_i \oplus h(ID_i \| X_i)$ and $R_i = h(ID_i \| W_i \| PWI_i)$ and checks whether $R_i \stackrel{?}{=} S_i$.
2. If $R_i = S_i$, the user U_i sets the new password as PW_i^{new} and computes $PWI_i^{new} = PW_i^{new} \oplus h(ID_i \| X_i)$ and $R_i^{new} = h(ID_i \| W_i \| PWI_i^{new})$.
3. Finally, PWI_i and R_i are replaced by PWI_i^{new} and R_i^{new} respectively.

4 Security analysis of PALK

In this section, we will first present some arguments for the workability of PALK, which has been independently reported in [38]. Next, assuming that the protocol is operational, we demonstrate that it has critical security flaws.

U_i	$Message$	U_j
$PWI_i = PW_i \oplus h(ID_i\|X_i)$, $R_i = h(ID_i\|W_i\|PWI_i)$, $R_i \stackrel{?}{=} S_i$, $r_i \in Z_q^*$, $Z = r_i.P$, $ID_{i1} = ID_i \oplus h(X_i\|PWI_i\|W_i)$, $L_1 = h(r_i.P\|ID_i\|X_i)$, $K_{i1} = h((TS_i \oplus r_i.P)\|r_i.P)$, $E_1 = ES_{K_{i1}}(ID_{i1}\|L_1\|W_i\|PWI_i\|X_i)$, $C = r_i \oplus h(Z.sk_i.P\|TS_i)$	$\xrightarrow{M_1=\{E_1,TS_i,C,Z\}}$	Checks TS_i, $r_i = C \oplus h(Z.PK_i\|TS_i)$, $K_{j1} = h((TS_i \oplus r_i.P)\|r_i.P)$, $DS_{K_{j1}}(E_1) = (ID_{i1}\|L_1\|W_i\|PWI_i\|X_i)$, $ID_i^* = ID_{i1} \oplus h(X_i\|PWI_i\|W_i)$, $L_1^* = h(r_i.P\|ID_i^*\|X_i)$, $L_1^* \stackrel{?}{=} L_1$, $r_j \in Z_q^*$, $L_2 = h(ID_i^*\|ID_j\|W_i\|W_j)$, $MAC_j = h(ID_i^*\|ID_j\|X_i\|X_j\|W_i\|W_j\|TS_j)$, $SK_{ji} = h(ID_i\|ID_j\|L_2\|MAC_j\|W_i\|W_j\|r_j.r_i.P\|TS_j)$, $K_{j2} = h(L_1\|ID_i\|TS_i\|r_i)$, $ID_{j1} = ID_j \oplus h(X_j\|ID_{i1}\|L_1)$, $E_2 = ES_{K_{j2}}(ID_{j1}\|W_j\|X_j\|MAC_j\|r_j.P\|L_2)$
Checks TS_j, $K_{i2} = h(L_1\|ID_i\|TS_i\|r_i)$, $DS_{K_{i2}}(E2) = (ID_{j1}\|W_j\|X_j\|MAC_j\|r_j.P\|L_2)$, $ID_j^* = ID_{j1} \oplus h(X_j\|ID_{i1}\|L_1)$, $L_2^* = h(ID_i\|ID_j^*\|W_i\|W_j)$, $L_2^* \stackrel{?}{=} L_2$, $MAC_i = h(ID_i\|ID_j^*\|X_i\|X_j\|W_i\|W_j\|TS_j)$, $MAC_i \stackrel{?}{=} MAC_j$, $SK_{ij} = h(ID_i\|ID_j^*\|L_2^*\|MAC_j\|W_i\|W_j\|r_j.r_i.P\|TS_j)$.	$\xleftarrow{M_2=\{E_2,TS_j\}}$	

Fig. 3 Login and key agreement phase of PALK

4.1 On the workability of PALK

In the registration phase of PALK, U_i computes $\{PWI_i, ID_i, X_i, TS_i\}$ and sends it to TA, where in response TA computes and sends $\{W_i, B_i, S_i'\}$ to U_i and it checks the received data and stores $\{S_i^*, W_i\}$ in its database. However, later in the login and key agreement phase of the protocol, U_i needs the value of X_i which has been computed in the registration phase as $X_i = h(PW_i\|r_i\|ID_i).P$ and r_i is a random number that has been generated by U_i. Given that U_i does not keep a record of r_i and X_i, based on the protocol description, U_i cannot perform the protocol correctly due to the dependency to X_i. In addition, in the login and key agreement phase, U_i is expected to compute $C = r_i \oplus h(Z.sk_i.P\|TS_i)$, where $Z = r_i.P$. Hence U_i needs to compute $(r_i.P).(sk_i.P)$ which equals to the multiplication of two points over curves which is meaningless. These mentioned properties of the protocol prevent the proper functioning of PALK.

4.2 PALK cryptanalysis

Assuming that PALK works properly, in this section, we present several efficient attacks against PALK that rule out its usability in practice, due to the security risks of the users.

4.2.1 Lacks mutual authentication

Through its computations of M_1, U_i does not use any information related to U_j. Hence, there will not be any difference between U_j and any other user, e.g. U_f. Therefore, it is not possible for the target U_j to identify that it should share a session key with U_i. It also means that U_j has no advantage to other parties in terms of possible access to the sent information. Therefore, the adversary will be able to achieve what U_j is able to. This fact also is used in the rest of the presented attacks.

4.2.2 Vulnerable to the identity retrieval, password retrieval, and session key retrieval

In the login and key agreement phase of PALK, where the messages are transferred over the public channel, let's assume that a passive adversary $\mathcal{A}$ eavesdrops $M_1 = \{E_1, TS_i, C, Z\}$, where TS_i is the timestamp and:

$$\begin{aligned} E_1 &= ES_{K_{i1}}(ID_{i1}\|L_1\|W_i\|PWI_i\|X_i) \\ C &= r_i \oplus h(Z.sk_i.P\|TS_i) \\ Z &= r_i.P \\ PWI_i &= PW_i \oplus h(ID_i\|X_i) \\ ID_{i1} &= ID_i \oplus h(X_i\|PWI_i\|W_i) \\ K_{i1} &= h((TS_i \oplus r_i.P)\|r_i.P) \\ &= h((TS_i \oplus Z)\|Z) \end{aligned}$$

Given that Z and TS_i are known to $\mathcal{A}$, it can recalculate the value of $K_{i1} = h((TS_i \oplus Z)\|Z)$, decrypt E_1 using it and extracts $(ID_{i1}\|L_1\|W_i\|PWI_i\|X_i)$. Given $\{ID_{i1}, W_i, PWI_i, X_i\}$ from the encryption of E_1, $\mathcal{A}$ extracts the permanent identifier (ID_i) and password (PW_i) of U_i respectively as $ID_i = ID_{i1} \oplus h(X_i\|PWI_i\|W_i)$ and $PW_i = PWI_i \oplus h(ID_i\|X_i)$. Next, assume that $\mathcal{A}$ also eavesdrops $M_2 = \{E_2, TS_j\}$, where TS_j is the timestamp and:

$$\begin{aligned} E_2 &= ES_{K_{j2}}(ID_{j1}\|W_j\|X_j\|MAC_j\|r_j.P\|L_2) \\ ID_{j1} &= ID_j \oplus h(X_j\|ID_{i1}\|L_1) \\ K_{j2} &= h(L_1\|ID_i\|TS_i\|r_i) \\ MAC_j &= h(ID_i^*\|ID_j\|X_i\|X_j\|W_i\|W_j\|TS_j) \end{aligned}$$

In addition, $sk_i.P = PK_i$ is the public key of U_i and it is known by anyone, including the adversary. Given C, Z, TS_i from M_1 and PK_i, $\mathcal{A}$ can extract $r_i = C \oplus h(Z.PK_i\|TS_i)$. Now, given $\{L_1, TS_i\}$ from M_1 and the extracted values of ID_i and r_i, $\mathcal{A}$ can recompute $K_{j2} = h(L_1\|ID_i\|TS_i\|r_i)$, which is enough to decrypt E_2 from M_2 and extract $(ID_{j1}\|W_j\|X_j\|MAC_j\|r_j.P\|L_2)$. Given $\{ID_{j1}, X_j\}$ from the decrypted E_2 and $\{L_1, ID_{i1}\}$ from the decrypted E_1, $\mathcal{A}$ extracts the permanent identifier (ID_j) of U_j as $ID_j = ID_{j1} \oplus h(X_j\|ID_{i1}\|L_1)$. The adversary extracts r_i and also obtains $r_j.P$ from decryption of E_2, so it can compute $r_i.r_j.P$. On the other hand, the shared session key is computed as follows:

$$SK_{ij} = h(ID_i\|ID_j^*\|L_2^*\|MAC_j\|W_i\|W_j\|r_j.r_i.P\|TS_j)$$

where $\mathcal{A}$ has all required information to extract it. Hence, a passive attacker can extract the session key and decrypt any transferred message which is encrypted using this session key.

