Practical Two-Factor Authentication Protocol for Real-Time Data Access in WSNs

Meijia Xu and Ding Wang

***Abstract*—With the rapid development of sensors and communication technologies, the Internet of Things (IoT) paradigm has increasingly expanded to cover various daily, industrial, and military applications. As a core component of the IoT responsible for environmental sensing and data collection, wireless sensor networks (WSNs) face significant security threats. The design of multi-factor authentication schemes to secure real-time data transmission in WSNs has garnered considerable research efforts. However, a common trend in existing multi-factor authentication protocols is emphasizing performance and security benefits, while possible limitations are rarely subjected to thorough analysis. To fill this gap, we first select two representative multi-factor authentication schemes (i.e., Chaudhry et al.'s scheme at ACM ToIT'21 and Jabbari-Mohasef's scheme at IEEE TII'22) as case studies, and point out that both schemes are vulnerable to offline password guessing and node capture attacks, and fail to achieve forward secrecy. We then investigate the fundamental causes of these weaknesses underlying the schemes and propose corresponding solutions. After rethinking previous schemes, we present a new robust two-factor authentication scheme for WSNs and formally prove its security under the Random Oracle Model. Furthermore, we compare our scheme with 18 state-of-the-art protocols using the widely accepted evaluation framework. The comparison results show that our protocol outperforms its foremost counterparts.**

***Index Terms*—Multi-factor authentication, Internet of Things, wireless sensor networks, offline password guessing, forward secrecy.**

I. Introduction

As Internet of Things (IoT) technology matures and evolves toward intelligence, wireless sensor networks (WSNs) have been widely adopted in various domains (e.g., Industrial Internet of Things [1], smart home [2], and environmental monitoring [3]). In many critical applications (e.g., battlefield environmental monitoring [4], patient heartbeat monitoring [5]), external users often require real-time access to sensor node data. However, the long communication links between sensor nodes and gateways in WSNs negatively affect the efficiency of data queries. Consequently, enabling users to access real-time data directly from sensor nodes on demand has become a fundamental feature of WSNs [6]. Meanwhile, given the openness of WSNs channels and the mobility of users, it is necessary to protect sensitive data and user behavior information from various attacks, eavesdropping, modification, etc. The consequences could be potentially disastrous once malicious/unauthorized users access sensitive data. To mitigate these risks, user authentication is essential to prevent malicious or unregistered users from illegally accessing private data. User authentication is typically divided into three types: 1) something the user knows (e.g., passwords), 2) something the user has (e.g., smart cards), and 3) something the user is (e.g., fingerprints) [7].

Passwords remain the dominant form of user authentication today due to their key advantages, such as low deployment cost, convenient account recovery, and remarkable simplicity [8], [9]. Password-based authentication key exchange (PAKE) protocols allow two parties to establish a shared session key based on a low-entropy password and are one of the most widely used cryptographic protocols. A key characteristic of these protocols is that they require servers to store a password verifier (i.e., parameters derived from the user's password via a one-way function, such as a salted hash) for authentication [10], [11], [12]. Nowadays, catastrophic data breaches have become alarmingly common (e.g., 3 billion Yahoo leak [13], 1.1 billion Alibaba breach [14]). Once password verifiers are exposed, attackers can immediately exploit them to conduct offline password guessing attacks. This vulnerability is further exacerbated by the continuous advancement of offline guessing algorithms [15]. In recent years, multi-factor authentication (MFA) has become a widely adopted method to enhance user authentication security without requiring servers to store password files [16]. For example, Google Cloud plans to enforce MFA for all users by the end of 2025. MFA leverages two or more authentication factors to verify the user's identity, ensuring that even if an adversary compromises $n-1$ factors in an n-factor authentication scheme (e.g., $n=2,3$), the system remains secure [16]. This method significantly reduces reliance on a single factor.

