A New PUF-Based Authenticated Key Establishment Protocol for V2G Networks

Milad Seddigh*, Seyed Hamid Baghestani*, Mahdi Esfahani†
* Cyberspace Research Institute, Shahid Beheshti University, Iran
† Department of Electrical Engineering, Sharif University of Technology, Tehran, Iran

***Abstract*—Vehicle-to-grid (V2G) refers to the bidirectional communication and energy flows that allow renewable energy sources to supply supplementary electrical services between electric cars (EVs) and the power grid. Additionally, V2G lowers environmental pollution and energy issues while providing efficient charging services. A PUF-based, reliable, anonymous authentication and key establishment scheme for V2G networks was recently presented by Sungjin Yu et al. In this paper, we show that the Yu et al. protocol is vulnerable to tracking attacks and does not guarantee user anonymity. We also discovered that ephemeral secret leakage attacks can target their scheme. Additionally, we propose a new PUF-based authenticated key establishment scheme for V2G networks that is more effective than the most recent relevant scheme and is resistant to all known attacks. We prove that the presented scheme is semantically secure, and we also simulate our protocol using the Scyther tool.**

Keywords: Vehicle-to-grid, user anonymity, ephemeral, key agreement

I. Introduction

After the improvement of “5G, smart grid (SG), and electric vehicle (EV)” technology, the vehicle-to-grid (V2G) is appearing as an attractive new network paradigm that has grasped the attention of scientific and industrial communities and has aroused their interest in using it [1], [2], [3]. Besides, V2G provides efficient charging services by establishing two-way communication and electricity transmission between the power grid and electric vehicles (EVs). In addition to reducing the energy issue, V2G encourages consumers to migrate to more environmentally friendly plug-in hybrid electric vehicles (PHEVs) and electric cars (EVs), which are some of the main drivers of IoT. Power links are utilized to recharge EV batteries by taking power from the grid and returning the stored energy of EVs to the grid thanks to the V2G infrastructure. Though V2G offers certain advantages, there are still a lot of problems and difficulties that need to be resolved. As the V2G communication among vehicle users, charging stations, and utility service providers occurs in a public channel, a malicious attacker will try to forge, modify, or eavesdrop on the data sent on the public channel. Additionally, the attacker can obtain the confidential information of an authorized user by differential power analysis and some cyber attacks, including "forgery, insider, and offline password guessing attacks." Despite these cyber attacks, a hacker may alter usage information and send false energy charging information to smart devices, resulting in resource waste and making consumers pay extra for electricity they haven’t used [4], [5].

A new key agreement scheme for the vehicle-to-grid network has been presented by Yu et al.[6]. It involves three entities: the charging station (CS), the fog server (FS), the utility service provider (USP), and the electrical vehicle user (U_i). All participant registrations are handled by USP, which also creates the participant’s private credentials and parameters. An ordinary server can only process data from one vehicle at a time. Because of this, V2G requires a CS to perform parallel processing. In real-time, the FS also directs and manages the CS and vehicle. The FS sends a message to the CS to establish a connection with another FS when the cars leave the smart city. To obtain a session key and be authenticated, a user additionally communicates with CS and USP. Yu’s scheme [6] is fascinating, but we believe it to be weak because it is vulnerable to tracing attacks and does not preserve user anonymity. Furthermore, an ephemeral secret leakage attack cannot be prevented by this protocol [6].

A. Motivations

This paper’s primary objective is to illustrate and address Yu et al.[6]’s security flaws. We demonstrate the attack vectors against Yu’s scheme [6], including "tracing attacks" and "ephemeral secret leakage attacks." Despite Yu et al. attempt to provide a high-level security-supported system for V2G networks, their plan failed to adequately address key security requirements. The discovery of these attacks has motivated us to propose a novel AKE strategy based on PUF that is secure, anonymous, and able to prevent possible vulnerabilities in V2G networks.

B. *Contributions*

- We present a “New PUF-Based Authenticated Key Establishment Protocol for V2G Networks” to fix the security drawbacks of [6].
- We evaluate the security of our work using the Scyther tool under the CK-adversary model [7].
- We show our scheme has lower computation costs than other related AKE schemes.

II. Related Works

In this section, we present a comparative study of entire AKE schemes in V2G networks. Two protocol scenarios for smart grid networks were developed by Mohammadali et al. [8]: an identity-based AKE scheme and an elliptic curve cryptosystem (ECC)-based AKE scheme. These AKE protocols reduce the processing cost of the smart meter and are resistant to replay and desynchronization attacks, but they are also susceptible to masquerade, fake data injection, and MITM attacks. An asymmetric key-based AKE scheme and an ECC-based AKE scheme are the two protocol scenarios that Nicanfar and Leung [9] offered to provide scalability and security for data transfer in smart grid systems. Moreover, their technique has a high computing cost during the AKE phase and is susceptible to fake data injection attacks. A safe and lightweight AKE protocol for smart grid networks—which combine symmetric-key and public-key cryptosystems—was developed by Wu and Zhou [10]. Subsequently, Wu and Zhou’s scheme [10] was declared insecure against MITM attacks by Xia and Wang [11], who also suggested a new secure key distribution scheme for smart grid networks. Park et al. [12] subsequently demonstrated that Xia and Wang’s approach [11] is not immune to forgery attacks and user privacy leaks. Tsai and Lo [13] developed a secure key distribution strategy for V2G networks using identity-based encryption and signatures. The session key security and privacy of the smart meters are not provided by Tsai and Lo’s scheme [13], as shown by Odelu et al. [14]. Next, it was demonstrated by Gope and Sikdar [15] that the AKE scheme introduced in [14] is not resistant to DoS and MITM attacks.

To solve privacy issues for V2G networks in 2019, several AKE techniques have been proposed [8] - [16]. Unfortunately, these AKE techniques do not meet poor performance since they take advantage of computationally expensive cryptographic primitives such as sign encryption and group signature procedures. Furthermore, there is still a problem with users of electrical vehicles privacy concerns. For V2G networks, Gope and Sikdar suggested an efficient-cost privacy-preserving AKE technique [15]. Nevertheless, Gope and Sikdar’s scheme [15] was shown by Irshad et al. [17] to have a desynchronization problem during device login and to be vulnerable to key compromise impersonation vulnerability due to erroneous assumptions, which would reveal the private secret key to the attacker. In response to Gope and Sikdar’s scheme’s security vulnerabilities, Irshad et al. [17] introduced a lightweight and secure AKE scheme for V2G networks [15]. For V2G networks, Sureshkumar et al. [18] proposed a strong and high-security AKE system in 2022. Nevertheless, their AKE scheme [18] is susceptible to various dangerous security threats and does not comply with essential security criteria. Thus, it was demonstrated by Sungjin Yu et al. [6] that the Sureshkumar scheme lacks "mutual authentication" and is vulnerable to security threats such as "session key disclosure and impersonation" attacks due to improper protocol design.

A "PUF-based robust and anonymous AKE scheme for V2G networks" was recently created by Sungjin Yu to address the security issues with [18]. Regretfully, we demonstrated that their scheme is vulnerable to tracing and ephemeral secret leakage attacks.

III. PRELIMINARIES

In this section, we explain the basics required for the protocol schemes of the V2G networks.

A. *Adversary Model*

In the Dolev-Yao [23] model, an adversary with probabilistic polynomial time has complete control over the communication lines, including the capacity to read, record, remove, or alter communications transmitted across the unsecured channel. In the CK-adversary model, an attacker can obtain secret information stored in the party’s memory by explicit attacks, ensuring that the disclosure

Table I: Comparison of entire AKE schemes in V2G network

Scheme	Cryptographic Algorithms	Advantages	Flaws
Wu and Zhou [10]	*Elliptic curve cryptography *Symmetric key encryption *One-way hash function	*Provide a fault-tolerant and scalable key manage ment for V2G *Produce a high-level of fault tolerance and scalability	*Vulnerable to man-in-the-middle (MITM) attack [11] *Does not meet session key security [15]
Xia and Wang [11]	*Symmetric key encryption *One-way hash function	*Propose a secure and efficient key distribution scheme for V2G networks *create high-level security and effective efficiency *Decrease computation cost	*Cannot resist against forgery attack [12] *Does not preserve the privacy of the user [12]
Tsai and Lo [13]	*Bilinear pairing *Multiplication point *Modular exponential *One-way hash function	*Create a secure anonymous key distribution scheme for V2G networks	*Does not provide session key security [14] *Does not provide privacy of the smart meter [14]
Odelu et al. [14]	*Bilinear Maps *Identity-based encryption *One-way hash function	*Present an efficient and robust authenticated key agreement scheme for V2G networks *Provide session key security and strong credentials' privacy *Reduce computation cost	*Vulnerable to MITM attack [15] Is fragile against denial of service (DoS) attack [15]
Gope and Sikdar [15]	*One-way hash function	*Provide a secure authentica tion scheme for energy internet-based V2G *Produce lightweight computation and Communication costs	*Has a desynchronization issue during login to the device [17] *Is vulnerable to key compromise im personation attack [17]
Kaveh et al. [19]	*One-way hash function *Physical unclonable function	*Create a secure and Robust AKE scheme for SG neighborhood area networks *Provide high-level security *Low computation cost	*Is fragile against smart meter impersonation attack [20] *Vulnerable to SG server impersonation attack
Bansal et al. [21]	*One-way hash function *Physical unclonable function	*Provide a lightweight AKE protocol for V2G networks using PUF *Provide lightweight computation cost and energy efficient	*Is vulnerable to privilege insider and physical attacks [22] *Does not sure user anonymity and untraceability [22]
Sureshkumar et al. [18]	*One-way hash function	*Produce a lightweight authenticated and key agreement scheme for V2G networks *Provide high-level security *Low computation cost	*Can cause session key disclosure attack Not secured impersonation attack *Does not provide mutual authentication
Sungjin Yu et al. [6]	*One-way hash function *Physical unclonable function	*Present a robust and lightweight authenticated and key agreement scheme for V2G networks	*Not secure against tracing attacks Is vulnerable to ephemeral secret leakage attacks

of any kind of secret information held at a party has the least potential impact on the security of other secrets [7]. In the CK-adversary model, the information revealed to the attacker is divided into three categories as follows:

- Session-state reveal: Except for the long-term keys, the internal state of a session (includes ephemeral secret parameters) is revealed to the attacker.
- Session-key query: The attacker acquires the session key of a specific session.
- Party corruption: In this case, the attacker extracts all the internal memory of that party.