4.2.3 User impersonation

Assuming that $\mathcal{A}$ has already did the secret disclosure attack of Sect. 4.2.2 against U_i and extracted $\{ID_i, PW_i, ID_{i1}, W_i, PWI_i, X_i\}$, to impersonate U_i toward any other user U_f, it does as follows:

1. $\mathcal{A}$ selects a random integer $r_a \in Z_q^*$ and proper timestamp TS_a, calculates $Z = r_a.P$, $L_1 = h(r_a.P\|ID_i\|X_i)$, $K_{a1} = h((TS_a \oplus r_a.P)\|r_a.P)$, $E_1 = ES_{K_{a1}}(ID_{i1}\|L_1\|W_i\|PWI_i\|X_i)$, $C = r_a \oplus h(Z.PK_i\|TS_a)$ and sends $M_1 = \{E_1, TS_a, C, Z\}$ to U_f, via a public channel.
2. Once received M_1, U_f checks the timestamp TS_a, computes $r_a = C \oplus h(Z.PK_i\|TS_a)$, $K_{a1} = h((TS_a \oplus r_a.P)\|r_a.P)$ and $DS_{K_{a1}}(E_1) = (ID_{i1}\|L_1\|W_i\|PWI_i\|X_i)$. Next, it calculates $ID_i^* = ID_{f1} \oplus h(X_i\|PWI_i\|W_i)$ and $L_1^* = h(r_a.P\|ID_i^*\|X_i)$ and verifies whether $L_1^* \stackrel{?}{=} L_1$ and authenticates $\mathcal{A}$ as the legitimate U_i. Thereafter, U_f chooses its timestamp TS_f and a random integer $r_f \in Z_q^*$, calculates $L_2 = h(ID_i^*\|ID_f\|W_i\|W_f)$, $MAC_f = h(ID_i^*\|ID_f\|X_i\|X_f\|W_i\|W_f\|TS_f)$, $SK_{fa} = h(ID_i\|ID_f\|L_2\|MAC_f\|W_i\|W_f\|r_f.r_a.P\|TS_f)$, $K_{f2} = h(L_1\|ID_i\|TS_a\|r_a)$, $ID_{f1} = ID_f \oplus h(X_f\|ID_{i1}\|L_1)$ and $E_2 = ES_{K_{f2}}(ID_{f1}\|W_f\|X_f\|MAC_f\|r_f.P\|L_2)$ and sends $M_2 = \{E_2, TS_f\}$ to U_i which is impersonated by $\mathcal{A}$.
3. $\mathcal{A}$ receives M_2, checks TS_f, calculates $K_{a2} = h(L_1\|ID_i\|TS_a\|r_a)$ and $DS_{K_{a2}}(E_2) = (ID_{f1}, W_f, X_f, MAC_f, r_f.P, L_2)$. Then, it computes $ID_f^* = ID_{f1} \oplus h(X_f\|ID_{i1}\|L_1)$, $L_2^* = h(ID_i\|ID_f^*\|W_i\|W_f)$ and verifies whether $L_2^* \stackrel{?}{=} L_2$ to authenticate U_f. Assuming that U_f has been authenticated, $\mathcal{A}$ computes $MAC_i = h(ID_i\|ID_f^*\|X_i\|X_f\|W_i\|W_f\|TS_f)$ and also checks whether $MAC_i \stackrel{?}{=} MAC_f$. Next, it calculates the session key as $SK_{af} = h(ID_i\|ID_f^*\|L_2^*\|MAC_f\|W_i\|W_f\|r_f.r_a.P\|TS_f)$.

It should be noted, the extracted information from U_j is also enough to impersonate U_j, whenever a user tries to communicate with U_j. The mentioned attack to impersonate U_j should be enough clear and we skip the details for this case.

4.2.4 User to user attacks on privacy, anonymity, and access rights

In any secure protocol, when U_i is communicating with U_j, the user U_j should not be able to achieve any information from U_i that can be used to compromise the privacy of U_i in any later session. However, in PALK, after a session between U_i and U_j, initiated by U_i, the user U_j achieves the set $\{ID_i, PW_i, ID_{i1}, W_i, PWI_i, X_i\}$ related to U_i and it also has PK_i. This information compromises the privacy and anonymity of U_i to U_j, in any later session in which U_i is involved. Besides, U_j has enough information to impersonate U_i in any later session. The process will be similar to the impersonation attack which has been described in Sect. 4.2.3.

4.2.5 Desynchronization attack

As a feature, PALK supports the changing of the password for the legitimate users. However, it can be a source of a desynchronization because any passive adversary who follows the attack given in Sect. 4.2.2 or any malicious user who is described in Sect. 4.2.4 (we denote both by $\mathcal{A}$ for simplicity) has access to ID_i and PW_i of U_i. Given this information, $\mathcal{A}$ changes the password as follows to desynchronize U_i:

1. $\mathcal{A}$ enters the extracted ID_i and PW_i, the device computes $PWI_i = PW_i \oplus h(ID_i \| X_i)$ and $R_i = h(ID_i \| W_i \| PWI_i)$ and checks whether $R_i \stackrel{?}{=} S_i$.
2. Because $R_i = S_i$, so $\mathcal{A}$ is authenticated as U_i. Next, $\mathcal{A}$ sets the new password of its choice as PW_a^{new} and the device computes $PWI_a^{new} = PW_a^{new} \oplus h(ID_i \| X_i)$ and $R_a^{new} = h(ID_i \| W_i \| PWI_a^{new})$.
3. PWI_i and R_i are replaced by PWI_a^{new} and R_a^{new} respectively.

Following these modifications, hence after, the legitimate user U_i will not be able to access the device, because its password PW_i will not be recognized by the device anymore. Hence, the user has been desynchronized.

4.2.6 Vulnerable to long-term user-traceability

It is trivial for any passive adversary who follows the attack that is given in Sect. 4.2.2 or any malicious user who is described in Sect. 4.2.4 to compromise the anonymity of U_i as far as it has not updated its password. It comes from the fact that given $\{ID_i, PW_i, ID_{i1}, W_i, PWI_i, X_i, PK_i\}$ it is possible to decrypt E_1 from the message M_1 and identify U_i based on its PW_i and ID_i. However, even after a password update, it is yet possible to compromise the anonymity of U_i, because ID_i is a constant value and will never be updated. Hence, it can be used as a source of traceability by $\mathcal{A}$.

4.2.7 Lacks message confidentiality

Given the transferred M_1 and M_2 for any desired session, between any pairs of users U_i and U_j, following the session key disclosure attack that has been described in Sect. 4.2.2, $\mathcal{A}$ can passively extract the session key SK_{ij} and decrypt any transferred message using that session key. Hence, this protocol does not provide message confidentiality.

4.2.8 Insider attack in the registration phase

The designers of PALK claimed that PALK supports *no-secure channel* in registration phase [19, P. 10, Table 4]. Let assume $\mathcal{A}$ can eavesdrop the channel between TA and U_i in the registration phase, which we can consider it as an insider attacker. At the first step of this phase, U_i chooses its password PW_i, ID_i and generates a random integer $r_i \in Z_q^*$, calculates $A_i = h(PW_i \| r_i \| ID_i), X_i = A_i.P$ and $PWI_i = PW_i \oplus h(ID_i \| X_i)$ and sends the tuple $\{PWI_i, ID_i, X_i, TS_i\}$ to TA. $\mathcal{A}$ eavesdrops the message and extracts the user's password as $PW_i = PWI_i \oplus h(ID_i \| X_i)$ which should not be possible commonly. Even if we consider the channel secure, a curious TA can extract PW_i which should not be possible in a secure protocol.

5 The improved protocol

The improved protocol, like its predecessor i.e. PALK, is an authentication and key agreement (AKA) protocol and has four phases of initialization, registration, login and key agreement, and password and identifier modification phase. The proposed protocol acts like PALK in the initialization phase. The difference between the proposed protocol and PALK is described as follows:

- In the proposed protocol, each client is equipped with a PUF through which message A_i is calculated.
- The other messages exchanged in the other three phases of the improved protocol have been modified to eliminate the weaknesses of PALK in the face of attacks presented in this paper.

5.1 System model

To deal with the security concerns of PALK, we amend it by proposing an improved protocol. Given that the attacker can corrupt a user U_i and retrieves its secrets, to avoid the attacks based on the reveal of the user's secret information, we assume that each user is equipped with a secure and reliable *PUF*. In this model, given challenges $\mathcal{C} \neq \mathcal{C}'$ then $PUF(\mathcal{C})$ and $PUF(\mathcal{C}')$ will be completely different but a PUF returns the same $PUF(\mathcal{C})$ for the same $\mathcal{C}$; even if it is tested for the same $\mathcal{C}$ again and again. In addition, different PUFs also return completely different responses for the same challenge. It is worth noting that designing such a PUF is an active research area itself and out of the scope of this paper, an interested reader can see [48–52] for the state of the art of the designing a reliable PUF and its challenges. The system level representation of the improved protocol is depicted in Fig. 4.

The adversary model, which is considered in our design, follows "Dolev-Yao (DY)" adversary model [37]. In this model, $\mathcal{A}$ can control the communications between all the protocol's parties over public channel and could interact passively or actively with them. In addition, for the case of forward secrecy we assume that the adversary can compromise a target smart meter or a service provider in off-line mode and reveal the stored information in non-volatile memory, including any stored secret. However, $\mathcal{A}$ has no access to the internal values in an active session, e.g. the inserted values by the user such as its password and identifier. In the designed protocol, we assume that each party knows the other party's public key and identifier.

5.2 Initialization phase

We keep the initialization phase of the improved protocol same as that of PALK, exclude that we assume each client is equipped with a $PUF(.)$. Hence, in this phase *TA* releases the system parameters, i.e. $\{E_q(c, d), q, P, h(.)\}$ and its public key $PK_{TA} = sk_{TA}.P$ and keeps sk_{TA} secret.