As shown in Fig. 1, a typical multi-factor authentication protocol in WSNs involves three types of participants: gateways, users, and a large number of sensor nodes [6]. Unlike traditional client-server structures, WSNs consist of numerous sensor nodes constrained by limited storage and computational resources. These nodes are deployed in unattended environments, making them vulnerable to physical attacks. Therefore, in WSNs, beyond addressing the security challenges of conventional MFA

Received 27 September 2022; revised 11 April 2025; accepted 11 April 2025. This work was supported in part by the National Natural Science Foundation of China under Grant 62222208 and Grant 62172240 and in part by the Fundamental Research Funds for the Central Universities (Nankai University) under Grant 63253229. *(Corresponding author: Ding Wang.)*

The authors are with the College of Cryptology and Cyber Science, Nankai University, Tianjin 300350, China, also with the Key Laboratory of Data and Intelligent System Security, Ministry of Education, Tianjin 300350, China, and also with the Tianjin Key Laboratory of Network and Data Security Technology, Nankai University, Tianjin 300350, China (e-mail: xumeijia@mail.nankai.edu.cn; wangding@nankai.edu.cn).

Digital Object Identifier 10.1109/TDSC.2025.3563552

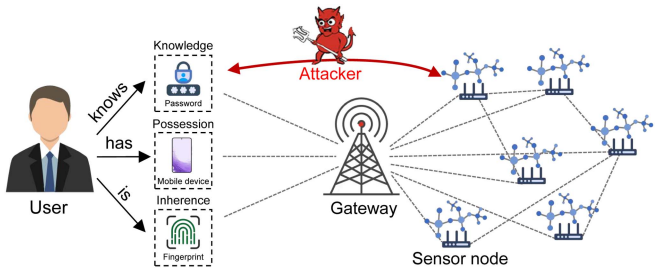

Fig. 1. Multi-factor authentication scheme in WSNs.

schemes, it is necessary to ensure security when a subset of nodes is compromised [7]. *The unique challenges posed by WSNs demand a robust multi-factor authentication scheme to ensure the security of authentication and key establishment.*

A. Related Work

In 2009, Das et al. [17] first proposed a two-factor (password+smart card) authentication scheme in WSNs, allowing users to access the information stored in sensor nodes securely. However, subsequent works [18], [19] revealed vulnerabilities in Das et al.'s scheme. The scheme is vulnerable to privileged insider attacks, offline password guessing attacks, user impersonation attacks, etc. In 2011, Fan et al. [18] proposed a new two-factor authentication scheme to improve the security of Das et al.'s and similar schemes. It repaired some defects of Das et al.'s [17] scheme, but Wang et al. [20] subsequently demonstrated that Fan et al.'s [18] scheme failed to ensure user anonymity. Subsequently, Das et al. [21] proposed a new scheme in 2012 and introduced dynamic node addition for sensor network expansion. Still, Li et al. [22] later revealed that Das et al.'s protocol cannot prevent the adversary from obtaining long-term session keys and cannot ensure user anonymity.

In 2013, Xue et al. [23] proposed a lightweight two-factor authentication scheme, which first uses temporary credentials to achieve user anonymity. However, Xue et al.'s method was later shown to be ineffective. Specifically, Ma et al. [24] highlighted the necessity of public-key techniques in password authentication schemes to ensure forward secrecy, achieve user anonymity, and resist offline password guessing attacks. Nonetheless, prior failures and Ma et al.'s principle [24] failed to attract sufficient attention from protocol designers. Furthermore, many protocol designers [25], [26], [27], [28], [29], [30], [31] tend to prioritize the efficiency of their own schemes, which hinders the objective analysis of their deficiencies.

In 2018, Wazid et al. [31] proposed a smart card-based password authentication protocol. Then, Wang et al. [6] found Wazid et al.'s scheme cannot achieve forward secrecy and cannot resist offline password guessing attacks. Moreover, to address the "break-fix-break-fix" cycle in multi-factor authentication for WSNs, Wang et al. [6] proposed a systematic evaluation framework for objectively assessing authentication schemes. Using this framework, they conducted a comprehensive evaluation of 44 representative schemes.