According to CK paper [7] in the party corruption, as the attacker has all long-term secrets of that party, it can impersonate that party from the time of corruption. In this case, nothing must be learned about the sessions within the corrupted party, which has been kept before party corruption.

B. Physical Unclonable Function

Many smart devices with limited computing power use PUF as an effective way to improve their security [24], [25]. Furthermore, PUF is a popular method for producing an output from a given input—like a fingerprint—that is retrieved from the physical microstructure of smart devices. PUF presents a major obstacle to the effective replication of an identical PUF since it does not save a private key. The characteristics of unpredictability, originality, and dependability—all essential elements for preserving the security of smart devices—are guaranteed by the ideal PUF. PUF is particularly helpful in defending against attacks including side-channel, cloning, and tampering attacks against the smart devices utilized in WMSN-based healthcare systems. The properties of the PUF can be summarized as follows:

- PUF is comparatively easy to use and evaluate.
- PUF is reliant on the system's physical microstructure.
- Any attempt to interfere with smart devices that contain PUF will cause PUF to change its behavior, which will ultimately lead to its destruction [26].

C. System Model

The system model for V2G network communication is displayed in Figure 1. "Utility service provider (USP), smart electric vehicle (SEV), cloud server (CS), and fog server (FS)" are components of the system model. Different communication levels, including "vehicle-to-charging station (V2C), vehicle-to-vehicle (V2V), and charging station-to-utility service provider (C2U)," are compatible with this concept. For V2G networks, this model presents an anonymous, lightweight, and robust AKE technique that guarantees effective and secure communication. A CS must perform parallel

Table II: Symbols

Symbol	Meaning
U_i	i^{th} *electrical vehicle user*
USP	*Utility service provider*
CS	*Charging station*
ID_U, ID_{CS}, ID_{USP}	*Identity of* U_i, *CS, and USP*
PW_i	*Password of* U_i
BIO	*Biometric of* U_i
C_U^x, R_U^x	*Challenge/response of* U_i
C_{CS}^x, R_{CS}^x	*Challenge/response of CS*
$\leq \Delta T_i$	*Acceptable time delay*
T_i	*Timestamp*
MK_{CS}, MK_{USP}	*Master key of CS and USP*
SK	*A session key among* U_i, *CS, and USP*
$E_K(\cdot)/D_K(\cdot)$	*Symmetric key encryption/decryption*
h(·)	*Hash function*
H(·)	*Bio − hash function*
$\oplus$	*XOR function*
$\parallel$	*Concatenation*

processing because a server can only process data from one car at a time. The FS in the system model monitors the vehicle and CS. The FS notifies the CS to link to another FS when the vehicles exit the smart city.

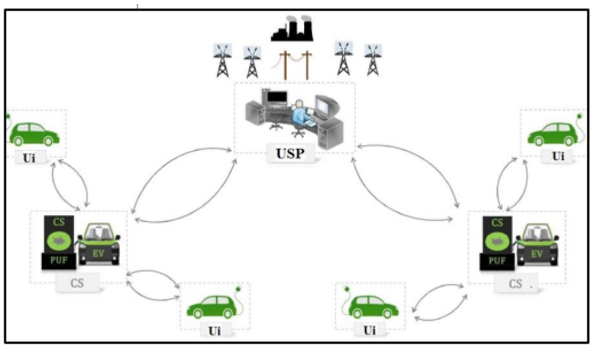


Figure 1: System Model for V2G Networks

IV. Review of the Sungjin Yu Scheme

We present the scheme reviews for Sungjin Yu et al. [6] and show that this scheme is not without security holes. This paper's symbols are represented by Table II.

A. Initial Setup Phase

In addition to selecting a master private key MK_{USP} and comprising the h(·), USP makes the h(·) publicly available.

B. Registration Phase

The U_i and CS registration phases are the two components of the registration phase, which is carried out over a secure channel.

1) Charging Station Registration Phase: CS chooses a set of (C_{CS}^x, R_{CS}^x) and an identity ID_{CS}. Subsequently, CS issues ID_{CS},(C_{CS}^x, R_{CS}^x) across a secure channel to the USP. Subsequently, USP computes $c_j = h(ID_{CS} \parallel MK_{USP} \parallel R_{CS}^x)$ and $Z_j = h(ID_{CS} \parallel ID_{USP} \parallel MK_{USP})$ before forwarding it to the CS. Finally, USP retains (C_{CS}^x, R_{CS}^x), and ID_{CS} in the database (DB) in addition to discarding Z_j and c_j. Additionally, (C_{CS}^x, R_{CS}^x), Z_j, c_j are safely stored in CS.
2) User Registration Phase: To obtain the necessary V2G services and credentials from USP, U_i registers with USP before the AKE phase. First, U_i imprints BIO and creates ID_U and PW_i. Subsequently, U_i selects a set of (C_U^x, R_U^x) and computes $RPW_i = h(PW_i \parallel BIO)$ and $RID_i = h(ID_i \parallel BIO)$, and $RID_i, RPW_i, (C_U^x, R_U^x)$ are then sent to the USP.
   After that, USP computes $X_i = h(RID_i \parallel MK_{USP} \parallel R_U^x), Q_i = X_i \oplus h(RID_i \parallel R_U^x) \oplus RPW_i$, and $W_i = h(RID_i \parallel R_U^x \parallel X_i \parallel RPW_i)$. Additionally, USP retains Q_i, W_i in the SC and transmits the SC to the U_i. Finally, USP keeps $E_i, (C_U^x, R_U^x)$ in the DB [6] and computes $E_i = X_i \oplus ID_{USP} \oplus MK_{USP}$.

C. Authentication and Key Establishment Phase

Following the registration step, a session key (SK) between U$_i$, CS, and USP must be established, and U$_i$ and USP must have mutual authentication via CS. According to Table III, this authentication key establishment (AKE) phase is imposed over an unsecured channel.

1) U_i inputs ID_U, PW_i, and imprints BIO in SC. Later on, SC calculates $RID_i = h(ID_U \parallel BIO)$, $RPW_i = h(PW_i \parallel BIO)$, $X_i = Q_i \oplus h(RID_i \parallel R_U^x) \oplus RPW_i, W_i^* = h(RID_i \parallel R_U^x \parallel X_i \parallel RPW_i)$, and verifies whether $W_i^* = W_i$. If it matches, SC accepts U_i; if not, it ends the session and rejects it. From the premise set (C_U^x, R_U^x), SC generates a random nonce R_1, a timestamp T_1, and a pair of (C_U^1, R_U^1). Subsequently, CS receives $Msg_1 = RID_i, M_1, Auth_U, C_U^1, T_1$ from SC after it has computed $M_1 = (ID_U \parallel R_1) \oplus h(X_i \parallel RID_i \parallel R_U^1 \parallel T_1)$ and $Auth_U = h(ID_U \parallel R_1 \parallel R_U^1 \parallel X_i \parallel T_1)$.
2) The freshness of $|T_2 - T_1| \leq \Delta T_i$ is verified using CS. From the premise set (C_{CS}^x, R_{CS}^x), CS constructs a R_2, a T_2, and a pair of (C_{CS}^1, R_{CS}^1) if T_1 is matched. Then, CS calculates $TK = h(Z_j \parallel R_{CS}^1)$, $M_2 = (R_2 \parallel Z_j) \oplus h(ID_{CS} \parallel R_{CS}^x \parallel c_j \parallel TK \parallel T_2)$ and

$Auth_{CS} = h(ID_{CS} \parallel R^1_{CS} \parallel R_2 \parallel Z_j \parallel T_2)$. $Msg_2 = MI_1, ID_{CS}, C^1_{CS}, T_2, C^1_U, T_1$ is sent to USP by CS after it has encrypted $MI_1 = E_{TK}(M_2, Auth_{CS}, RID_i, M_1, Auth_U)$.