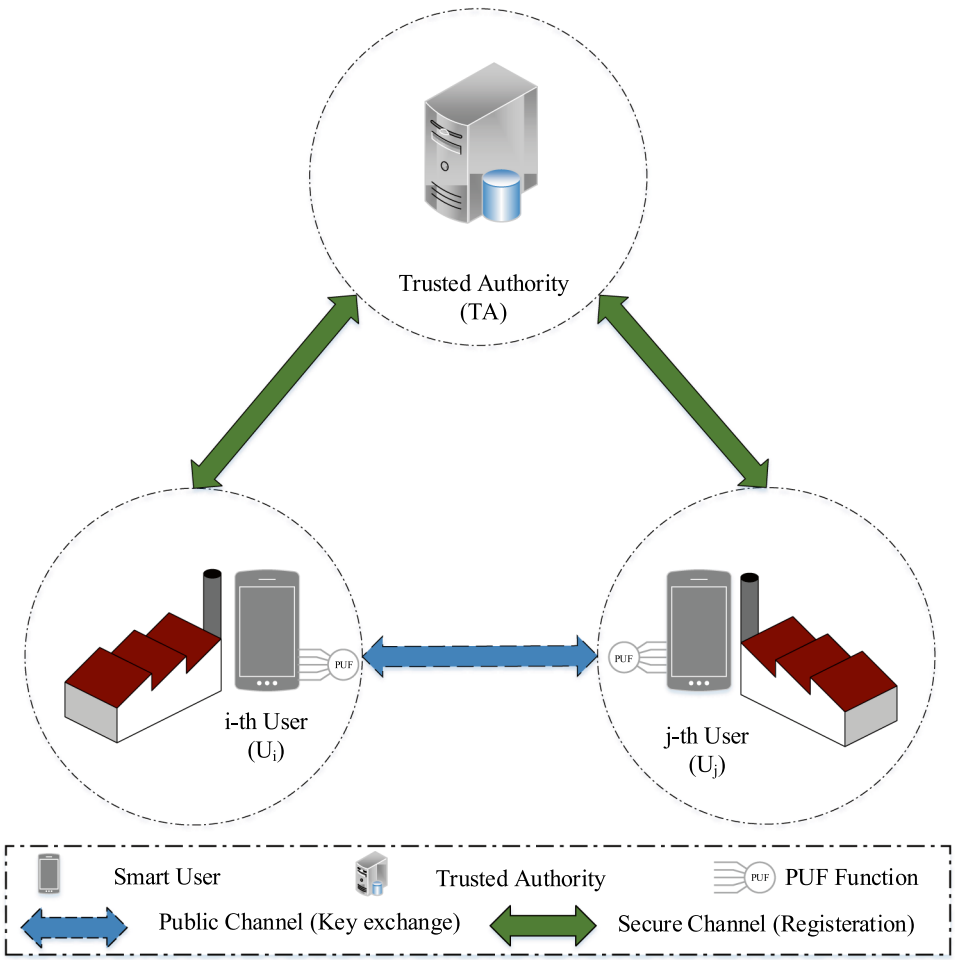

Fig. 4 System model of the improved protocol

5.3 Registration phase

The next phase of the improved protocol, following Fig. 5, is the registration phase which is run between TA and a user U_i over a secure channel as follows:

1. U_i chooses its password PW_i and its identifier ID_i, generates a random integer $s_i \in Z_q^*$, calculates $A_i = PUF(PW_i \| s_i \| ID_i)$ and $X_i = A_i.P$ and sends the tuple $\{ID_i, X_i, TS_i\}$ to TA.
2. Once received the message, TA checks the timestamp TS_i, generates a random integer $r_{TA} \in Z_q^*$, calculates $B_i = h(X_i \| r_{TA} \| ID_i)$, $Y_i = B_i.P$ and $PK_i = X_i + Y_i$ and sends the tuple $\{PK_i, B_i\}$ to U_i, over a secure channel. It also stores and distributes PK_i.
3. U_i computes $sk_i = A_i + B_i$ and verifies whether $PK_i \stackrel{?}{=} sk_i.P$. Next, it computes $D_i = h(A_i \| s_i \| B_i)$ and stores $\{s_i, B_i, D_i, PK_i\}$ in its database.

5.4 Login and key agreement phase

In the login and key agreement phase of the protocol, U_i initiates a session with U_j to establish a session key. In this model, U_j should has already been turned on and verified its password and identifier. More precisely, we assume that U_j has already been turned on, verified the user identity ID_j and its password PW_j by computing $A_j = PUF(PW_j \| s_j \| ID_j)$ and checking $h(A_j \| s_j \| B_j) \stackrel{?}{=} D_j$ to accept the login and compute $sk_j = A_j + B_j$ and store sk_j in volatile memory. Then, the process of this phase is as follows, see Fig. 6:

1. U_i enters its identity ID_i and its password PW_i, calculates $A_i = PUF(PW_i \| s_i \| ID_i)$ and checks $h(A_i \| s_i \| B_i) \stackrel{?}{=} D_i$ to accept the login and compute $sk_i = A_i + B_i$. Next, it selects a random integer $r_i \in Z_q^*$, computes $Z_i = r_i.PK_i$, $W_i = r_i.sk_i.PK_j$, $K_{ij} = h(W_i \| TS_i)$, $E_i = ES_{K_{ij}}(ID_i \| ID_j \| r_i)$ and $L_i = h(Z_i \| E_i \| K_{ij} \| ID_i \| ID_j \| TS_i)$ and sends $M_1 = \{L_i, E_i, Z_i, TS_i\}$ to U_j, via a public channel.
2. Once received M_1, U_j checks the timestamp TS_i, computes $W_i^* = sk_j.Z_i$ and $K_{ij}^* = h(W_i^* \| TS_i)$, extracts $(ID_i^* \| ID_j^* \| r_i^*) = DS_{K_{ij}^*}(E_i)$, verifies $r_i.sk_j.PK_i \stackrel{?}{=} W_i^*$ and $L_i = h(Z_i \| E_i \| K_{ij}^* \| ID_i^* \| ID_j^* \| TS_i)$ and $ID_j^* \stackrel{?}{=} ID_j$ to accept the U_i request. Then, U_j chooses a random integer $r_j \in Z_q^*$, computes $Z_j = r_j.PK_j$, $W_j = r_j.W_i^*$ and $L_j = h(Z_j \| W_j \| ID_j \| ID_i^* \| TS_j)$ and sends $M_2 = \{L_j, Z_j, TS_j\}$ to U_i, via a public channel. It also computes the session key as $SK_{ji} = h(ID_i^* \| ID_j \| W_i^* \| W_j \| TS_i \| TS_j)$.
3. Once received M_2, U_i checks the timestamp TS_j, computes $W_j^* = r_i.sk_i.Z_j$ and verifies $L_j \stackrel{?}{=} h(Z_j \| W_j^* \| ID_j \| ID_i^* \| TS_j)$ to authenticate U_j and compute the session key as $SK_{ij} = h(ID_i \| ID_j \| W_i \| W_j^* \| TS_i \| TS_j)$.

5.5 Password and identifier modification phase

The improved protocol supports changing the password and also identifier for the legitimate users, as follows:

1. U_i enters its identity ID_i and its password PW_i, calculates $A_i = PUF(PW_i \| s_i \| ID_i)$ and checks $h(A_i \| s_i \| B_i) \stackrel{?}{=} D_i$ to accept the login and compute $sk_i = A_i + B_i$. Next, it selects a random integer $r_i \in Z_q^*$, computes $Z_i = r_i.PK_i$, $W_i = r_i.sk_i.PK_{TA}$ and $K_{iTA} = h(W_i \| TS_i)$. It also selects its new password PW_i^{new} and its new identifier ID_i^{new}, generates a random integer $s_i^{new} \in Z_q^*$, calculates $A_i^{new} = PUF(PW_i^{new} \| s_i^{new} \| ID_i^{new})$ and $X_i^{new} = A_i^{new}.P$. It then computes $E_i = ES_{K_{iTA}}(ID_i \| ID_i^{new} \| ID_{TA} \| X_i^{new} \| TS_i)$ and $L_i = h(PK_i \| Z_i \| E_i \| K_{iTA} \| ID_i \| ID_i^{new} \| ID_{TA} \| X_i^{new} \| TS_i)$ and sends $M_1 = \{L_i, E_i, Z_i, TS_i\}$ to TA, via a public channel.
2. Once received M_1, TA checks the timestamp TS_i, computes $W_i^* = sk_{TA}.Z_i$ and $K_{iTA}^* = h(W_i^* \| TS_i)$, extracts $(ID_i \| ID_i^{new} \| ID_{TA} \| X_{i,}^{new} \| TS_i) = DS_{K_{iTA}^*}(E_i)$, verifies $sk_{TA}.Z_i \stackrel{?}{=} W_i^*$ and $L_i \stackrel{?}{=} h(PK_i \| Z_i \| E_i \| K_{iTA} \| ID_i \| ID_i^{new} \| ID_{TA} \| X_i^{new} \| TS_i)$, $ID_{TA}^* \stackrel{?}{=} ID_{TA}$ and compares ID_i^* with the