In 2022, based on the protocol of Wazid et al. [31], Jabbari and Mohasef [26] proposed a multi-factor authentication protocol for the long range wide area network (LoRaWAN) using AES symmetric encryption technology. Jabbari and Mohasef claimed that the scheme could achieve all proposed security goals. Moreover, Wazid et al. [27] recently proposed a multi-factor authentication protocol based on cloud IoT. Later, Chaudhry et al. [25] found that when there are multiple users, mutual authentication and key agreement cannot be implemented in Wazid et al.'s protocol [27]. Then, Chaudhry et al. [25] proposed an improved protocol to remedy this problem. Still, as our analysis reveals, the protocols of Jabbari-Mohasef [26] and Chaudhry et al. [25] fail to achieve the security goals they claimed.

In 2022, Wang et al. [7] pointed out that node capture attacks are one of the most common security threats to multi-factor authentication schemes in WSNs, yet they have received limited attention in research. Wang et al. [7] conducted an in-depth analysis of node capture attacks and their causes, categorizing them into ten types and proposing modification suggestions for several typical vulnerable protocols. Although Wang et al. [7] intended to help protocol designers better understand node capture attacks, many newly proposed protocols are still vulnerable to such attacks (e.g., [32], [33]). Moreover, these newly proposed protocols (e.g., [33], [34], [35]) also suffer from various security issues. Designing a practical multi-factor user authentication scheme for WSNs remains a challenging task.

B. Motivations and Contributions

Research on multi-factor authentication protocols over the past 30 years [16], [36], [37] demonstrates that designing secure password-based MFA protocols remains challenging. For WSNs, their unique characteristics bring more challenges. Specifically, sensor nodes are often deployed in unattended environments, making them highly vulnerable to physical attacks [6]. In other words, adversaries can easily capture these nodes. Consequently, in addition to common attacks on multi-factor authentication schemes, it is essential to prevent node capture attacks, which significantly increase the challenge of designing secure authentication schemes for WSNs. Although researchers continuously propose new schemes [26], [32], [33], [38], most still contain vulnerabilities. Instead, they rely on superficial fixes to address vulnerabilities in previous insecure protocols, ultimately perpetuating the same issues. This lack of fundamental analysis limits the reliability of these protocols and also hinders the development of secure schemes for WSNs.

To address this issue, this work aims to provide deep insights into the inherent problems of existing protocols and their corresponding countermeasures. Specifically, we conduct a detailed analysis of two representative MFA protocols for WSNs [25], [26], identifying several critical flaws that are also prevalent in many newly proposed protocols [28], [29], [30], [32], [33], [39], [40]. By analyzing the root causes of these vulnerabilities, we highlight the fundamental design challenges (e.g., resistance to offline password guessing, node capture, and de-synchronization attacks) that *must* be addressed. Based on these findings, we propose a new practical two-factor authentication protocol tailored to the security requirements of WSNs. In summary, our contributions are three-fold:

1) We analyze two representative multi-factor authentication schemes for WSNs: Chaudhry et al. (ACM ToIT'21) [25] and Jabbari-Mohasef (IEEE TII'22) [26]. Our investigation demonstrates that neither scheme achieves truly multi-factor security and forward secrecy, and both are vulnerable to node capture attacks and offline password guessing attacks. Furthermore, we systematically reveal the root causes of these security weaknesses and propose corresponding effective countermeasures.
2) Based on the empirical insights gained from analyzing design vulnerabilities in state-of-the-art MFA schemes for WSNs and their corresponding remediation strategies, we present a new two-factor authentication protocol for WSNs. We formally prove the security of our proposed protocol under the random oracle model (ROM). Considering that certain practical scenarios like de-synchronization attacks [41] cannot be fully captured by ROM analysis, we complement this with comprehensive heuristic analysis.
3) Following the well-established evaluation framework of Wang et al. [6], we conduct a systematic comparison between our scheme and 18 state-of-the-art protocols. The comparison results show that our protocol outperforms its counterparts significantly.

II. Adversary Model & Evaluation Criteria

A. Adversary Model

A reasonable adversary model should assume that the attacker possesses the most powerful capabilities possible, except for those that would trivially compromise the security of any protocol. We employ Wang et al.'s adversary model for WSN environment proposed in 2018 [6]. It is one of the few harsh but reasonable models, which has been highly recommended (see [42], [43]) and widely accepted (see [44], [45], [46]). The capabilities of the adversary $\mathcal{A}$ are as follows.