3) $|T_3 - T_2| \leq \Delta T_i$ is verified by USP, and $ID^*_{CS} = ID_{CS}$ is checked. In the case that T_1 and ID_{CS} match, USP recovers the R^1_{CS} based on C^1_{CS}. It then computes $Z_j = h(ID_{CS} \parallel ID_{USP} \parallel MK_{USP} \parallel R^x_{CS}), TK = h(Z_j \parallel R^1_{CS}), c_j = h(ID_{CS} \parallel MK_{USP})$, and decrypts $(M_2, Auth_{CS}, RID_i, M_1, Auth_U) = D_{TK}(MI_1)$. Subsequently, USP computes $(R_2 \parallel Z_j) = M_2 \oplus h(ID_{CS} \parallel R^x_{CS} \parallel c_j \parallel TK \parallel T_2)$, and $Auth^*_{CS} = h(ID_{CS} \parallel R^1_{CS} \parallel R_2 \parallel Z_j \parallel T_2)$. Finally, USP confirms if $Auth^*_{CS} = Auth_{CS}$. When it matches, USP authenticates CS and then recovers the R^1_U based on C^1_U and computes $X_i = E_i \oplus ID_{USP} \oplus MK_{USP}, (ID_U \parallel R_1) = M_1 \oplus h(X_i \parallel RID_i \parallel R^1_U \parallel T_1)$, and $Auth^*_U = h(ID_U \parallel R_1 \parallel R^1_U \parallel X_i \parallel T_1)$ and verifies whether $Auth^*_U = Auth_U$. If it matches, USP authenticates U_i successfully. Later on, USP produces a R_3, T_3 and computes $M_3 = (R_1 \parallel R_3) \oplus h(TK||R^1_{CS} \parallel Z_j \parallel R_2 \parallel ID_{CS})$, $Auth_{USP-CS} = h(ID_{CS} \parallel R_2 \parallel R_3 \parallel R^1_{CS} \parallel Z_j \parallel T_3)$, $M_4 = (R_2 \parallel R_3) \oplus h(R^1_U \parallel X_i \parallel R_1 \parallel ID_U)$, and $Auth_{USP-U} = h(ID_U \parallel R_1 \parallel R_3 \parallel R^1_U \parallel X_i \parallel T_3)$, and encrypts $MI_2 = E_{(TK \parallel R_2)}(M_3, Auth_{USP-CS}, M_4, Auth_{USP-U})$. Ultimately, $Msg_3 = MI_2, T_3$ is sent to CS by USP. The freshness of $|T_4 - T_3| \leq \Delta T_i$ is verified using CS. $Auth^*_{USP-CS} = h(ID_{CS} \parallel R_2 \parallel R_3 \parallel R^1_{CS} \parallel Z_j \parallel T_3)$ is computed by CS if T_3 matches. CS decrypts $(M_3, Auth_{USP-CS}, M_4, Auth_{USP-U}) = D_{(TK \parallel R_2)}(MI_2)$, and confirms that $Auth_{USP-CS} = Auth^*_{USP-CS}$. If it equals, CS verifies the USP. $Auth_{CS-U} = h(ID_{CS} \parallel R_1 \parallel R_2 \parallel T_4)$ is computed by CS after choosing a T_4. Then, $Msg_4 = ID_{CS}, M_4, Auth_{USP}, Auth_{CS}, T_3, T_4$ is sent.

4) In order to verify that $Auth^*_{USP-U} = Auth_{USP-U}$, U_i computes $(R_2 \parallel R_3) \oplus h(R^1_U \parallel X_i \parallel R_1 \parallel ID_U), Auth^*_{USP-U} = h(ID_U \parallel R_1 \parallel R_3 \parallel R^1_U \parallel X_i \parallel T_3)$. U_i authenticates USP when it matches. Next, $Auth^*_{CS-U} = h(ID_CS \parallel R_1 \parallel R_2 \parallel T_4)$ is computed by U_i, and $Auth^*_{CS-U} = Auth_{CS-U}$ is verified. U_i successfully authenticates CS if it is equal. Consequently, U_i, CS, and USP form a common SK = h($R_1 \parallel R_2 \parallel R_3$) and are mutually authenticated.

V. Security Flaws of Sungjin Yu Scheme

We demonstrate that Sungjin Yu's suggested approach has certain intrinsic security weaknesses [6].

A. The Loss of Anonymity and Untraceability

The goal of anonymity is to prevent an attacker from obtaining the ID of the user of an electrical vehicle through message interceptions made over an unsecured communication channel. Moreover, an attacker may not even be able to figure out any connection between two distinct sessions. According to [6], without the "biometric (BIO), secret credentials (X_i), and PUF secret value R^1_U," an attacker eavesdropping in on the exchanged communications during the AKE phase cannot discover the actual ID of the electrical vehicle user.

We consider their assertion to be false and misleading. An attacker can directly recover the pseudo-identity RID by capturing messages sent via the insecure channel. Note that the pseudo-identity is transmitted by the user in the registration phase and is unchanged in distinct sessions. Thus, the pseudo-identity RID (Figure 2) can be used by the attacker to link different sessions that the user U_i creates. Although the attacker cannot retrieve ID_i from the equation $RID_i = h(ID_i \parallel BIO)$, the exposure of RID_i cannot lead to the anonymity of the user. In other words, identifier ID_i, characteristics of the electrical vehicle user, uniquely corresponds to the pseudo-identifier RID_i, and the attacker can identify the identity of the user by extracting RID_i. As a result, the attacker can link the sessions and track the user after determining the RID. In order to thwart this attack, the user's identity needs to be distinct and altered for every session.

B. Ephemeral Secret Leakage Attack

A protocol is not vulnerable to an ephemeral secret leakage attack when all random session numbers are disclosed and all of the sensitive session parameters, such as the session key (SK), remain secure. The Yu et al. [6] scheme, however, is vulnerable to an ephemeral attack. According to the CK model, the session key (SK = h($R_1 \parallel R_2 \parallel R_3$)) is still unsafe if all random session numbers, such as R_1, R_2, and R_3, are disclosed.

Table III: Summary of Authentication and Key Establishment Phase of R2AKE-V2G [6]

Electrical Vehicle User (u_i)	Charging Station (CS)	Utility Service Provider (USP)
Inputs ID_u, PW_i and imprints BIO in SC Calculates $RID_i = h(ID_i \parallel BIO)$ $RPW_i = h(PW_i \parallel BIO)$ $X_i = Q_i \oplus h(RID_i \parallel R_U^x)$ $W_i^* = h(RID_i \parallel R_U^x \parallel X_i \parallel RPW_i)$ Verifies $W_i^* = W_i$ Creates a random nonce R_1 and a timestamp T_1 Generates a pair of $(C_U^1, R_U^1) from (C_U^x, R_U^x)$ Computes $M_1 = (ID_U \parallel R_1) \oplus h(X_i \parallel RID_i \parallel R_U^1 \parallel T_1)$ $Auth_U = h(ID_U \parallel R_1 \parallel R_U^1 \parallel X_i \parallel T_1)$ $\xrightarrow{Msg_1=RID_i,M_1,Auth_U,C_U^1,T_1}$	Checks $\lvert T_2 - T_1 \rvert \leq \Delta T_i$ Selects a random nonce R_2 and a timestamp T_2 Selects a pair of $(C_{CS}^1, R_{CS}^1) from (C_{CS}^x, R_{CS}^x)$ Calculates $TK = h(Z_j \parallel R_{CS}^1)$ $M_2 = (R_2 \parallel Z_j) \oplus h(ID_{CS} \parallel R_{CS}^x \parallel c_j \parallel TK \parallel T_2)$ $Auth_{CS} = h(ID_{CS} \parallel R_{CS}^1 \parallel R_2 \parallel Z_j \parallel T_2)$ Encrypts $MI_1 = E_{TK}(M_2, Auth_{CS}, RID_i, M_1, Auth_U)$ $\xrightarrow{Msg_2=MI_1,ID_{CS},C_{CS}^1,T_2,C_U^1,T_1}$	
		Verifies $\lvert T_3 - T_2 \rvert \leq \Delta T_i$ *and checks* $ID_{CS}^* = ID_{CS}$ *Retrieves* R_{CS}^1 *on the basis of* C_{CS}^1 $Zj = h(ID_{CS} \parallel ID_{USP} \parallel MK_{USP} \parallel R_{CS}^x)$ $TK = h(Z_j \parallel R_{CS}^1)$ $c_j = h(ID_{CS} \parallel MK_{USP})$ Decrypts $(M_2, Auth_{CS}, RID_i, M_1, Auth_U) = D_{TK}(MI_1)$ Computes $(R_2 \parallel Z_j) = M_2 \oplus h(ID_{CS} \parallel R_{CS}^x \parallel c_j \parallel TK \parallel T_2)$ $Auth_{CS}^* = h(ID_{CS} \oplus R_{CS}^1 \oplus R_2 \oplus Z_j \oplus T_2)$ *Checks* $Auth_{CS}^* = Auth_{CS}$ *Retrieves the* R_U^1 *on the basis of* C_U^1 Calculates $X_i = E_i \oplus ID_{USP} \oplus MK_{USP}$ $(ID_U \parallel R_1) = M_1 \oplus h(X_i \parallel RID_i \parallel R_U^1 \parallel T_1)$ $Auth_U^* = h(ID_U \parallel R_{\parallel} R_U^1 \parallel X_i \parallel T_1)$ $Verifies\ Auth_U^* = Auth_U$ *Generates a random nonce* R_3 *and a timestamp* T_3 $M_3 = (R_1 \parallel R_3) \oplus h(TK \parallel R_{CS}^1 \parallel Z_j \parallel R_2 \parallel ID_{CS})$ $Auth_{USP-CS} = h(ID_{CS} \parallel R_2 \parallel R_3 \parallel R_{CS}^1 \parallel Z_j \parallel T_3)$ $M_4 = (R_2 \parallel R_3) \oplus h(R_U^1 \parallel X_i \parallel R_1 \parallel ID_U)$ $Auth_{USP-U} = h(ID_U \parallel R_1 \parallel R_3 \parallel R_U^1 \parallel X_i \parallel T_3)$ *Encrypts* $MI_2 = E_{(TK \parallel R_2)}(M_3, Auth_{USP-CS}, M_4, Auth_{USP-U})$ $\xleftarrow{Msg_3=MI_2,T_3}$
	Verifies $\lvert T_4 - T_3 \rvert \leq \Delta T_i$ Decrypts $(M_3, Auth_{USP-CS}, M_4, Auth_{USP-U}) = D_{(TK \parallel R_2)}(MI_2)$ Computes $Auth_{USP-CS}^* = h(ID_{CS} \parallel R_2 \parallel R_3 \parallel R_{CS}^1 \parallel Z_j \parallel T_3)$ Checks $Auth_{USP-CS}^* = Auth_{USP-CS}$ *Generates a timestamp* T_4 Calculates $Auth_{CS-U} = h(ID_{CS} \parallel R_1 \parallel R_2 \parallel T_4)$ $\xleftarrow{Msg_4=ID_{CS},M_4,Auth_{USP-U},Auth_{CS-U},T_3,T_4}$	
Computes $(R_2 \parallel R_3) \oplus h(R_U^1 \parallel X_i \parallel R_1 \parallel ID_U)$ $Auth_{USP-U}^* = h(ID_U \parallel R_1 \parallel R_3 \parallel R_U^1 \parallel X_i \parallel T_3)$ Checks $Auth_{USP-U}^* = Auth_{USP-U}$ Computes $Auth_{CS-U}^* = h(ID_{CS} \parallel R_1 \parallel R_2 \parallel T_4)$ Verifies $Auth_{CS-U}^* = Auth_{CS-U}$		
	U_i, *CS, and USP establish a common session key* $SK = h(R_1 \parallel R_2 \parallel R_3)$	