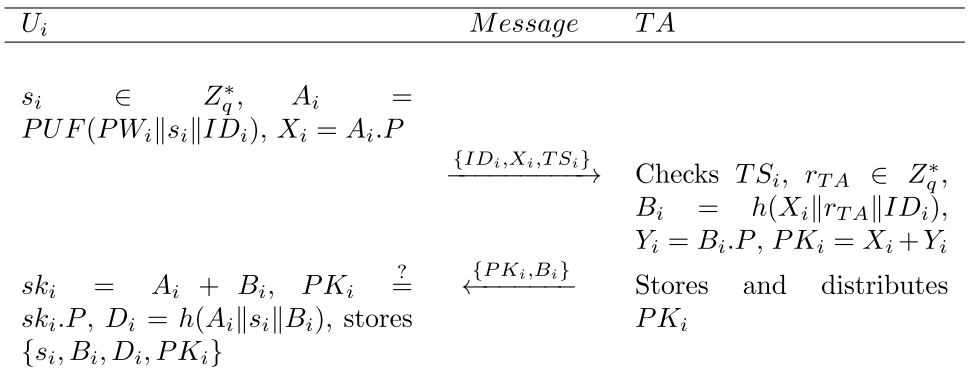

Fig. 5 Registration phase of the improved protocol over the secure channel

U_i	$Message$	U_j
$A_i = PUF(PW_i\|\|s_i\|\|ID_i)$, $h(A_i\|\|s_i\|\|B_i) \stackrel{?}{=} D_i$, $sk_i = A_i+B_i$, $r_i \in Z_q^*$, $Z_i = r_i.PK_i$, $W_i = r_i.sk_i.PK_j$, $K_{ij} = h(W_i\|\|TS_i)$, $E_i = ES_{K_{ij}}(ID_i\|\|ID_j\|\|r_i)$, $L_i = h(Z_i\|\|E_i\|\|K_{ij}\|\|ID_i\|\|ID_j\|\|TS_i)$	$\xrightarrow{M_1=\{L_i,E_i,Z_i,TS_i\}}$	Checks TS_i, $W_i^* = sk_j.Z_i$ and $K_{ij}^* = h(W_i^*\|\|TS_i)$, $(ID_i\|\|ID_j\|\|r_i) = DS_{K_{ij}^*}(E_i)$, $r_i.sk_j.PK_i \stackrel{?}{=} W_i^*$, $L_i = h(Z_i\|\|E_i\|\|K_{ij}^*\|\|ID_i^*\|\|ID_j^*\|\|TS_i)$, $ID_j^* \stackrel{?}{=} ID_j$, $r_j \in Z_q^*$, $Z_j = r_j.PK_j$, $W_j = r_j.W_i^*$, $L_j = h(Z_j\|\|W_j\|\|ID_j\|\|ID_i^*\|\|TS_j)$
Checks TS_j, $W_j^* = r_i.sk_i.Z_j$, $L_j \stackrel{?}{=} h(Z_j\|\|W_j^*\|\|ID_j\|\|ID_i\|\|TS_j)$	$\xleftarrow{M_2=\{L_j,Z_j,TS_j\}}$	
$SK_{ij} = h(ID_i\|\|ID_j\|\|W_i\|\|W_j^*\|\|TS_i\|\|TS_j)$		$SK_{ji} = h(ID_i^*\|\|ID_j\|\|W_i^*\|\|W_j\|\|TS_i\|\|TS_j)$

Fig. 6 Login and key agreement phase of the improved protocol

stored ID_i to accept the U_i password/identifier change request. Then, TA generates a random integer $r_{TA} \in Z_q^*$, calculates $B_i^{new} = h(X_i^{new}\|r_{TA}\|ID_i^{new})$, $Y_i^{new} = B_i^{new}.P$, $PK_i^{new} = X_i^{new} + Y_i^{new}$, $E_{TA} = ES_{(1\oplus K_{iTA}^*)}(PK_i^{new}\|B_i^{new}\|ID_i^{new}\|X_i^{new}\|TS_{TA})$ and sends $M_2 = \{E_{TA}, TS_{TA}\}$ to U_i. It also stores and distributes (ID_i^{new}, PK_i^{new}). However, it keeps (ID_i, PK_i) in its database for avoiding desynchronization, due to the possible block of M_2 by the adversary.

3. U_i verifies TS_{TA}, extracts $(PK_i^{new}\|B_i^{new}\|ID_i^{new}\|X_i^{new}\|TS_{TA}) = DS_{(1\oplus K_{iTA}^*)}(E_{TA})$ and verify its integrity based on $\{ID_i^{new}, X_i^{new}, TS_{TA}\}$. Next, it computes $sk_i^{new} = A_i^{new}+B_i^{new}$ and verifies whether $PK_i^{new} \stackrel{?}{=} sk_i^{new}.P$, computes $D_i^{new} = h(A_i^{new}\|s_i^{new}\|B_i^{new})$ and replaces $\{s_i, B_i, D_i, PK_i\}$ by $\{s_i^{new}, B_i^{new}, D_i^{new}, PK_i^{new}\}$ in its database. If this steps fails, U_i re-initiates this phase of protocol.

The process of this phase of the improved protocol is also depicted in Fig. 7.

6 Security analysis of the improved protocol

Given that any device is registered over a secure channel once a while but it shares session key many times, we only evaluate the security of the key agreement phase of the improved protocol in this section.

6.1 Heuristic security analysis

In this section, we show that the improved protocol provides desired security against attacks in the context. A summary of the security comparison of the improved protocol versus PALK is provided in Table 2.

6.1.1 Mutual authentication

In the proposed protocol, U_j receives $M_1 = \{L_i, E_i, Z_i, TS_i\}$, calculates $W_i^* = sk_j.Z_i$ and $K_{ij}^* = h(W_i^*\|TS_i)$, extracts $(ID_i^*\|ID_j^*\|r_i^*) = DS_{K_{ij}^*}(E_i)$, verifies $r_i^*.sk_j.PK_i \stackrel{?}{=} W_i^*$ and

U_i	$Message$	TA
$A_i = PUF(PW_i \| s_i \| ID_i)$, $h(A_i \| s_i \| B_i) \stackrel{?}{=} D_i$, $sk_i = A_i + B_i$, $r_i \in Z_q^*$, $Z_i = r_i.PK_i$, $W_i = r_i.sk_i.PK_{TA}$, $K_{iTA} = h(W_i \| TS_i)$, selects PW_i^{new}, ID_i^{new}, $s_i^{new} \in Z_q^*$, $A_i^{new} = PUF(PW_i^{new} \| s_i^{new} \| ID_i^{new})$, $X_i^{new} = A_i^{new}.P$, $E_i = ES_{K_{iTA}}(ID_i \| ID_i^{new} \| ID_{TA} \| X_i^{new} \| TS_i)$, $L_i = h(PK_i \| Z_i \| E_i \| K_{iTA} \| ID_i \| ID_i^{new} \| ID_{TA} \| X_i^{new} \| TS_i)$ and generates $M_1 = \{L_i, E_i, Z_i, TS_i\}$	$\xrightarrow{M_1}$	Checks TS_i, $W_i^* = sk_{TA}.Z_i$, $K_{iTA}^* = h(W_i^* \| TS_i)$, $(ID_i^* \| ID_i^{*new} \| ID_{TA}^* \| X_i^{*new} \| TS_i^*) = DS_{K_{iTA}^*}(E_i)$, $TS_i^* \stackrel{?}{=} TS_i$ $r_i.sk_{TA}.PK_i \stackrel{?}{=} W_i^*$, $L_i \stackrel{?}{=} h(PK_i \| Z_i \| E_i \| K_{iTA} \| ID_i^* \| ID_i^{*new} \| ID_{TA} \| X_i^{*new} \| TS_i)$, $ID_{TA}^* \stackrel{?}{=} ID_{TA}$, $ID_i^* \stackrel{?}{=} ID_i$ to accept the U_i change request. $r_{TA} \in Z_q^*$, $B_i^{new} = h(X_i^{new} \| r_{TA} \| ID_i^{new})$, $Y_i^{new} = B_i^{new}.P$,$PK_i^{new} = X_i^{new} + Y_i^{new}$, $E_{TA} = ES_{(1 \oplus K_{iTA}^*)}(PK_i^{new} \| B_i^{new} \| ID_i^{new} \| X_i^{new} \| TS_{TA})$, stores PK_i^{new}, and (ID_i, PK_i) and generates $M_2 = \{E_{TA}, TS_{TA}\}$
Verifies TS_{TA}, $(PK_i^{new} \| B_i^{new} \| ID_i^{new} \| X^{new} i \| TS_{TA}) = DS_{(1 \oplus K_{iTA}^*)}(E_{TA})$ and verifies it based on $\{ID_i^{new}, X_i^{new}, TS_{TA}\}$, $sk_i^{new} = A_i^{new} + B_i^{new}$, $PK_i^{new} \stackrel{?}{=} sk_i^{new}.P$. $D_i^{new} = h(A_i^{new} \| s_i^{new} \| B_i^{new})$, replaces $\{s_i, B_i, D_i, PK_i\}$ by $\{s_i^{new}, B_i^{new}, D_i^{new}, PK_i^{new}\}$.	$\xleftarrow{M_2\}}$	

Fig. 7 Password and identifier change phase of the improved protocol

$L_i \stackrel{?}{=} h(Z_i \| E_i \| K_{ij}^* \| ID_i^* \| ID_j^* \| TS_i)$ and $ID_j^* \stackrel{?}{=} ID_j$ to authenticate U_i, where $Z_i = r_i.PK_i$ and $L_i = h(Z_i \| E_i \| K_{ij} \| ID_i \| ID_j \| TS_i)$, $E_i = ES_{K_{ij}}(ID_i \| ID_j \| r_i)$, $W_i = r_i.sk_i.PK_j$ and $K_{ij} = h(W_i \| TS_i)$. Following the calculations below, U_j successfully authenticates the legitimate U_i:

$$\begin{aligned} sk_j.Z_i &= sk_j.r_i.PK_i \\ &= sk_j.r_i.sk_i.P \\ &= r_i.sk_i.PK_j \\ &= W_i \end{aligned}$$