C1. $\mathcal{A}$ can offline enumerate all items in the space of identity and password. The user identity ID usually has a fixed format and is randomly selected. Meanwhile, Wang et al. have proved that the distribution of user passwords usually follows the Zipf's law [47]. This implies that the space of password $\mathcal{D}_{pw}$ and identity $\mathcal{D}_{id}$ are limited in reality ($\mathcal{D}_{id} \leq \mathcal{D}_{pw} \approx 10^6$).
C2. $\mathcal{A}$ can fully control the open channel. According to Dolev-Yao model [48], the adversary $\mathcal{A}$ can eavesdrop, intercept, modify, delete and insert any transmitted information over the public channel.
C3. $\mathcal{A}$ can obtain previous session keys. A memory leak or improper sanitization of the sensor node or user device will disclose the disclosure of the session key. It could cause the attacker to launch a known-key attack.
C4. $\mathcal{A}$ can access the information stored in the GWN. When $\mathcal{A}$ is an insider, she can obtain all information stored in the GWN except for its secret key.
C5. $\mathcal{A}$ can acquire the long-term secret key when considering the forward secrecy. Forward secrecy is regarded as the last line of defense for system security. We assume that $\mathcal{A}$ can obtain the long-term private key only when evaluating the protocol's forward secrecy property.
C6. $\mathcal{A}$ can access secret information stored in sensor nodes. Sensor nodes are often deployed in unattended environments making them vulnerable to physical attacks [6], [7]. The attackers can launch attacks and get secret information stored in the nodes.
C7. $\mathcal{A}$ can compromise any n-1 factors in an n-factor authentication scheme. Multi-factor security means the attacker cannot break the security of an n-factor scheme even if she obtains n-1 authentication factors.

TABLE I
NOTATIONS AND ABBREVIATIONS

Symbol	Description	Symbol	Description
U_i	the i^{th} user	S_j	the j^{th} sensor node
GW	the gateway node	NS	the network server
RID_{Ui}	pseudo identity of U_i	s	the long-term secret key of NS
RID_{Sj}	pseudo identity of S_j	K_j	shared key of trusted authority
ID_i, ID_{Sj}	identity of U_i, S_j	TD_i	dynamic identity of U_i
TA	the trusted authority	PW_i, SC_i	password, smart card of U_i
Gen/Rep	fuzzy extractor	X	long-term secret key of GWN
E_j	the j^{th} end-device	π_i, ρ_i	long-term secret keys of U_i
aes_cmac	AES-CMAC	$aes\_encrypt$	AES encryption algorithm
$h(\cdot)$	one-way hash function	RTS_{Sj}	registration timestamp of S_j
$\rightarrow$	an insecure channel	$\oplus$	bitwise XOR operation
$\Longrightarrow$	a secure channel	$\|$	concatenation operation

B. Evaluation Criteria

Wang et al. [6] propose the evaluation criteria tailored to WSNs. These criteria are systematic, comprehensive, and concrete, and have been widely accepted (see [7], [45]). We adopt these criteria to evaluate recently proposed protocols, focusing on aspects such as security and user-friendliness. A summary of the evaluation criteria is provided in Table II.

III. Review of Chaudhry et al.'s Scheme

Recently, Chaudhry et al. [25] point out that Wazid et al.'s protocol [27] proposed in 2020 fails to achieve mutual authentication in scenarios with multiple users. Chaudhry et al. [25] make improvements to Wazid et al.'s protocol and prove that their protocol is secure and effective. In this section, we briefly review Chaudhry et al.'s protocol [25]. The intuitive notations and abbreviations are listed in Table I.

A. Registration Phase

1) Sensor Nodes Registration

Step 1: TA computes $RID_{Sj} = h(ID_{Sj} \parallel K_j)$ and $TC_{Sj} = h(ID_{Sj} \parallel K_j \parallel RTS_{Sj})$ for S_j.
Step 2: TA stores RID_{Sj}, TC_{Sj} and $h(RID_{Sj} \parallel K_j)$ in S_j.
Step 3: TA stores RID_{Sj}, TC_{Sj}, $h(RID_{Sj} \parallel K_j)$ and secret key X in gateway GWN's memory.