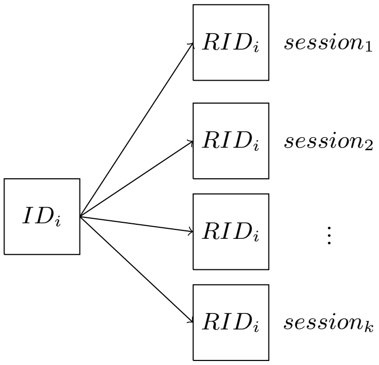


Figure 2: The false anonymity

C. Performance Flaws

Encrypting MI_1 and MI_2 with TK is a trivial operation, which can cause a rise in the computation cost. The authors of [6] thought that they could reduce communication costs by using symmetric cryptography. Since they expected the output of symmetric encryption with an input greater than 256 bits to be equal to 256 bits, this is where they misunderstood, and in symmetric cryptography, the number of bits of input is equal to the number of bits of output. So the use of symmetric cryptography in their scheme has not only been useless but has also increased the computation cost of their design. Furthermore, they have used only one hash function in the scheme, while they must utilize different hash functions with different numbers of bits. The number of bits of the parameters before and after the xor operation must be the same, but this scheme ignores this point. For this reason, their work is not efficient. Besides, in Msg_4, CS sends T_4, but U_i does not check the freshness of $|T_5 - T_4| \leq \Delta T_i$. As a consequence, the use of T_4 does not have any benefit for the [6] scheme.

VI. Our Proposed Scheme

We propose a “PUF-based robust and anonymous AKE scheme for V2G networks” to solve the security flaws of [6].

A. Initial Setup Phase

USP first selects a master private key MK_{USP} and comprises the h_1, h_2, h_3 and h_4. Following that, USP makes all hash functions available to the public.

B. Registration Phase

To access the valuable V2G services and get credentials from USP, U_i and CS must register with USP before the authentication key establishment (AKE) step in this scheme.

The CS and U_i registration phases are the two components of the registration phase that are carried out over a secure channel.

1) Charging Station Registration Phase:A set of (C^x_{CS}, R^x_{CS}) and an identity ID_{CS} are produced by CS. Afterwards, CS uses a secure channel to deliver ID_{CS},(C^x_{CS}, R^x_{CS}) to the USP. Afterwards, USP computes $c_j = h_1(ID_{CS} \parallel MK_{USP} \parallel R^x_{CS})$ and $Z_j = h_1(ID_{CS} \parallel ID_{USP} \parallel MK_{USP})$ before forwarding it to the CS. In the end, USP removes Z_j and c_j from the database (DB) while retaining $(C^x_{CS}, R^x_{CS}), ID_{CS}$. $(C^x_{CS}, R^x_{CS}), Z_j, c_j$ is similarly safely stored by CS.
2) User Registration Phase: To use the helpful V2G services, U_i registers with USP before the AKE phase and gets the necessary credentials from USP.
   First, U_i imprints BIO and creates ID_U and PW_i. Afterwards, U_i chooses a set of (C^x_U, R^x_U) and computes $RPW_i = h_1(PW_i \parallel BIO)$ and $RID_i = h_1(ID_i \parallel BIO)$. $RID_i, RPW_i, (C^x_U, R^x_U)$ is then transmitted to the USP.
   Then, USP first generates a random nonce R_r, and then calculates $y_i = h_1(MK_{USP} \parallel R_r) \oplus RID_i$, $L_i = h_1(MK_{USP} \parallel y_i) \oplus R_r$, $X_i = h_1(RID_i \parallel MK_{USP} \parallel R^x_U), Q_i = X_i \oplus h_1(RID_i \parallel R^x_U) \oplus RPW_i$, and $W_i = h_1(RID_i \parallel R^x_U \parallel X_i \parallel RPW_i)$. Afterwards, USP forwards the SC to the U_i while keeping (Q_i, W_i, y_i, L_i) in the SC. In the DB, $(E_i, (C^x_U, R^x_U), RID_i)$ is stored after USP_i computes $E_i = X_i \oplus ID_{USP} \oplus MK_{USP}$ [6].

C. Authentication and Key Establishment Phase

Following the registration step, U$_i$ and USP need to establish a session key (SK) and have a mutual authentication via CS. Over an insecure channel, this authentication key establishment (AKE) phase is carried out (Table IV).

1) U_i inputs ID_U, PW_i, and imprints BIO in SC. After that, SC calculates $RID_i = h_1(ID_U \parallel BIO)$, $RPW_i = h_1(PW_i \parallel BIO)$, $X_i = Q_i \oplus h_1(RID_i \parallel R^x_U) \oplus RPW_i$, $W^*_i = h(RID_i \parallel R^x_U \parallel X_i \parallel RPW_i)$, and verifies whether $W^*_i = W_i$. When it matches, SC accepts U_i, otherwise; aborts, and rejects the current session. From the premise set (C^x_U, R^x_U), SC chooses a timestamp T_1, a random nonce R_1, and a pair of (C^1_U, R^1_U). Later on, SC not only computes $M_1 = (ID_U \parallel R_1) \oplus h_2(X_i \parallel RID_i \parallel R^1_U \parallel T_1)$ and $Auth_U = h_1(ID_U \parallel R_1 \parallel R^1_U \parallel X_i \parallel T_1)$ but also sends $Msg_1 = y_i, L_i, M_1, Auth_U, C^1_U, T_1$ to CS.
2) The freshness of $|T_2 - T_1| \leq \Delta T_i$ is assessed by CS. From the premise set (C^x_{CS}, R^x_{CS}), CS creates a R_2, a T_2, and a pair of (C^1_{CS}, R^1_{CS}) if T_1 is matched. Then, CS computes $M_2 = (R_2 \parallel Z_j) \oplus h_3(ID_{CS} \parallel R^x_{CS} \parallel c_j \parallel T_2)$ and $Auth_{CS} = h_1(ID_{CS} \parallel R^1_{CS} \parallel R_2 \parallel Z_j \parallel T_2)$. CS transmits $Msg_2 = M_2$, $Auth_{CS}$, M_1, $Auth_U$, ID_{CS}, C^1_{CS}, T_2, C^1_U, T_1, y_i, L_i to USP.
3) In order to verify that $|T_3 - T_2| \leq \Delta T_i$, USP looks at $ID^*_{CS} = ID_{CS}$. In the case that T_1 and ID_{CS} match, USP computes $Z_j = h_1(ID_{CS} \parallel ID_{USP} \parallel MK_{USP} \parallel R^x_{CS}), and c_j = h_1(ID_{CS} \parallel MK_{USP})$. It then recovers the R^1_{CS} based on C^1_{CS}. Next, USP computes $(R_2 \parallel Z_j) = M_2 \oplus h_3(ID_{CS} \parallel R^x_{CS} \parallel c_j \parallel T_2)$, and $Auth^*_{CS} = h_1(ID_{CS} \parallel R^1_{CS} \parallel R_2 \parallel Z_j \parallel T_2)$. Subsequently, USP determines if $Auth^*_{CS} = Auth_{CS}$. If it matches, USP authenticates CS and then computes $R^*_r = L_i \oplus h_1(MK_{USP} \parallel y_i)$, and $RID^*_i = h_1(MK_{USP} \parallel R^*_r) \oplus y_i$.Subsequently, USP verifies that RID^*_i=RID_i. Based on C^1_U, it then retrieves the R^1_U and computes $X_i = E_i \oplus ID_{USP} \oplus MK_{USP}, (ID_U \parallel R_1) = M_1 \oplus h_2(X_i \parallel RID_i \parallel R^1_U \parallel T_1)$, and $Auth^*_U = h_1(ID_U \parallel R_1 \parallel R^1_U \parallel X_i \parallel T_1)$. Finally, USP verifies that $Auth^*_U = Auth_U$. If it matches, USP authenticates U_i successfully. Later on, USP generates a new R_3, T_3 and computes $y^+_i = h_1(MK_{USP} \parallel R_3) \oplus RID_i$, $L^+_i = h_1(MK_{USP} \parallel y^+_i) \oplus R_3$, and $f_{USP} = h_1(R_3 \parallel ID_{USP} \parallel MK_{USP})$. Then, USP not only calculates $M_3 =$

$(R_1 \parallel f_{USP}) \oplus h_3(R^1_{CS} \parallel Z_j \parallel R_2 \parallel ID_{CS})$, but also obtains $Auth_{USP-CS} = h_1(ID_{CS} \parallel R_2 \parallel f_{USP} \parallel R^1_{CS} \parallel Z_j \parallel T_3)$, $M_4 = (R_2 \parallel f_{USP} \parallel y_i^+ \parallel L_i^+) \oplus h_4(R^1_U \parallel X_i \parallel R_1 \parallel ID_U)$, and $Auth_{USP-U} = h_1(ID_U \parallel R_1 \parallel f_{USP} \parallel y_i^+ \parallel L_i^+ \parallel R^1_U \parallel X_i \parallel T_3)$. Lastly, $Msg_3 = M_3, Auth_{USP-CS}, M_4, Auth_{USP-U}, T_3$ is sent to CS by USP.