Table 2 Security comparison, where MA, SKA, RA, ImA, TA, SDA, DOS, FS, FSC, InA, PG, U2UP, MIMA and DC respectively denote mutual authentication, secret key agreement, replay attack resistance, impersonation attack resistance, traceability attack resistance, secret disclosure attack resistance, desynchronization attack resistance, forward secrecy resistance, forward secrecy with compromise device resistance, insider attack resistance, password guessing resistance, user to user privacy, man in the middle attack resistance and data confidentiality. We omitted the functionality issues of PALK in this table

Protocol	[18]	[30]	[20]	PALK [19]	iPALK [38]	Ours
MA	✓	✓	✓	×	✓	✓
SKA	✓	✓	✓	✓	✓	✓
RA	✓	✓	✓	✓	✓	✓
ImA	✓	✓	✓	×	✓	✓
TA	✓	✓	✓	×	✓	✓
SDA	✓	✓	✓	×	✓	✓
DOS	✓	✓	✓	×	✓	✓
FS	✓	✓	✓	×	✓	✓
FSC	✓	✓	✓	×	✓	✓
InA	✓	✓	✓	×	✓	✓
PG	✓	✓	✓	×	✓	✓
U2UP	✓	✓	✓	×	✓	✓
MIMA	✓	✓	✓	×	✓	✓
DC	✓	✓	✓	×	✓	✓

On the other hand, U_i receives $M_2 = \{L_j, Z_j, TS_j\}$, calculates $W_j^* = r_i.sk_i.Z_j$ and verifies $L_j \stackrel{?}{=} h(Z_j \| W_j^* \| ID_j^* \| ID_i \| TS_j)$ to authenticate U_j, where $Z_j = r_j.PK_j$, $W_j = r_j.W_i$ and $L_j = h(Z_j \| W_j \| ID_j \| ID_i \| TS_j)$. Following the calculations below, U_i successfully authenticates the legitimate U_j:

$$\begin{aligned} r_i.sk_i.Z_j &= r_i.sk_i.r_j.PK_j \\ &= r_j.r_i.sk_i.PK_j \\ &= r_j.W_i \\ &= W_j \end{aligned}$$

Hence legitimate entities in this protocol are successfully authenticated. It should be noted it is not feasible for any other party which has no access to sk_i or sk_j to perform this authentication.

6.1.2 Session key agreement

The session key is calculated as $SK_{ij} = h(ID_i \| ID_j \| W_i \| W_j \| TS_i \| TS_j)$. Both U_i and U_j has access to ID_i, ID_j, TS_i and TS_j. In addition, based on the argument provided in Sect. 6.1.1, U_i can successfully calculate W_j and U_j can successfully calculate W_i. Hence, they both drive an identical value for the session key.

6.1.3 Replay attack

Any session is refreshed by the timestamps TS_i and TS_j which are verified respectively by U_i and U_j. In addition, the integrity of the timestamps is guaranteed by using one-way hash functions. Hence, the adversary cannot replay a message from a session in a later session, without been detected.

6.1.4 Impersonation attack

To impersonate U_i, the adversary should either do a replay attack or generate a valid M_1. However, following Sect. 6.1.3, replay attack is not feasible and the adversary also has no chance to generate a valid M_1, because it has no access to sk_i. The same argument can be deduced for the impersonation of U_j. Hence, the improved protocol is secure against impersonation attacks.

6.1.5 Traceability and anonymity

In the proposed protocol, the exchanged messages are M_1 and M_2. In these messages, exclude TS_i and TS_j that are the timestamps and cannot be connected to any identity to trace or compromise its anonymity, the rest of the information are encrypted values or the output of the one-way hash function and from a session to another session are randomized by fresh nonce values. Hence, the exchanged messages do not reveal any information to trace U_i or U_j or compromise their anonymity.

6.1.6 Secret disclosure attack

Transferred messages over public channel, in the improved protocol, are as follows, exclude timestamps:

$Z_i = r_i.PK_i$

$E_i = ES_{K_{ij}}(ID_i \| ID_j \| r_i)$

$L_i = h(Z_i \| E_i \| K_{ij} \| ID_i \| ID_j \| TS_i)$

$Z_j = r_j.PK_j$

$L_j = h(Z_j \| W_j \| ID_j \| ID_i \| TS_j)$

$\mathcal{A}$ is not able to retrieve any information from L_i and L_j, because they are produced by one-way hash function and are randomized each session. Z_i and Z_j are each the multiplication of a random value in a point of a large group over ECC and extracting any information from them equals to dealing with ECDLP or EC-CDHP, which is computationally infeasible. E_i is also the symmetric encryption of a message with a randomized key that is randomized by W_i and hash function. Hence, $\mathcal{A}$ cannot extract any secret information from the transferred messages over the secure channel in polynomial time.

6.1.7 Desynchronization attack

In the improved protocol, neither of U_i or U_j updates their shared values. Hence, the adversary cannot desynchronize them by forcing them to update the shared values to different values which is the source of desynchronization in some protocols [53]. The only way to desynchronize could be running a successful password and identifier change phase between U_i and TA, for which $\mathcal{A}$ needs the information of PW_i and ID_i or correctly guessing $A_i = PUF(PW_i \| s_i \| ID_i)$. Hence, the proposed protocol is secure against desynchronization attack. It should be noted $\mathcal{A}$ always can block M_2, where U_i will not set the session key or update its (ID_i^{new}, PK_i^{new}) while U_j or TA did. However, it is not a permanent desynchronization attack and U_i is expected to re-initiate the session (login and key agreement with U_j or password and identifier change with TA).

6.1.8 Provides message confidentiality

The shared session key in the improved protocol is computed as follows:

$SK_{ij} = h(ID_i \| ID_j \| W_i \| W_j \| TS_i \| TS_j)$

where, $W_i = r_i.sk_i.PK_j$ and $W_j = r_j.r_i.sk_i.PK_j$ and r_i and r_j are random numbers respectively contributed by U_i and U_j. Given that $\mathcal{A}$ cannot extract r_i and r_j without solving ECDLP, even given the secrets of U_i (includes PW_i, ID_i) or U_j at time t, to determine the session key of any time $t' \neq t$, the adversary should solve ECDLP.

6.1.9 Message confidentiality with compromised edge device

Given that smart meters are distributed over field, it could be possible for the adversary to compromise a smart meter U_i and read the stored data in the non-volatile memory, i.e. $\{s_i, B_i, D_i, PK_i\}$, where, $X_i = A_i.P$ and $A_i = PUF(PW_i \| s_i \| ID_i)$ and $D_i = h(A_i \| s_i \| B_i)$. However, U_i is equipped with a PUF and the adversary cannot clone it. On the other hand, to impersonate U_i or extract the session key at any time t', the adversary needs $PUF(PW_i \| s_i \| ID_i)$, where the adversary cannot achieve PW_i and ID_i from the stored data, because they are masked by PUF and one-way hash function. Hence, the adversary neither can clone U_i nor impersonate it. The same argument works for U_j, given that we assumed that the adversary cannot access the values that are stored in volatile memory, i.e. sk_j in this case, it cannot impersonate U_j or contradict its message confidentiality, because it needs sk_j to pass the U_i's challenge. Hence, the improved protocol provides message confidentiality with compromised edge devices.

6.1.10 Insider attack

An insider attacker in TA, with access to its memory, could access PK_i and PK_j and their identifiers which could not be used to extract their passwords or sk_i or sk_j. Even if we assume that it also monitors the transferred messages over the secure channel, in the registration phase, it can also access B_i, B_j, X_i and X_j. None of this information helps the insider adversary to access a user's password or trace U_i or U_j, assuming that password is selected randomly. The reason comes from the fact the computation of U_i is also a factor of $A_i = PUF(PW_i \| s_i \| ID_i)$ which is not known by the adversary and the computation of U_j is performed using $A_j = PUF(PW_j \| s_j \| ID_j)$ which is also unknown to $\mathcal{A}$. Hence, the improved protocol is secure against insider attacks.

6.1.11 Password guessing

Among the transferred messages over the public channel exclude timestamps, i.e. $\{Z_i, L_i, E_i, Z_j, L_j\}$, the values of L_i, Z_i and E_i are randomized by r_i and L_j and Z_j are also randomized by r_j. On the other hand, r_i and r_j are fresh nonces that are contributed by U_i and U_j respectively and are a function of sk_i or sk_j, that are computed using PUF. Each of those messages is produced either by one-way hash function, symmetric encryption, PUF or ECC multiplication. Hence, it is not feasible for the adversary to guess the password of U_i or U_j at least without cloning PUF, which is not feasible.

6.1.12 User to user privacy

Exclude the timestamp, U_j receives $\{Z_i, L_i, E_i\}$. The content of each value is masked either by one-way hash function or a point in ECC that has been multiplied by a random number. Hence, even a malicious adversary (insider U_j) cannot extract any information that can be used to impersonate U_i in a later session, exclude ID_i which is public. Hence, the proposed attacks against PALK in Sect. 4.2.4 do not work against the improved protocol. In addition, if U_i changes its identifier and password through participating password/identifier modification phase, an insider adversary in U_j will not be able to link its current records including its new public key to its old records. Hence, after password/identifier modification, U_i could be completely anonymous even to an identical U_j.