2) User Registration

Step 1: $U_i \Rightarrow TA$: $\{ID_i\}$.
Step 2: $TA \Rightarrow U_i$: $SC_i : \{TD_i, TC_{Ui}, R_i, h(.)\}$.
When receiving the message, TA selects K_i, TD_i and computes $TC_{Ui} = h(ID_i \| K_i)$. Then TA gets $R_i = h(ID_i \parallel X)$, $K_i' = h(K_i \parallel R_i)$ and $RID_{Ui} = h(ID_i \parallel R_i)$. TA stores $TD_i, TID_{Ui} =$

TABLE II
EVALUATION CRITERIA

Short term		Definition in WSNs
S1	No password verifier-table	The verifier-table related to password should not be stored in the gateway and sensor nodes.
S2	Password friendly	The user can choose the password at will and change it locally.
S3	Password security	The gateway's privileged administrator cannot break the user's password.
S4	No smart card loss attack	If the attacker obtains the information stored in the smart card, she cannot increase her attack advantage.
S5	Resistance to known attacks	The protocol can resist all known attacks.
S6	Scheme repairability	The scheme is able to revoke smart cards and add dynamic sensor nodes.
S7	Establishment of the session key	A shared session key must be established to protect the security of the next communication.
S8	No clock synchronization	It is best not to use clock synchronization, which might bring about de-synchronization attacks.
S9	Timely typo detection	The smart card should detect and prompt the user for authentication failure due to a typo in time.
S10	Mutual authentication.	Participants should be able to authenticate each other's identities.
S11	User anonymity	The adversary cannot obtain the identity and cannot distinguish whether the two messages are from the same user.
S12	Forward secrecy	If the attacker obtains a long-term secret key, she cannot get the previous session key.

$ID_i \oplus K_i$, $WID_{Ui} = K_i \oplus h(TD_i \parallel X)$ in the verifier-table maintained by corresponding GW and sends a smart card SC_i: $\{TD_i, TC_{Ui}, R_i, h(.)\}$ to U_i in a secure channel.

Step 3: U_i selects password PW_i, secret number k and imprints biometrics BIO_i. Then U_i gets $Gen(BIO_i) = (\sigma_i, \tau_i)$, $RPW_i = h(PW_i \parallel k)$, $A_i = h(\ ID_i \parallel \sigma_i) \oplus k$, $F_i = h(RPW_i \parallel \sigma_i \parallel R_i \parallel TC_{Ui})$, $R_i^* = R_i \oplus h(ID_i \parallel k)$, $TD_i^* = TD_i \oplus h(k \parallel \sigma_i)$ and $TC_{Ui}^* = TC_{Ui} \oplus h(ID_i \parallel \sigma_i \parallel PW_i)$. After that, U_i replaces TD_i, TC_{Ui} by TD_i^*, TC_{Ui}^* in SC_i. Finally, the smart card contains $\{TD_i^*, TC_{Ui}^*, R_i^*, F_i, A_i, \tau_i, h(.), Gen(.), Rep(.), t\}$.