4) CS first checks freshness of $|T_4 - T_3| \leq \Delta T_i$. If T_3 is matches, CS computes $(R_1 \parallel f_{USP}) = M_3 \oplus h_3(R^1_{CS} \parallel Z_j \parallel R_2 \parallel ID_{CS})$, and $Auth^*_{USP-CS} = h_1(ID_{CS} \parallel R_2 \parallel f_{USP} \parallel R^1_{CS} \parallel Z_j \parallel T_3)$ and checks $Auth^*_{USP-CS} = Auth_{USP-CS}$. If it matches, USP is verified by CS. Furthermore, after CS chooses a T_4, and $Auth_{CS-U} = h_1(ID_{CS} \parallel R_1 \parallel R_2 \parallel T_4)$ is calculated. It also sends $Msg_4 = ID_{CS}, M_4, Auth_{USP-U}, Auth_{CS-U}, T_3, T_4$.

5) First, U_i determines whether $|T_5 - T_4| \leq \Delta T_i$ is fresh. In the case that T_4 is matched, U_i computes $(R_2 \parallel f_{USP} \parallel y_i^+ \parallel L_i^+) = M_4 \oplus h_4(R^1_U \parallel X_i \parallel R_1 \parallel ID_U)$. This yields $Auth^*_{USP-U} = h_1(ID_U \parallel R_1 \parallel f_{USP} \parallel y_i^+ \parallel L_i^+ \parallel R^1_U \parallel X_i \parallel T_3)$. U_i subsequently confirms that $Auth^*_{USP-U} = Auth_{USP-U}$. USP is authenticated by U_i when it is equal. $Auth^*_{CS-U} = h_1(ID_{CS} \parallel R_1 \parallel R_2 \parallel T_4)$ is computed by U_i in addition to $Auth^*_{CS-U} = Auth_{CS-U}$. U_i properly authenticates CS when it matches.

   After mutual authentication, U_i, CS, and USP generate a common SK $= h_1(R_1 \parallel R_2 \parallel f_{USP})$.

VII. Security Analysis of Our Scheme

In this section, we first analyze our scheme's security and then demonstrate its lack of flaws.

A. Formal Security Analysis

We assess the session key's semantic security in this part. We employ the RoR oracle model [27], where the goal of the adversary is to distinguish bits of SK from bits of a random number.

Three types of participants are involved in our protocol: $\Gamma^{t_1}_U$, $\Gamma^{t_2}_{CS}$, and $\Gamma_U SP^{t_3}$ represent instances t_1^{th} of the charging station and t_3^{th} of utility service provider, respectively. We use Hash(·) and PUF(·) functions as random oracles. To simulate real attacks in the RoR Model, according to Table V, the required query oracles for an adversary A are defined.

Theorem. The advantage of the adversary A to violate the semantic security of the session key in our proposed scheme is given as follows:

$$A = Adv(A) \leqslant \frac{(q_h^2)}{|Hash|} + \frac{(q_P^2)}{|PUF|} + 2\{C.q^s_{send}, \frac{q_s}{2^{(l_1)}}, \frac{q_s}{2^{(l_2)}}\} \tag{1}$$

Phrases q_h, Hash, q_P, and q_{send} are respectively "range space of h(·)", "some Hash query", "range space of PUF(·)" and "Send(·) query". And also, l_n, s, l_m and C are Zipf's credentials [28].

Proof. With the help of $GM_i (i \in [0, 4])$ games, we show that the relationship stated in the Theorem section holds.

Game GM_0: GM_0 is an actual attack executed by Adversary (A) in this scheme. The GM_0 result is as follows:

$$Adv(A) = |2.Adv(A, GM_0) - 1| \tag{2}$$

Game GM_1: GM_1 is an eavesdropping attack by an adversary (A). In this attack, the adversary intercepts the transmitted messages between three parties to the protocol with the help of an Execute(·) query. In this game, the adversary (A) uses Test(·) and Reveal(·) queries to obtain the session key. Adversary (A) needs R_1, R_2, and f_{USP} values to obtain the session key. Therefore, the probability of the adversary winning in this game does not increase by eavesdropping on the messages. Therefore, we have:

$$Adv(A, GM_1) = Adv(A, GM_0) \tag{3}$$

Game GM_2: GM_2 is active/passive attacks". Adversary by using Hash(·) and Send(·) queries intercepts the messages sent between three parties of the protocol. With the help of hash functions with random values and PUF values, all messages are protected against adversaries:

$$|Adv(A, GM_2) - Adv(A, GM_1)| \leqslant \frac{(q_h^2)}{2|Hash|} \tag{4}$$

Game GM_3: GM_3 is similar to game GM_2, except that the adversary uses a PUF query. Therefore, we have:

$$|Adv(A, GM_3) - Adv(A, GM_2)| \leqslant \frac{(q_P^2)}{2|PUF|} \tag{5}$$

Game GM_4: GM_4 is done using the CorruptSC(·) and CorruptCS(·) queries. Therefore, the attacker gets access to the values of Q_i and W_i in SC's memory. According to the relationships of $Q_i = X_i \oplus h(RID_i \parallel R^x_U) \oplus RPW_i$ and $W_i = h(RID_i \parallel R^x_U \parallel X_i \parallel RPW_i)$, the attacker cannot