6.1.13 A man in the middle attack

Given that $\mathcal{A}$ could not do impersonation attack, it cannot do related man in the middle attacks also. In addition, the integrity of transferred messages is guaranteed by a one-way hash function, i.e. in the computation of L_i or L_j, and its travel time is controlled by timestamp. Hence, any modification in the transferred messages or relay it for a long-distance will be detected by U_i or U_j with the high probability. Therefore, the improved protocol provides desired security against man in the middle attacks.

6.2 Formal security analysis in RoR model

In this section, following [39], we formally evaluate the security of the improved protocol in real or random model (RoR).

Theorem 1 *Let ES and h(.) be a secure symmetric cipher and a secure hash function respectively and q_{exe}, q_{send} and q_{test} respectively represent the number of queries to* Execute, Send *and* Test *oracles on the improved protocol (for simplicity denoted by IP). Then*:

$$Adv^{RoR}_{\mathcal{D},IP}(t;q_{exe};q_{test};q_{send}) \leq 4.q.\varepsilon_h + q.\varepsilon_{ES} + 2.q.\varepsilon_{ECC}$$

where ε_{ECC} denotes the maximum advantage of solving ECDLP or EC-CDHP by the adversary on each query, ε_h denotes the maximum advantage of contradicting collision resistance property of h(.) and ε_{ES} denotes the maximum advantage of contradicting the indistinguishability property of ES with adaptive chosen cipher $(IND - CCA1)$ and $q = q_{exe} + q_{test} + q_{send}$.

Proof Let clients U_i and U_j are communicating to share a session key and let $\mathcal{A}$ be an adversary against the semantic security of the improved protocol in the Real-or-Random model. To prove the theorem, we define a series of games $\mathcal{G}$, started from ideal world denoted by (IKA) and ended in real world with the improved protocol denoted by (IP). For each game $\mathcal{G}_n$, we define an event $Adv^{RoR-\mathcal{G}_n}_{\mathcal{D},\mathcal{P}}(t,R)$ corresponding to the adversary's advantage to correctly guess the hidden bit b involved in the Test queries (see Sect. 3.2).

Game $\mathcal{G}_0$. This game defines an ideal key agreement (IKA) protocol and $Adv^{RoR-\mathcal{G}0}_{\mathcal{D},IKA}(t,R) = 0$

Game $\mathcal{G}_1$. This game is identical to G_0, exclude that U_i and U_j follows the structure of the transferred messages in the improved protocol. However, all messages are selected completely random. It is clear $Adv^{RoR-\mathcal{G}0}_{\mathcal{D},IKA}(t,R) - Adv^{RoR-\mathcal{G}1}_{\mathcal{D},IKA}(t,R) = 0$

Game $\mathcal{G}_2$. This game is identical to G_1, exclude that timestamps are not random any more and follow the expected structure. Given that yet the session key is generated randomly, this modification has no impact on the adversary's advantage and $Adv^{RoR-\mathcal{G}_2}_{\mathcal{D},IKA}(t,R) - Adv^{RoR-\mathcal{G}_1}_{\mathcal{D},IKA}(t,R) = 0$.

Game $\mathcal{G}_3$. This game is identical to G_2, exclude that Z_i and Z_j are calculated using ECC point multiplication, i.e. $Z_i = r_i.P$ and $Z_j = r_j.P$. Given that r_i and r_j are fresh random numbers and are generated freshly in each session, on each query, the adversary's advantage to distinguish $\mathcal{G}_3$ from $\mathcal{G}_4$ is upper-bounded by $2.\varepsilon_{ECC}$. Therefore:

$$Adv^{RoR-\mathcal{G}_3}_{\mathcal{D},IKA}(t,R) \leq Adv^{RoR-\mathcal{G}_2}_{\mathcal{D},IKA}(t,R) + 2.q.\varepsilon_{ECC}$$

where $q = q_{exe} + q_{send} + q_{test}$.

Game $\mathcal{G}_4$. This game is identical to G_3, exclude that $E_i = ES_{K_{ij}}(ID_i \| ID_j \| r_i)$ and $K_{i,j} = h(W_i \| TS_i)$. Given that TS_i is a counter by nature and r_i is a round dependent nonce, the adversary's advantage comes from collision in K_{ij} or IND-CCA1 of ES. Therefore:

$$Adv^{RoR-\mathcal{G}_4}_{\mathcal{D},IKA}(t,R) \leq Adv^{RoR-\mathcal{G}_3}_{\mathcal{D},IKA}(t,R) + q.(\varepsilon_h + \varepsilon_{ES}).$$

Game $\mathcal{G}_5$. This game is the same as G_4, except that L_i and L_j are calculated using a real hash function, as shown below:

$$L_i = h(Z_i \| E_i \| K_{ij} \| ID_i \| ID_j \| TS_i)$$
$$L_j = h(Z_j \| W_j \| ID_j \| ID_i \| TS_j)$$

Given that input values for L_i and L_j are randomized by nonce, e.g. through calculation of Z_i and Z_j, therefore:

$$Adv^{RoR-\mathcal{G}_5}_{\mathcal{D},IKA}(t,R) \leq Adv^{RoR-\mathcal{G}_4}_{\mathcal{D},IKA}(t,R) + 2.q.\varepsilon_h.$$

Game $\mathcal{G}_6$. This game is identical to G_5, exclude that the session key is calculated using hash function as $SK_{ij} = h(ID_i \| ID_j \| W_i \| W_j \| TS_i \| TS_j)$. Given that input value for SK_{ij} is randomized by nonce, e.g. through calculation of W_i and W_j, therefore:

$$Adv^{RoR-\mathcal{G}_6}_{\mathcal{D},IKA}(t,R) \leq Adv^{RoR-\mathcal{G}_5}_{\mathcal{D},IKA}(t,R) + q.\varepsilon_h$$

where $q = q_{exe} + q_{send} + q_{test}$. On the other hand, $\mathcal{G}_6$ represents the implementation of the improved protocol (IP). Hence:

$$\begin{aligned} Adv^{RoR}_{\mathcal{D},IP}(t;q_{exe};q_{test};q_{send}) &= Adv^{RoR}_{\mathcal{D},IP}(t,R) - Adv^{RoR}_{\mathcal{D},IKA}(t,R) \\ &= Adv^{RoR-\mathcal{G}6}_{\mathcal{D},IKA}(t,R) - Adv^{RoR-\mathcal{G}_0}_{\mathcal{D},IKA}(t,R) \\ &\leq q.\varepsilon_h + 2.q.\varepsilon_h + q.(\varepsilon_h + \varepsilon_{ES}) + 2.q.\varepsilon_{ECC} \\ &= 4.q.\varepsilon_h + q.\varepsilon_{ES} + 2.q.\varepsilon_{ECC} \end{aligned}$$

which completes the proof.

7 Performance evaluation

In this section, we intend to compare the proposed protocol with its previous protocol i.e. PALK and other similar protocols in terms of security, computational and communication costs.

7.1 Simulation metrics

Through our analysis, 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 respectively. It should be noted we are considering SHA-256 but truncate its output to 160-bit, to avoid the recent security flaws of SHA-1 [54].

Energy consumption can be computed as $Ec = V.I.T_c$, where Ec is the energy consumption, I is the consumed current, V is the working voltage and Tc is the total computational time to share a session key [55].

7.2 Results

7.2.1 Computational cost analysis

Any client in PALK should support ECC, hash function and symmetric encryption/decryption. However, a client in the improved protocol should also supports PUF. Given that PUF is a lightweight function in general, we have not increased the required resources in the new design compared to PALK significantly. However, in the terms of computational complexity, U_i performs 5 calls to the hash function (T_h), a PUF invocation (T_{PUF}), a call to symmetric cipher (T_{Es}) and 3 calls to ECC point-multiplications (T_{ECC}) while U_j does 4 calls to the hash function, a call to symmetric cipher and 4 calls to ECC point-multiplication. Hence, totally, a login and key agreement phase of the improved protocol costs $9 \times T_h + T_{PUF} + 7 \times T_{ECC} + 2 \times T_{Es}$. On the other hand, based on [19], a key agreement phase of PALK costs $19 \times T_h + 4 \times T_{Es} + 8 \times T_{ECC}$. It is clear the revised protocol outperforms PALK and iPALK [38], in terms of computational complexity.

A comparison between computational complexity of the improved protocol and related protocols are presented in Table 3.

7.2.2 Experimental evaluation

For each client, we used an Arduino UNO R3 board with a microcontroller ATmega328P for testing. We achieved $T_{ECC} \approx 21ms$, $T_{2ECC} \approx 26ms$, $T_h \approx 3ms$ for SHA-256 and $T_{Es} = 3.7ms$ using the mentioned platform. We also considered the time of a PUF invocation (T_{PUFn}) equal to T_h. Based on this experiment, the execution time of a key agreement session in PALK and the improved protocol is 239.8 *ms* and 184.4 *ms*, respectively. It demonstrates that the improved protocol is significantly faster than PALK on this platform (almost 23%).