B. Login and Authentication Phase

Step 1. $U_i \rightarrow GW$: $Msg_1 = \{M_1, M_2, M_3, M_4, T_1\}$. U_i user enters $\{ID_i, PW_i, BIO_i\}$. SC_i computes $\sigma_i^* = Rep(BIO_i^*, \tau_i)$, $k = A_i \oplus h(ID_i \parallel \sigma_i^*)$, $R_i = R_i^* \oplus h(ID_i \parallel k)$, $RID_{Ui} = h(ID_i \parallel R_i)$, $TC_{Ui} = TC_{Ui}^* \oplus h(ID_i \parallel \sigma_i^* \parallel PW_i^*)$, $RPW_i^* = h(PW_i^* \parallel k)$ and $F_i^* = h(RPW_i^* \parallel \sigma_i^* \parallel R_i \parallel TC_{Ui})$. After that, SC_i checks if $F_i^* = F_i$. If $F_i^* \neq F_i$, rejects the request. The SC_i selects T_1, r_1 and computes $M_1 = TD_i^* \oplus h(k \parallel \sigma_i)$, $M_2 = RID_{Sj} \oplus h(TC_{Ui} \parallel ID_i \parallel T_1)$, $M_3 = h(RID_{Ui} \parallel TC_{Ui} \parallel T_1) \oplus r_1$ and $M_4 = h(ID_i \parallel RID_{Sj} \parallel TC_{Ui} \parallel r_1 \parallel T_1)$. U_i sends $Msg_1 = \{M_1, M_2, M_3, M_4, T_1\}$ to GW.

Step 2. $GW \rightarrow S_j$: $Msg_2 = \{M_6, M_7, M_8, T_2\}$. After receiving the request, GW checks the freshness of T_1 by comparing with current timestamp $T_{cur} - T_1 \leq \Delta T$. GW uses M_1 to extract TID_{Ui} and WID_{Ui}. GW computes $K_i = h(M_1 \parallel X) \oplus WID_{Ui}$, where X is the shared secret key. Then computing $ID_i = K_i \oplus TID_{Ui}$, $R_i = h(ID_i \parallel X)$, $RID_{Ui} = h(ID_i \parallel R_i)$, $RID_{Sj} = M_2 \oplus h(TC_{Ui} \parallel ID_i \parallel T_1)$, $r_1 = M_3 \oplus h(RID_{Ui} \parallel TC_{Ui} \parallel T_1)$ and $M_5 = h(ID_i \parallel RID_{Sj} \parallel TC_{Ui} \parallel r_1 \parallel T_1)$. GW will reject the request if $M_5 \neq M_4$. Otherwise, GW generates r_2, T_2 and computes $M_6 = h(TC_{Sj} \parallel RID_{Sj}) \oplus r_2$, $M_7 = h(RID_{Ui} \parallel TC_{Ui} \parallel r_1) \oplus h(TC_{Sj} \parallel T_2)$, and $M_8 = h(RID_{Sj} \parallel TC_{Sj} \parallel h(RID_{Sj} \parallel K_j) \parallel r_2 \parallel T_2)$. Then GW sends $Msg_2 = \{M_6, M_7, M_8, T_2\}$ to the sensor node S_j.

Step 3. $S_j \rightarrow GW$: $Msg_3 = \{M_{10}, M_{11}, M_{12}, T_3\}$. S_j checks the freshness of T_2 by comparing with current timestamp $T_{cur} - T_2 \leq \Delta T$. Then S_j computes $r_2 = M_6 \oplus h(TC_{Sj} \parallel RID_{Sj})$, $h(RID_{Ui} \parallel TC_{Ui} \parallel r_1) = M_7 \oplus h(TC_{Sj} \parallel T_2)$ and $M_9 = h(RID_{Sj} \parallel TC_{Sj} \parallel h(RID_{Sj} \parallel K_j) \parallel r_2 \parallel T_2)$. S_j rejects if $M_9 \neq M_8$ and otherwise generates r_3, T_3 and computes $M_{10} = h(h(RID_{Ui} \parallel TC_{Ui} \parallel r_1) \parallel T_3) \oplus r_3$, $M_{11} = h(h(RID_{Ui} \parallel TC_{Ui} \parallel r_1) \parallel RID_{Sj} \parallel r_3) \oplus h(h(RID_{Sj} \parallel K_j) \parallel T_3)$, $SK_{ij} = h(h(h(RID_{Sj} \parallel K_j) \parallel T_3) \parallel h(RID_{Ui} \parallel TC_{Ui} \parallel r_1) \parallel RID_{Sj} \parallel r_3 \parallel T_3)$, and $M_{12} = h(SK_{ij} \parallel T_3)$. Then the S_j sends $Msg_3 = \{M_{10}, M_{11}, M_{12}, T_3\}$ to the gateway node GW.