Table IV: Authentication and key agreement phase of our proposed scheme

Electrical Vehicle User (u_i)	Charging Station (CS)	Utility Service Provider (USP)
Inputs ID_u, PW_i and imprints BIO in SC Calculates $RID_i = h_1(ID_i \parallel BIO)$ $RPW_i = h_1(PW_i \parallel BIO)$ $X_i = Q_i \oplus h_1(RID_i \parallel R_U^x)$ $W_i^* = h_1(RID_i \parallel R_U^x \parallel X_i \parallel RPW_i)$ Verifies $W_i^* = W_i$ Produces a random nonce R_1 and a timestamp T_1 Chooses a pair of $(C_U^1, R_U^1) from (C_U^x, R_U^x)$ Computes $M_1 = (ID_U \parallel R_1) \oplus h_2(X_i \parallel RID_i \parallel R_U^1 \parallel y_i \parallel L_i \parallel T_1)$ $Auth_U = h_1(ID_U \parallel R_1 \parallel R_U^1 \parallel X_i \parallel T_1)$ $\xrightarrow{Msg_1=y_i,L_i,M_1,Auth_U,C_U^1,T_1}$	Verifies $\lvert T_2 - T_1 \rvert \leq \Delta T_i$ Creates a random nonce R_2 and a timestamp T_2 Selects a pair of $(C_{CS}^1, R_{CS}^1) from (C_{CS}^x, R_{CS}^x)$ Calculates $M_2 = (R_2 \parallel Z_j) \oplus h_3(ID_{CS} \parallel R_{CS}^x \parallel c_j \parallel T_2)$ $Auth_{CS} = h_1(ID_{CS} \parallel R_{CS}^1 \parallel R_2 \parallel Z_j \parallel T_2)$ $\xrightarrow{Msg_2=ID_{CS},C_{CS}^1,T_2,C_U^1,T_1,M_2,Auth_{CS},y_i,L_i,M_1,Auth_U}$	Verifies $\lvert T_3 - T_2 \rvert \leq \Delta T_i$ *and checks* $ID_{CS}^* = ID_{CS}$ *Retrieves* R_{CS}^1 *on the basis of* C_{CS}^1 $Zj = h_1(ID_{CS} \parallel ID_{USP} \parallel MK_{USP} \parallel R_{CS}^x)$ $c_j = h_1(ID_{CS} \parallel MK_{USP})$ Calculates $(R_2 \Vert Z_j) = M_2 \oplus h_3(ID_{CS} \parallel R_{CS}^x \parallel c_j \parallel T_2)$ $Auth_{CS}^* = h_1(ID_{CS} \oplus R_{CS}^1 \oplus R_2 \oplus Z_j \oplus T_2)$ *Checks* $Auth_{CS}^* = Auth_{CS}$ Computes $R_r^* = L_i \oplus h_1(MK_{USP} \parallel y_i)$ $RID_i^* = h_1(MK_{USP} \parallel R_r^*) \oplus y_i$ *Checks* $RID_i^* = RID_i$ *Retrieves the* R_U^1 *on the basis of* C_U^1 Computes $X_i = E_i \oplus ID_{USP} \oplus MK_{USP}$ $(ID_U \parallel R_1) = M_1 \oplus h_2(X_i \parallel RID_i \parallel R_U^1 \parallel y_i \parallel L_i \parallel T_1)$ $Auth_U^* = h_1(ID_U \parallel R_1 \parallel R_U^1 \parallel X_i \parallel T_1)$ *Checks* $Auth_U^* = Auth_U$ Selects a new random nonce R_3 and a timestamp T_3 $y_i^+ = h_1(MK_{USP} \parallel R_3) \oplus RID_i$ $L_i^+ = h_1(MK_{USP} \parallel y_i^+) \oplus R_3$ $f_{USP} = h_1(R_3 \parallel ID_{USP} \parallel MK_{USP})$ $M_3 = (R_1 \parallel f_{USP}) \oplus h_3(R_{CS}^1 \parallel Z_j \parallel R_2 \parallel ID_{CS})$ $Auth_{USP-CS} = h_1(ID_{CS} \parallel R_2 \parallel f_{USP} \parallel R_{CS}^1 \parallel Z_j \parallel T_3)$ $M_4 = (R_2 \parallel f_{USP} \parallel y_i^+ \parallel L_i^+) \oplus h_4(R_U^1 \parallel X_i \parallel R_1 \parallel ID_U)$ $Auth_{USP-U} = h_1(ID_U \parallel R_1 \parallel f_{USP} \parallel y_i^+ \parallel L_i^+ \parallel R_U^1 \parallel X_i \parallel T_3)$ $\xleftarrow{Msg_3=M_3,Auth_{USP-CS},M_4,Auth_{USP-U},T_3}$
	Verifies $\lvert T_4 - T_3 \rvert \leq \Delta T_i$ Computes $(R_1 \parallel f_{USP}) = M_3 \oplus h_3(R_{CS}^1 \parallel Z_j \parallel R_2 \parallel ID_{CS})$ $Auth_{USP-CS}^* = h_1(ID_{CS} \parallel R_2 \parallel f_{USP} \parallel R_{CS}^1 \parallel Z_j \parallel T_3)$ Checks $Auth_{USP-CS}^* = Auth_{USP-CS}$ *Creates a timestamp* T_4 Calculates $Auth_{CS-U} = h_1(ID_{CS} \parallel R_1 \parallel R_2 \parallel T_4)$ $\xleftarrow{Msg_4=ID_{CS},M_4,Auth_{USP-U},Auth_{CS-U},T_3,T_4}$	
Verifies $\lvert T_5 - T_4 \rvert \leq \Delta T_i$ Computes $(R_2 \parallel f_{USP} \parallel y_i^+ \parallel L_i^+) = M_4 \oplus h_4(R_U^1 \parallel X_i \parallel R_1 \parallel ID_U)$ $Auth_{USP-U}^* = h_1(ID_U \parallel R_1 \parallel f_{USP} \parallel y_i^+ \parallel L_i^+ \parallel R_U^1 \parallel X_i \parallel T_3)$ Verifies $Auth_{USP-U}^* = Auth_{USP-U}$ Replace(y_i, L_i *with* y^+, L^+) Computes $Auth_{CS-U}^* = h_1(ID_{CS} \parallel R_1 \parallel R_2 \parallel T_4)$ Verifies $Auth_{CS-U}^* = Auth_{CS-U}$		
U_i, CS, and USP establish a common session key $SK = h_1(R_1 \parallel R_2 \parallel f_{USP})$		

obtain the value of PW_i without having BIO and PUF secret. Therefore, we have:

$$|Adv(A, GM_4) - Adv(A, GM_3)| \leqslant \{C.q_{send}^s, \frac{q_s}{2^{(l_b)}}\} \tag{6}$$

After finishing all the games, the attacker needs to guess c to win. Therefore, we have:

$$Adv(A, GM_4) = \frac{1}{2} \tag{7}$$

Combining the "formulas (2), (3), and (7)", we have:

$$\begin{aligned} \frac{1}{2}Adv(A) &= |Adv(A, GM_0) - \frac{1}{2}| \\ &= |Adv(A, GM_1) - \frac{1}{2}| \\ &= |Adv(A, GM_1) - Adv(A, GM_4)| \end{aligned} \tag{8}$$

Based on"triangular inequality"and formulas (4, 5, 6, and 8), we have:

$$\begin{aligned} \frac{1}{2}Adv(A) &= |Adv(A, GM_1) - Adv(A, GM_4)| \\ &\leqslant |Adv(A, GM_1) - Adv(A, GM_3)| \\ &\quad + |Adv(A, GM_3) - Adv(A, GM_4)| \\ &\leqslant |Adv(A, GM_1) - Adv(A, GM_2)| \\ &\quad + |Adv(A, GM_2) - Adv(A, GM_3)| \\ &\quad + |Adv(A, GM_3) - Adv(A, GM_4)| \\ &\leqslant \frac{(q_h^2)}{2|Hash|} + \frac{(q_P^2)}{2|PUF|} + \{C.q_{send}^s, \frac{q_s}{2^{(l_1)}}, \frac{q_s}{2^{(l_2)}}\} \end{aligned} \tag{9}$$

Finally, by using the formula (9) we have:

$$Adv(A) \leqslant \frac{(q_h^2)}{|Hash|} + \frac{(q_P^2)}{|PUF|} + 2\{C.q_{send}^s, \frac{q_s}{2^{(l_1)}}, \frac{q_s}{2^{(l_2)}}\} \tag{10}$$

Table V: Queries and Purposes

Query	**Purpose**
Send(Γ^t, Msg)	For this query, A can issue the message Msg to the Γ^t, and obtain the response message accordingly
CorruptSC($\Gamma_U^{t_1}$)	This query relates to smart card stolen attacks where A can recover the secret parameters stored in SC.
CorruptCS($\Gamma_U^{t_2}$)	This query is relevant to the physical capture attacks where A can recover the secret parameters stored in CS.
Test(Γ^t)	A biased coin c is tossed before the game starts. When A gets c = 1 under the test($\cdot$) , it means a SK among $\Gamma_U^{t_1}$, $\Gamma_{CS}^{t_2}$, and $\Gamma_{USP}^{t_3}$ are fresh. If A get the c = 0, it means SK is not fresh; Otherwise, A derives a null value ($\perp$).
Execute($\Gamma_U^{t_1}$, $\Gamma_{CS}^{t_2}$, and $\Gamma_{USP}^{t_3}$)	Under this query, A performs the passive/active attacks by eavesdropping the sent messages among $\Gamma_U^{t_1}$, $\Gamma_{CS}^{t_2}$, and $\Gamma_{USP}^{t_3}$ over a public channel.
Reveal(Γ^t)	Under this query, A compromises a SK cre ated among $\Gamma_U^{t_1}$, $\Gamma_{CS}^{t_2}$, and $\Gamma_{USP}^{t_3}$.

B. Informal Security Analysis

1) Mutual Authentication: In R2AKE-V2G, successful mutual authentication is performed by entire entities. USP checks CS by examining the condition $Auth_{CS}^* = Auth_{CS}$. The suggested scheme's stages state that the proper $Auth_{CS}$ can only be calculated by authorized CS who have the correct Z_j and M_2. Similarly, USP checks that $Auth_U^* = Auth_U$ in order to validate u_i. The proposed scheme's stages state that the right $Auth_U$ can only be calculated by authorized users who have the correct ID_U, M_1, and X_i. Furthermore, CS verifies USP's authenticity by assessing the condition $Auth_{USP-CS}^* = Auth_{USP-CS}$. The correct $Auth_{USP-CS}$ can only be computed by the authorized USP who has the correct Z_j and M_3. The correct $Auth_{USP-CS}$ can only be computed by the authorized USP who has the correct Z_j and M_3. Similarly, $Auth_{USP-U}^* = Auth_{USP-U}$ is evaluated by the user to authenticate USP. Only the authorized USP with the correct X_i and M_4 can compute the correct $Auth_{USP-U}$. Furthermore, the user authenticates CS by evaluating the condition $Auth_{CS-U}^* = Auth_{CS-U}$. Only the authorized USP with the correct R_1 and R_2 can compute the correct $Auth_{CS-U}$.
2) Replay Attack: Our scheme makes each session unique from the others by using random numbers and timestamps in all transmitted communications. This freshness of messages depends on random numbers that each party acquires and benefits from the random value used by the other party; thus, as deploying old messages would not be accepted by the other party, our scheme is resistant to replay attacks.
3) Man-in-the-Middle Attack (MITM): In the MITM attack, the attacker is in the middle of communicating parties and attempts to send and change messages to obstruct the protocol's normal execution. Assume, for instance, that the attacker wants to alter message Msg_1. The attack fails because X_i, which is computed using the master key (MK) and is unknown to the attacker, is included in M_1 and $Auth_U$. Furthermore, none of the messages y_i and L_i may be altered by the attacker. USP verifies the obtained identity with the identity stored in its database as it generates the RID_i using messages y_i and L_i and its master key Mk. When the condition $RID_i^* = RID_i$ is not met, the protocol is terminated.
4) Impersonation Attack: In the proposed scheme, the request messages Msg_1, Msg_2 and response messages Msg_3, Msg_4 related to mutual authentication between other entities cannot be created by the attacker. The credentials X_i and Z_j, as well as the PUF secret values R_U^1 and R_{CS}^1, cannot be extracted by the attacker. Because the attacker is unable to generate the request and response messages necessary for mutual authentication, our work is safe from this attack.
5) Anonymity and Untraceability: The feature's goal is to stop attackers from using intercepted messages delivered across an unsecured channel to determine the user's true ID. Moreover, an attacker might not even be able to determine a connection between two distinct sessions. In this scheme, the actual IDs of the users are not sent over the insecure channel without the use of a one-way hash function. They are also merged with random values that cause these values to vary in each session. Additionally, all communications exchanged at each session must be distinct from those transmitted at previous sessions to prevent tracking of crucial parameters. Thus, in every session, every message (Msg_1, Msg_2, Msg_3, and Msg_4) will be unique.
6) Forward/Backward Secrecy: Forward/backward secrecy guarantees that the security of the subsequent or prior session will not be compromised in the case that an adversary retrieves the current session key. The session key of the other sessions is determined using ephemeral session parameters like R_1, R_2, and R_3 of the current session that are independent of the other sessions, and all sensitive parameters in the session key are secured by the hash function

in our suggested protocol.