7.2.3 Communication cost analysis

Based on our parameters setting that are given in Sect. 7.1, which is also similar to the setting used in PALK, the communication cost of PALK has been reported to be 1184 bits. However, there should be a typo that led to the underestimation of the communication cost. The source of the mistake could be the considered bit length of E_1 and E_2 in M_1 and M_2 respectively. Those values are computed using symmetric encryption and their length should be at least as long as the length of the encrypted values. Hence, the bit-length of E_1 and E_2 should have been

Table 3 Cost comparison of the improved protocol and related protocols (* The proposed protocol, PALK and iPALK are computing symmetric encryption on long strings. In such cases the correct cost is based on the number of blocks that are encrypted)

Protocol	Computations	Time (ms)	Communication (bits)
[18]	$10 \times T_h + 8 \times T_{ECC}$	198	1440
[30]	$5 \times T_h + 6 \times T_{ECC} + 2 \times T_{2ECC}$	193	1632
[20]	$11 \times T_h + 6 \times T_{ECC} + 2 \times T_{2ECC}$	211	1600
PALK [19]	$19 \times T_h +^* 38 \times T_{Es} + 8 \times T_{ECC}$	365.6	2912
iPALK [38]	$14 \times T_h +^* 30 \times T_{Es} + 6 \times T_{ECC}$	279	2272
Ours	$7 \times T_h + T_{PUF} + 8 \times^* T_{Es} + 7 \times T_{ECC}$	200.6	1504

Table 4 Energy comparison of the improved protocol and related protocols

Protocol	Time (*ms*)	Energy consumption (*mJ*)
[18]	198	23.76
[30]	193	23.16
[20]	211	25.32
PALK [19]	365.6	43.87
iPALK [38]	279	33.48
Ours	**200.6**	**24.07**

considered at least $64 + 160 + 320 + 160 + 320 = 1024$ and $64 + 320 + 320 + 160 + 320 + 160 = 1344$ respectively. Following this correction, the communication overhead of PALK will be 2912 bits while for the improved protocol it is $160 + (160 + 160 + 160) + 320 + 32 = 992$ bits for M_1 and $320 + 160 + 32 = 512$ bits for M_2, totally 1504 bits. It shows that the improved protocol reduces the communication cost by a factor of 46%, compared with PALK and based on an identical setting. Based on Table 3, the improved protocol has reasonable computational time and communication cost.

7.2.4 Energy analysis

According to the ATmega328P datasheet [56], the maximum power, i.e. $(V.I)$, of the ATmega328P is less than $20mA \times 6V = 120mw$. Following that, a comparison of the improved protocol's energy consumption with other schemes is provided in Table 4. These results show that a session of the improved protocol consumes less energy than PALK and iPALK [38]. As a result, the improved protocol is a good fit for constrained environments like IoE.

8 Conclusion

In this paper, we analysed the security of a recently proposed protocol for smart grid (SG), called PALK, and pointed out its critical security issues. For instance, we have shown that a passive adversary who eavesdrops the transferred messages over a public channel and also has access to the public key of the users, can disclose the session key and also the permanent parameters of the user, e.g. its password and identifier.

To provide a secure protocol for SG, we proposed an improved protocol, as the amended version of PALK, by adding PUF function to the clients. Our security analysis has shown that the new protocol meets required security for SG applications and its cost analysis showed that it is more efficient compared to PALK, in terms of computational complexity (on Arduino UNO R3 board having microcontroller ATmega328P as the platform) and communications overheads.

The only reason to use PUF in the improved protocol was to provide security against attacks related to compromised users. Hence, if we remove PUF from the users and for example replace it by hash function, yet the proposed protocol will be secure against any considered attack in this paper, exclude attacks related to compromised nodes. The used primitives by the improved protocol in this case will be identical to PALK. However, the improved protocol will be more efficient than PALK and more secure. It shows that, while a designer tries to design a security protocol, beside selecting secure component, S/he should be very careful with the structure of the transferred messages over public channel also.

Funding Information The founding sponsors had no role in the design of the study; in the collection, analysis, or interpretation of data; in the writing of the manuscript, and in the decision to publish the results. In summary, the authors have no competing interests to declare that are relevant to the content of this article.

Declarations

Conflicts of Interests The authors declare no conflict of interest/competing interests.

References

1. Curiale M (2014) From smart grids to smart city. In: 2014 Saudi Arabia Smart Grid Conference (SASG), IEEE, pp 1–9
2. Bui N, Castellani AP, Casari P, Zorzi M (2012) The internet of energy: a web-enabled smart grid system. IEEE Network 26(4):39–45
3. Rana MM (2017) Architecture of the internet of energy network: An application to smart grid communications. IEEE Access 5:4704–4710
4. Lin C, Deng D, Liu W, Chen L (2017) Peak load shifting in the internet of energy with energy trading among end-users. IEEE Access 5:1967–1976
5. Morello R, Mukhopadhyay SC, Liu Z, Slomovitz D, Samantaray SR (2017) Advances on sensing technologies for smart cities and power grids: A review. IEEE Sens J 17(23):7596–7610
6. Photovoltaics DG, Storage E (2011) Ieee guide for smart grid interoperability of energy technology and information technology operation with the electric power system (eps), end-use applications, and loads. Institute of Electrical and Electronics Engineers, New York, NY
7. Saleh MS, Althaibani A, Esa Y, Mhandi Y, Mohamed AA (2015) Impact of clustering microgrids on their stability and resilience during blackouts. In: 2015 International Conference on Smart Grid and Clean Energy Technologies (ICSGCE), IEEE, pp 195–200
8. Caballero V, Vernet D, Zaballos A (2019) Social internet of energy - A new paradigm for demand side management. IEEE Internet Things J 6(6):9853–9867. https://doi.org/10.1109/JIOT.2019.2932508
9. Fang D, Guan X, Lin L, Peng Y, Sun D, Hassan MM (2020) Edge intelligence based economic dispatch for virtual power plant in 5g