Step 4. $GW \rightarrow U_i : Msg_4 = \{Msg_3, M_{13}, \overline{TD}_{i_{new}}, T_4\}$. After receiving Msg_3, GW checks the freshness of T_3 by comparing if $T_{cur} - T_3 \leq \Delta T$. GW generates $\{TD_{i_{new}}, T_4\}$ and computes $\overline{TD}_{i_{new}} = TD_{i_{new}} \oplus h(RID_{Ui} \parallel TC_{Ui} \parallel r_1 \parallel T_4)$, $TD_i = TD_{i_{new}}$, $M_{13} = h(RID_{Ui} \parallel TD_{i_{new}} \parallel T_4)$ and $WID_{Ui} = K_i \oplus h(TD_{i_{new}} \parallel X)$. Then GW sends $Msg_4 = \{Msg_3, M_{13}, \overline{TD}_{i_{new}}, T_4\}$ to U_i. To avoid any de-synchronization between GW and U_i, the GW stores previous TD_i in temporary variable, and on the next login, it is updated with the current one.

Step 5. When receiving the Msg_4, U_i checks the freshness of T_4 by comparing with current timestamp $T_{cur} - T_4 \leq \Delta T$. U_i gets $r_3 = M_{10} \oplus h(h(RID_{Ui} \parallel TC_{Ui} \parallel r_1) \parallel T_3)$, and $h(h(RID_{Sj} \parallel K_j \parallel T_3)) = M_{11} \oplus h(h(RID_{Ui} \parallel TC_{Ui} \parallel r_1) \parallel RID_{Sj} \parallel r_3)$, $SK'_{ij} = h(h(h(RID_{Sj} \parallel K_j) \parallel T_3) \parallel h(RID_{Ui} \parallel TC_{Ui} \parallel r_1) \parallel RID_{Sj} \parallel r_3 \parallel T_3)$, $M_{14} = h(SK'_{ij} \parallel T_3)$ and $M_{15} = h(RID_{Ui} \parallel TD_{i_{new}} \parallel T_4)$. U_i then checks if $M_{14} = M_{12}$ & $M_{15} = M_{13}$. Upon success, U_i computes $TD_{i_{new}} = \overline{TD}_{i_{new}} \oplus h(RID_{Ui} \parallel TC_{Ui} \parallel r_1 \parallel T_4)$ and replaces TD_i by $TD_{i_{new}}$. U_i keeps SK'_{ij} as shared key with S_j.

IV. ANALYSIS OF CHAUDHRY ET AL.'S SCHEME

In this section, we reveal that although Chaudhry et al. [25] proposed an improved protocol to address the problems of

Wazid et al.'s scheme [27], Chaudhry et al.'s scheme fails to achieve the claimed security properties.

A. No Truly Multi-Factor Security

Generally, a protocol can achieve multi-factor security if the adversary cannot break the security of the remaining factor when she obtains all the other factors [16]. However, we find that Chaudhry et al.'s scheme [25] cannot achieve this goal. The specific attack steps are as follows.

Step 1. $\mathcal{A}$ chooses ID_i^*, PW_i^* from $\mathcal{D}_{id} \times \mathcal{D}_{pw}$.
Step 2. $\mathcal{A}$ computes $\sigma_i^* = Rep(BIO_i, \tau_i)$.
Step 3. $\mathcal{A}$ computes $k^* = A_i \oplus h(ID_i^* \| \sigma_i^*)$.
Step 4. $\mathcal{A}$ computes $R_i = R_i^* \oplus h(ID_i^* \| k^*)$.
Step 5. $\mathcal{A}$ further computes the value $TC_{Ui} = TC_{Ui}^* \oplus h(ID_i^* \| \sigma_i^* \| PW_i^*)$.
Step 6. $\mathcal{A}$ computes $RPW_i^* = h(PW_i^* \| k^*)$.
Step 7. $\mathcal{A}$ computes $F_i^* = h(RPW_i^* \| \sigma_i^* \| R_i \| TC_{Ui})$.
Step 8. Repeat step 1∼7 until ID_i^*, PW_i^* meet F_i^*=F_i.