7) Perfect Forward Secrecy: Perfect forward secrecy ensures that past session keys remain hidden even in a case that both parties' long-term secret parameters, such as MK_{USP}, are disclosed. In order to obtain session key $SK = h(R_1 \parallel R_2 \parallel f_{USP})$, the attacker requires to obtain R_1, R_2, R_3, ID_{USP}.
8) Desynchronization Attack: The scheme may be vulnerable to desynchronization attacks when parties are required to update their values simultaneously, and after one party updates its desired values, the attacker forges the communicated data in a way that the other party cannot update the values concurrently. At the end of the protocol, the values of y_i and L_i are updated; however, only users of electrical vehicles are required to store these values, and the second side of the protocol, USP, is exempt from requiring the storage of updated values.
9) Denial-of-Service (DoS) Attack: In the suggested scheme, USP assesses the validity of the data received from U_i and CS at the start of the third phase and ends the session if it cannot be validated. Similarly, CS and U_i confirm the accuracy of the data acquired at the start of the fourth and fifth phases. As a result, our scheme is safe from denial-of-service attacks on all sides. A DoS attack can also be carried out by sending out outdated messages. Our scheme can also be secure against this kind of DoS attack because it is immune to replay attacks and takes advantage of time stamps.
10) Resistance Against Ephemeral Secret Leakage Attack: All of the crucial session parameters, including the session key (SK), MK_{USP}, y_i, and L_i, ID_U, will be safe even after all random session numbers, like R_1, R_2, and R_3, are revealed.

C. Security Verification Using Scyther Tool

Using the Scyther tool, we examined our scheme's objectives for secrecy and authentication. Our scheme is secure, as shown in Figure 3.

VIII. Performance Analysis

This section presents a performance comparison between our method and other R2AKE-V2G schemes.

A. Computational Costs

We compare the computational cost of our scheme with a similar scheme [6], using Table VI, which displays the execution time of various cryptographic operations. [6] obtained the execution times needed for cryptographic primitives by utilizing the well-known JCE [45] and PBC [44] libraries. In addition, they described the platform for U_i as a "Smartphone Lenovo Zuk Z1 with Quad-core 2.5 GHz processor using 4GB RAM and Android Operating System V5.1.2". Additionally, they employed a virtual machine running Ubuntu 16.11 OS and powered by an HP E8300 Core i5 2.93 GHz CPU with 4GB of RAM for the CS/USP server platform.

We present comparative results for the computation costs of our scheme and other comparable schemes in Table VIII. While the suggested AKE scheme has a lower computing cost and better security capabilities than the current relevant schemes, our scheme has a little higher computation cost than the existing related schemes. Thus, our scheme can be applied in real-world V2G scenarios.

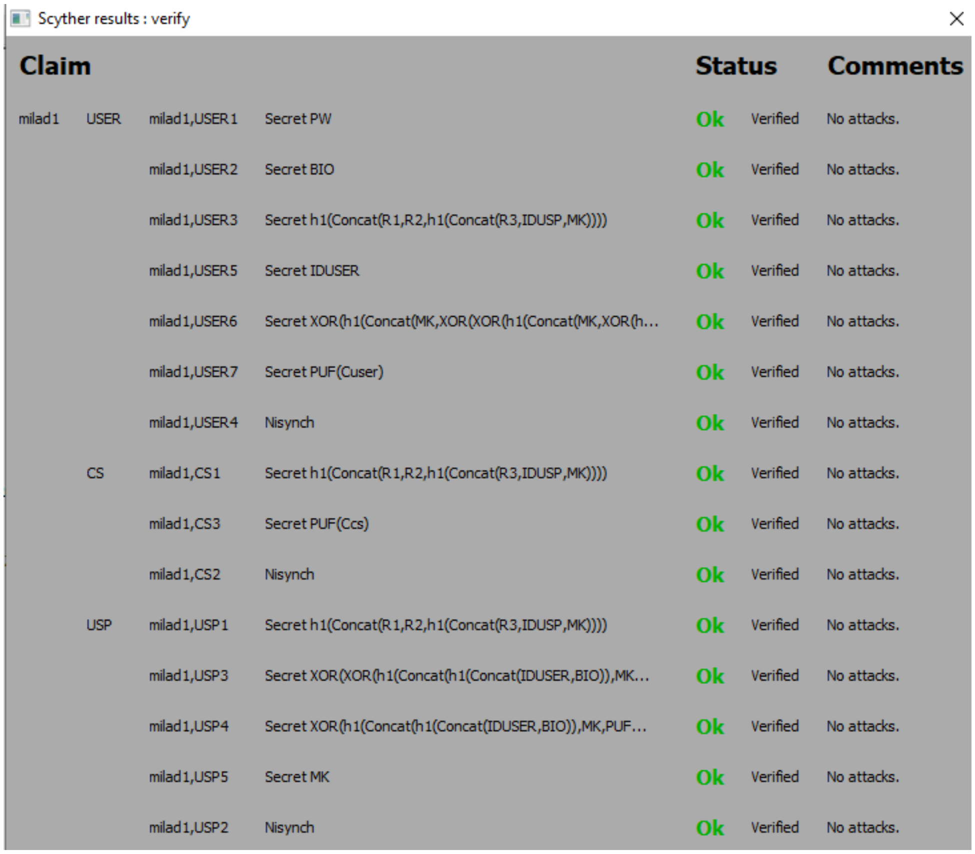
B. Communication Costs

The number of bits of data that have been transmitted is known as the lightness of a scheme, and this is measured by communication costs. We use the information in Table VII to compute the number of bits of messages delivered during the execution of the authentication phase to compare communication costs. In our scheme, the messages $Msg_1 = y_i, L_i, M_1, Auth_U, C_U^1, T_1$, $Msg_2 = ID_{CS}$, C_{CS}^1, T_2, C_U^1, T_1, M_2, $Auth_{CS}$, y_i, L_i, M_1, $Auth_U$, $Msg_3 = M_3, Auth_{USP-CS}, M_4, Auth_{USP-U}, T_3$ and $Msg_4 = ID_{CS}$, M_4, $Auth_{USP-U}$, $Auth_{CS-U}$, T_3, T_4 are $(160 + 160 + 220 + 160 + 60 + 32 = 792$ bits), $(60 + 60 + 32 + 60 + 32 + 320 + 160 + 160 + 160 + 220 + 160 = 1424$ bits), $(320 + 160 + 540 + 160 + 32 = 1212$ bits) and $(60 + 540 + 160 + 160 + 32 + 32 = 984$ bits), respectively, and hence, the total communication cost is 4412 bits. According to Section V-C, the communication cost of [6] is wrong on their paper since they have used only one hash function and must use different hash functions with different numbers of bits. As a result, their correct communication cost is 3652 bits.

Table VIII demonstrates the comparison of our scheme with other schemes in terms of communication costs.

C. Scalability and Data Overload

Unlike [6], our scheme does not employ symmetric encryption, so the secret token TK is not utilized. Considering the rise in the number of people using electric vehicles, this is highly ideal regarding scalability and data overload.



Scyther results : verify

Claim				Status		Comments
milad1	USER	milad1,USER1	Secret PW	Ok	Verified	No attacks.
		milad1,USER2	Secret BIO	Ok	Verified	No attacks.
		milad1,USER3	Secret h1(Concat(R1,R2,h1(Concat(R3,IDUSP,MK))))	Ok	Verified	No attacks.
		milad1,USER5	Secret IDUSER	Ok	Verified	No attacks.
		milad1,USER6	Secret XOR(h1(Concat(MK,XOR(XOR(h1(Concat(MK,XOR(h...	Ok	Verified	No attacks.
		milad1,USER7	Secret PUF(Cuser)	Ok	Verified	No attacks.
		milad1,USER4	Nisynch	Ok	Verified	No attacks.
	CS	milad1,CS1	Secret h1(Concat(R1,R2,h1(Concat(R3,IDUSP,MK))))	Ok	Verified	No attacks.
		milad1,CS3	Secret PUF(Ccs)	Ok	Verified	No attacks.
		milad1,CS2	Nisynch	Ok	Verified	No attacks.
	USP	milad1,USP1	Secret h1(Concat(R1,R2,h1(Concat(R3,IDUSP,MK))))	Ok	Verified	No attacks.
		milad1,USP3	Secret XOR(XOR(h1(Concat(h1(Concat(IDUSER,BIO)),MK...	Ok	Verified	No attacks.
		milad1,USP4	Secret XOR(h1(Concat(h1(Concat(IDUSER,BIO)),MK,PUF...	Ok	Verified	No attacks.
		milad1,USP5	Secret MK	Ok	Verified	No attacks.
		milad1,USP2	Nisynch	Ok	Verified	No attacks.