internet of energy. Comput Commun 151:42–50. https://doi.org/10.1016/j.comcom.2019.12.021
10. Kabalci E, Kabalci Y (2019) From Smart Grid to Internet of Energy. Academic Press
11. Sakib N, Hossain E, Ahamed SI (2020) A qualitative study on the united states internet of energy: A step towards computational sustainability. IEEE Access 8:69003–69037. https://doi.org/10.1109/ACCESS.2020.2986317
12. Zhong W, Xie K, Liu Y, Yang C, Xie S, Zhang Y (2019) ADMM empowered distributed computational intelligence for internet of energy. IEEE Comput Intell Mag 14(4):42–51. https://doi.org/10.1109/MCI.2019.2937611
13. Eder-Neuhauser P, Zseby T, Fabini J, Vormayr G (2017) Cyber attack models for smart grid environments. Sustainable Energy, Grids and Networks 12:10–29
14. Peng C, Sun H, Yang M, Wang Y (2019) A survey on security communication and control for smart grids under malicious cyber attacks. IEEE Trans Systems, Man, Cybernetics: Systems 49(8):1554–1569
15. Kumar P, Lin Y, Bai G, Paverd A, Dong JS, Martin A (2019) Smart grid metering networks: A survey on security, privacy and open research issues. IEEE Commun Surv Tutor 21(3):2886–2927
16. Ghosal A, Conti M (2019) Key management systems for smart grid advanced metering infrastructure: A survey. IEEE Commun Surv Tutor 21(3):2831–2848
17. Rostampour S, Safkhani M, Bendavid Y, Bagheri N (2020) Eccbap: a secure ecc based authentication protocol for iot edge devices. Pervasive and Mobile Computing pp 1–33
18. Abbasinezhad-Mood D, Nikooghadam M (2018a) An anonymous ecc-based self-certified key distribution scheme for the smart grid. IEEE Trans Industrial Electronics 65(10):7996–8004
19. Khan AA, Kumar V, Ahmad M, Rana S, Mishra D (2020) PALK: Password-based anonymous lightweight key agreement framework for smart grid author links open overlay panel. Int J Elect Power Energy Syst 121:106121
20. Wu F, Xu L, Li X, Kumari S, Karuppiah M, Obaidat MS (2019) A lightweight and provably secure key agreement system for a smart grid with elliptic curve cryptography. IEEE Syst J 13(3):2830–2838
21. Fouda MM, Fadlullah ZM, Kato N, Lu R, Shen XS (2011) A lightweight message authentication scheme for smart grid communications. IEEE Trans Smart Grid 2(4):675–685
22. Wu D, Zhou C (2011) Fault-tolerant and scalable key management for smart grid. IEEE Trans Smart Grid 2(2):375–381
23. Xia J, Wang Y (2012) Secure key distribution for the smart grid. IEEE Trans Smart Grid 3(3):1437–1443
24. Sule R, Katti RS, Kavasseri RG (2012) A variable length fast message authentication code for secure communication in smart grids. In: 2012 IEEE Power Energy Soc Gen Meet, IEEE, pp 1–6
25. Park JH, Kim M, Kwon D (2013) Security weakness in the smart grid key distribution scheme proposed by xia and wang. IEEE Trans Smart Grid 4(3):1613–1614
26. Nicanfar H, Leung VC (2013) Multilayer consensus ecc-based password authenticated key-exchange (mcepak) protocol for smart grid system. IEEE Trans Smart Grid 4(1):253–264
27. Tsai JL, Lo NW (2015) Secure anonymous key distribution scheme for smart grid. IEEE Trans Smart Grid 7(2):906–914
28. Odelu V, Das AK, Wazid M, Conti M (2016) Provably secure authenticated key agreement scheme for smart grid. IEEE Trans Smart Grid 9(3):1900–1910
29. He D, Wang H, Khan MK, Wang L (2016a) Lightweight anonymous key distribution scheme for smart grid using elliptic curve cryptography. IET Commun 10(14):1795–1802
30. He D, Wang H, Khan MK, Wang L (2016b) Lightweight anonymous key distribution scheme for smart grid using elliptic curve cryptography. IET Commun 10(14):1795–1802
31. Mahmood K, Chaudhry SA, Naqvi H, Shon T, Ahmad HF (2016) A lightweight message authentication scheme for smart grid communications in power sector. Comput Electr Eng 52:114–124
32. Mahmood K, Chaudhry SA, Naqvi H, Kumari S, Li X, Sangaiah AK (2018) An elliptic curve cryptography based lightweight authentication scheme for smart grid communication. Futur Gener Comput Syst 81:557–565
33. Chen Y, Martínez JF, Castillejo P, López L (2017) An anonymous authentication and key establish scheme for smart grid: Fauth. Energies 10(9):1354
34. Braeken A, Kumar P, Martin A (2018) Efficient and provably secure key agreement for modern smart metering communications. Energies 11(10):2662
35. Abbasinezhad-Mood D, Nikooghadam M (2018b) Design and extensive hardware performance analysis of an efficient pairwise key generation scheme for smart grid. Int J Commun Syst 31(5):e3507
36. Li X, Wu F, Kumari S, Xu L, Sangaiah AK, Choo KKR (2019) A provably secure and anonymous message authentication scheme for smart grids. J Parallel Distributed Comput 132:242–249
37. Dolev D, Yao A (1983) On the security of public key protocols. IEEE Trans Inf Theory 29(2):198–208
38. Chaudhry SA (2021) Correcting palk: Password-based anonymous lightweight key agreement framework for smart grid. Int J Electr Power Energy Syst 125:106529
39. Abdalla M, Fouque P, Pointcheval D (2005) Password-based authenticated key exchange in the three-party setting. In: Vaudenay S (ed) Public Key Cryptography - PKC 2005, 8th International Workshop on Theory and Practice in Public Key Cryptography, Les Diablerets, Switzerland, January 23-26, 2005, Proceedings, Springer, Lecture Notes in Computer Science, vol 3386, pp 65–84
40. Alheyasat A, Torrens G, Bota SA, Alorda B (2020) Selection of SRAM cells to improve reliable PUF implementation using cell mismatch metric. In: XXXV Conference on Design of Circuits and Integrated Systems, DCIS 2020, Segovia, Spain, November 18-20, 2020, IEEE, pp 1–6, 10.1109/DCIS51330.2020.9268669. https://doi.org/10.1109/DCIS51330.2020.9268669
41. Ge W, Hu S, Huang JQ, Liu B, Zhu M (2020) FPGA implementation of a challenge pre-processing structure arbiter PUF designed for machine learning attack resistance. IEICE Electron Express 17(2):20190670. https://doi.org/10.1587/elex.16.20190670
42. Hamadeh H, Tyagi A (2021) An FPGA implementation of privacy preserving data provenance model based on PUF for secure internet of things. SN Comput Sci 2(1):65. https://doi.org/10.1007/s42979-020-00428-0
43. Kumar MA, Bhakthavatchalu R (2017) Fpga based delay puf implementation for security applications. In: 2017 International Conference on Technological Advancements in Power and Energy (TAP Energy), pp 1–6, https://doi.org/10.1109/TAPENERGY.2017.8397339
44. Masoumi M, Dehghan A (2020) Design and implementation of a ring oscillator-based physically unclonable function on field programmable gate array to enhance electronic security. Int J Electron Secur Digit Forensics 12(3):243–261. https://doi.org/10.1504/IJESDF.2020.108295
45. Soybali M, Örs SB, Saldamli G (2011) Implementation of a PUF circuit on a FPGA. In: 4th IFIP International Conference on New Technologies, Mobility and Security, NTMS 2011, Paris, France, February 7-10, 2011, IEEE, pp 1–5. https://doi.org/10.1109/NTMS.2011.5720638
46. Zalivaka SS, Ivaniuk AA, Chang C (2019) Reliable and modeling attack resistant authentication of arbiter PUF in FPGA implementation with trinary quadruple response. IEEE Trans Inf Forensics Secur 14(4):1109–1123. https://doi.org/10.1109/TIFS.2018.2870835
47. Chatterjee U, Mukhopadhyay D, Chakraborty RS (2021) 3paa: A private PUF protocol for anonymous authentication.

IEEE Trans Inf Forensics Secur 16:756–769, 10.1109/TIFS.2020.3021917. https://doi.org/10.1109/TIFS.2020.3021917
48. Alkatheiri MS, Sangi AR, Anamalamudi S (2020) Physical unclonable function (puf)-based security in internet of things (iot): Key challenges and solutions. In: Gupta BB, Pérez GM, Agrawal DP, Gupta D (eds) Handbook of Computer Networks and Cyber Security. Springer, Principles and Paradigms, pp 461–473
49. Choi K, Baek S, Heo J, Hong J (2020) A 100% stable sense-amplifier-based physically unclonable function with individually embedded non-volatile memory. IEEE Access 8:21857–21865
50. Jeon D, Baek J, Kim Y, Lee J, Kim DK, Choi B (2020) A physical unclonable function with bit error rate ×10−8 based on contact formation probability without error correction code. J Solid-State Circuits 55(3):805–816
51. Lee S, Oh M, Kang Y, Choi D (2020) Design of resistor-capacitor physically unclonable function for resource-constrained iot devices. Sensors 20(2):404
52. Sahoo DP, Mukhopadhyay D, Chakraborty RS, Nguyen PH (2018) A multiplexer-based arbiter PUF composition with enhanced reliability and security. IEEE Trans Computers 67(3):403–417
53. Safkhani M, Bagheri N (2016) Generalized Desynchronization Attack on UMAP: Application to RCIA, KMAP, SLAP and SASI$^+$ protocols. IACR Cryptol ePrint Arch 2016:905. http://eprint.iacr.org/2016/905
54. 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 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part III, Springer, Lecture Notes in Computer Science, vol 11478, pp 527–555
55. Sa PK, Kumari S, Sharma V, Sangaiah AK, Wei J, Li X (2018) A certificateless aggregate signature scheme for healthcare wireless sensor network. Sustain Comput Informatics Syst 18:80–89
56. Atmel (last accessed 2020/6/10) 8-bit avr microcontroller with 32k bytes in-system programmable flash. microchip. http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

Masoumeh Safkhani received the Ph.D. degree in electrical engineering from the Iran University of Science and Technology, in 2012, with the security analysis of RFID protocols as her major field. She is currently an Associate Professor of Computer Engineering Department, Shahid Rajaee Teacher Training University, Tehran, Iran. Her current research interests include the security analysis of lightweight and ultra-lightweight protocols, targeting constrained environments, such as RFID, the IoT, VANET, and WSN. She is the author/co-author of over 50 technical articles in information security and cryptology in major international journals and conferences.

Saru Kumari is currently an Assistant Professor with the Department of Mathematics, Chaudhary Charan Singh University, Meerut, Uttar Pradesh, India. She received her Ph.D. degree in Mathematics in 2012 from Chaudhary Charan Singh University, Meerut, UP, India. She has published more than 200 research papers in reputed International journals and conferences, including more than 180 research papers in various SCI-Indexed Journals. She is on the editorial board of more than a dozen of International Journals, of high repute, under Elsevier, Springer, Wiley and others. She has served as the Guest Editor of many special issues in many SCI Journals under IEEE, Elsevier, Springer and Wiley. She has been involved in the research community as Technical Program Committee (TPC) member or PC chair for more than a dozen of International conferences of high repute. She is also serving as a reviewer of dozens of reputed Journals including SCI-Indexed of IEEE, Elsevier, Springer, Wiley, Taylor & Francis, etc. Her current research interests include information security and applied cryptography.

Mohammad Shojafar (M'17-SM'19) is a Senior Lecturer (Associate Professor) in the Network Security and an Intel Innovator, and a Marie Curie Alumni, working in the 5G Innovation Centre (5GIC) at the University of Surrey, UK. Before joining 5GIC, he was a Senior Researcher and a Marie Curie Fellow in the SPRITZ Security and Privacy Research group at the University of Padua, Italy. Also, he was a CNIT Senior Researcher at the University of Rome Tor Vergata contributed to the 5G PPP European H2020 ``SUPERFLUIDITY'' project. Dr Mohammad was a PI of the PRISENODE project, a 275k euro Horizon 2020 Marie Curie global fellowship project in the areas of Fog/Cloud security collaborating at the University of Padua, Padua, Italy. He also was a PI on an Italian SDN security and privacy (60k euro) supported by the University of Padua in 2018 and a Co-PI on an Ecuadorian-British project on IoT and Industry 4.0 resource allocation (20k dollars) in 2020. He was contributed to some Italian projects in telecommunications like GAUChO, SAMMClouds, and SC2. He received his PhD degree from Sapienza University of Rome, Rome, Italy, in 2016 with an ``Excellent'' degree. He is an Associate Editor in IEEE Transactions on Consumer Electronics, PPNA, Cluster Computing, and IET Communications.

Sachin Kumar has been working as a professor of Computer Science and Engineering at Ajay Kumar Garg Engineering College, Ghaziabad, India. He did Ph.D. in Computer Science from CCS University Meerut, India in 2007. He has more than 19 years of teaching experience. He has guided 04 Ph.D. Students and 10 M.Tech. students. He has published 10 papers in SCI Index Journals and 15 papers in Scopus Index Journals. He has also presented the papers and delivered a talk at various conferences of repute. He is also an author of 5 books on Computer Science.