Figure 3: Output reports of analysis using Scyther

Table VI: EXECUTION TIME OF DIFFERENT CRYPTOGRAPHIC ELEMENTS

Notation	Description)	User's Device (ms))	USP/CS Server (ms))
Th	General hash operation	0.019	0.012
Tse	Symmetric enc/dec	0.063	0.048
Tmp	EC point multiplication	10.235	5.387
Te	Exponentiation	8.341	3.362
Tb	Bilinear pairing	13.662	7.318
Tmac	Mac operation	5.012	2.002
$Tcert_{gen}$	Certificate generation	69.326	-
$Tcert_{ver}$	Certificate verification	-	21.257

Table VII: NUMBER OF BITS OF DIFFERENT PARAMETERS

Parameters	Value (bits)
Timestamp	32
Identity	60
PUF	60
Random Nonce	160
Hash Function (h_1, h_2, h_3, h_4)	160, 220, 320, 540
Symmetric Enc/Dec	256
Elliptic Curve Point	320
Bilinear Pairing	320
Digital Signature	1024

IX. Conclusion

In this article, after a security examination, it was shown that the protocol proposed by Sungjin Yu et al. [6] is vulnerable to ephemeral secret leakage attacks and tracing attacks, which result in the loss of anonymity. Since it does not meet suitable anonymity standards, it is not optimal to perform on vehicle-to-grid networks. After that, we introduced a new scheme based on the PUF, using the Scyther tool to formally examine its security, and proved that the suggested protocol is semantically secure. We demonstrated that our scheme is ideal to implement in the context of V2G since it is lightweight and realistic in terms of computation, memory utilization, and data exchange costs.

Table VIII: Comparison of Computation and Communication Costs

Scheme	**User's Device (ms)**	**USP/CS Server (ms)**	**Communication Cost**
[9]	3Tmp + Tmac + $Tcert_{gen}$ + Th = 105.062 ms	4Tmp + Tmac + $Tcert_{ver}$ + 4Th + Ts = 44.903 ms	2590 bits
[10]	2Tmp + Tmac + $Tcert_{gen}$ + Th + Ts = 94.89 ms	3Tmp + Tmac + $Tcert_{ver}$ + 3Th + Ts = 39.504 ms	4836 bits
[11]	Ts + 4Th = 0.139 ms	Ts + 4Th = 0.096 ms	3922 bits
[13]	4Tmp + Te + 5Th = 49.376 ms	3Tmp + Te + 2Tb + 5Th = 34.219 ms	8190 bits
[14]	3Tmp + Te + 6Th = 39.16 ms	2Tmp + Te + 2Tb + 6Th = 28.844 ms	3466 bits
[15]	6Th = 0.114 ms	8Th = 0.096 ms	2144 bits
[18]	11Th = 0.209 ms	18Th = 0.216 ms	3196 bits
[6]	9Th = 0.171 ms	15Th + 4Ts = 0.372 ms	3562 bits
Our Scheme	9Th = 0.171 ms	20Th = 0.240 ms	4412 bits

References

[1] Ming Tao, Kaoru Ota, and Mianxiong Dong. Foud: Integrating fog and cloud for 5g-enabled v2g networks. *IEEE Network*, 31(2):8–13, 2017.

[2] SungJin Yu, KiSung Park, JoonYoung Lee, YoungHo Park, YoHan Park, SangWoo Lee, and BoHeung Chung. Privacy-preserving lightweight authentication protocol for demand response management in smart grid environment. *Applied Sciences*, 10(5):1758, 2020.

[3] Ismaila Adeniyi Kamil and Sunday Oyinlola Ogundoyin. Lightweight privacy-preserving power injection and communication over vehicular networks and 5g smart grid slice with provable security. *Internet of Things*, 8:100116, 2019.

[4] Neetesh Saxena, Santiago Grijalva, Victor Chukwuka, and Athanasios V Vasilakos. Network security and privacy challenges in smart vehicle-to-grid. *IEEE Wireless Communications*, 24(4):88–98, 2017.

[5] Wenlin Han and Yang Xiao. Privacy preservation for v2g networks in smart grid: A survey. *Computer Communications*, 91:17–28, 2016.

[6] Sungjin Yu and Kisung Park. Puf-based robust and anonymous authentication and key establishment scheme for v2g networks. *IEEE Internet of Things Journal*, 2024.

[7] Ran Canetti and Hugo Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In *International conference on the theory and applications of cryptographic techniques*, pages 453–474. Springer, 2001.

[8] Amin Mohammadali, Mohammad Sayad Haghighi, Mohammad Hesam Tadayon, and Alireza Mohammadi-Nodooshan. A novel identity-based key establishment method for advanced metering infrastructure in smart grid. *IEEE Transactions on Smart Grid*, 9(4):2834–2842, 2016.

[9] Hasen Nicanfar and Victor CM Leung. Multilayer consensus ecc-based password authenticated key-exchange (mcepak) protocol for smart grid system. *IEEE Transactions on Smart Grid*, 4(1):253–264, 2013.

[10] Dapeng Wu and Chi Zhou. Fault-tolerant and scalable key management for smart grid. *IEEE Transactions on Smart Grid*, 2(2):375–381, 2011.

[11] Jinyue Xia and Yongge Wang. Secure key distribution for the smart grid. *IEEE Transactions on Smart Grid*, 3(3):1437–1443, 2012.

[12] Je Hong Park, Minkyu Kim, and Daesung Kwon. Security weakness in the smart grid key distribution scheme proposed by xia and wang. *IEEE Transactions on Smart Grid*, 4(3):1613–1614, 2013.

[13] Jia-Lun Tsai and Nai-Wei Lo. Secure anonymous key distribution scheme for smart grid. *IEEE transactions on smart grid*, 7(2):906–914, 2015.

[14] Vanga Odelu, Ashok Kumar Das, Mohammad Wazid, and Mauro Conti. Provably secure authenticated key agreement scheme for smart grid. *IEEE Transactions on Smart Grid*, 9(3):1900–1910, 2016.

[15] Prosanta Gope and Biplab Sikdar. An efficient privacy-preserving authentication scheme for energy internet-based vehicle-to-grid communication. *IEEE Transactions on Smart Grid*, 10(6):6607–6618, 2019.

[16] Luis FA Roman, Paulo RL Gondim, and Jaime Lloret. Pairing-based authentication protocol for v2g networks in smart grid. *Ad Hoc Networks*, 90:101745, 2019.

[17] Azeem Irshad, Muhammad Usman, Shehzad Ashraf Chaudhry, Husnain Naqvi, and Muhammad Shafiq. A provably secure and efficient authenticated key agreement scheme for energy internet-based vehicle-to-grid technology framework. *IEEE Transactions on Industry Applications*, 56(4):4425–4435, 2020.

[18] Venkatasamy Sureshkumar, P Chinnaraj, P Saravanan, Ruhul Amin, and Joel JPC Rodrigues. Authenticated key agreement protocol for secure communication establishment in vehicle-to-grid environment with fpga implementation. *IEEE Transactions on Vehicular Technology*, 71(4):3470–3479, 2022.

[19] Masoud Kaveh and Mohamad Reza Mosavi. A lightweight mutual authentication for smart grid neighborhood area network communications based on physically unclonable function. *IEEE Systems Journal*, 14(3):4535–4544, 2020.

[20] Masoumeh Safkhani, Nasour Bagheri, Saqib Ali, Mazhar Hussain Malik, Omed Hassan Ahmed, Mehdi Hosseinzadeh, and Amir H Mosavi. Improvement and cryptanalysis of a physically unclonable functions based authentication scheme for smart grids. *Mathematics*, 11(1):48, 2022.

[21] Gaurang Bansal, Naren Naren, Vinay Chamola, Biplab Sikdar, Neeraj Kumar, and Mohsen Guizani. Lightweight mutual authentication protocol for v2g using physical unclonable function. *IEEE Transactions on Vehicular Technology*, 69(7):7234–7246, 2020.

[22] Alavalapati Goutham Reddy, Ponnuru Raveendra Babu, Vanga Odelu, Li Wang, and Sathish AP Kumar. V2g-auth: lightweight authentication and key agreement protocol for v2g environment leveraging physically unclonable functions. *IEEE Transactions on Industrial Cyber-Physical Systems*, 2023.

[23] Danny Dolev and Andrew Yao. On the security of public key protocols. *IEEE Transactions on information theory*, 29(2):198–208, 1983.

[24] Muhammad Naveed Aman, Kee Chaing Chua, and Biplab Sikdar. Mutual authentication in iot systems using physical unclonable functions. *IEEE Internet of Things Journal*, 4(5):1327–1340, 2017.

[25] Sungjin Yu and Youngho Park. A robust authentication protocol for wireless medical sensor networks using blockchain and physically unclonable functions. *IEEE Internet of Things Journal*, 9(20):20214–20228, 2022.

[26] Keith B Frikken, Marina Blanton, and Mikhail J Atallah. Robust authentication using physically unclonable functions. In *International Conference on Information Security*, pages 262–277. Springer, 2009.

[27] Michel Abdalla, Pierre-Alain Fouque, and David Pointcheval. Password-based authenticated key exchange in the three-party setting. In *Public Key Cryptography-PKC 2005: 8th International Workshop on Theory and Practice in Public Key Cryptography,*

Les Diablerets, Switzerland, January 23-26, 2005. Proceedings 8, pages 65–84. Springer, 2005.

[28] Ding Wang, Haibo Cheng, Ping Wang, Xinyi Huang, and Gaopeng Jian. Zipf's law in passwords. *IEEE Transactions on Information Forensics and Security*, 12(11):2776–2791, 2017.