The time complexity: $\mathcal{O}((5T_H + T_B + 3T_{XOR}) \times |\mathcal{D}_{ID}| \times |\mathcal{D}_{PW}|)$, where T_H is the time of hash-function, T_B is the time of fuzzy extractor, $|\mathcal{D}_{PW}|$ and $|\mathcal{D}_{ID}|$ are the space of ID and PW. Generally, the user's identity has a fixed format, and the protocol allows users to choose their passwords freely. Therefore, the space of password and identity is limited in reality ($\mathcal{D}_{ID} \leq \mathcal{D}_{PW} \approx 10^6$ [47]). It is reasonable that the adversary can launch this attack.

B. Offline Password Guessing Attack

Chaudhry et al.'s scheme [25] is also vulnerable to another offline password guessing attack. This attack also makes the protocol unable to achieve true multi-factor security. The specific attack steps are as follows.

Step 1. $\mathcal{A}$ chooses ID_i^*, PW_i^* from $\mathcal{D}_{id} \times \mathcal{D}_{pw}$.
Step 2. $\mathcal{A}$ computes $\sigma_i^* = Rep(BIO_i, \tau_i)$.
Step 3. $\mathcal{A}$ computes $k^* = A_i \oplus h(ID_i^* \| \sigma_i^*)$.
Step 4. $\mathcal{A}$ computes $R_i = R_i^* \oplus h(ID_i^* \| k^*)$.
Step 5. $\mathcal{A}$ computes $RID_{Ui}^* = h(ID_i^* \| R_i)$.
Step 6. $\mathcal{A}$ further computes the value $TC_{Ui} = TC_{Ui}^* \oplus h(ID_i^* \| \sigma_i^* \| PW_i^*)$.
Step 7. $\mathcal{A}$ computes $r_1^* = M_3 \oplus h(RID_{Ui}^* \| TC_{Ui} \| T_1)$.
Step 8. $\mathcal{A}$ further computes the value $RID_{Sj}^* = M_2 \oplus h(TC_{Ui} \| ID_i \| T_1)$.
Step 9. $\mathcal{A}$ computes the message $M_4^* = h(ID_i^* \| RID_{Sj}^* \| TC_{Ui} \| r_1^* \| T_1)$.
Step 10. Repeat step 1 ∼9 until PW_i^*, ID_i^* meet M_4^*=M_4.

The time complexity: $\mathcal{O}((7T_H + T_B + 5T_{XOR}) \times |\mathcal{D}_{ID}| \times |\mathcal{D}_{PW}|)$. This attack is similar to the above attack. The adversary can obtain the victim's password in polynomial time, which breaks the multi-factor security of the protocol.

C. No Forward Secrecy

For a secure authentication scheme, forward secrecy is a significant feature, which concerns the security of the previous session key. Even if the adversary can get the long-term private key of the gateway/server, she cannot calculate any previous session key [16]. We reveal that Chaudhry et al.'s scheme [25] cannot achieve forward secrecy.

Step 1. $\mathcal{A}$ uses M_1 to extract TID_{Ui}, WID_{Ui}.
Step 2. $\mathcal{A}$ computes $K_i = WID_{Ui} \oplus h(M_1 \| X)$.
Step 3. $\mathcal{A}$ computes $R_i = h(ID_i \| X)$.
Step 4. $\mathcal{A}$ computes $RID_{Ui} = h(ID_i \| R_i)$.
Step 5. $\mathcal{A}$ computes $TC_{Ui} = h(ID_i \| K_i)$.
Step 6. $\mathcal{A}$ computes $r_1 = M_3 \oplus h(RID_{Ui} \| TC_{Ui} \| T_1)$.
Step 7. $\mathcal{A}$ computes the hash value $h(RID_{Ui} \| TC_{Ui} \| r_1)$=$M_7 \oplus h(TC_{Sj} \| T_2)$.
Step 8. $\mathcal{A}$ uses the value of Step 7 to compute $r_3 = M_{10} \oplus h(h(RID_{Ui} \| TC_{Ui} \| r_1) \| T_3)$.
Step 9. $\mathcal{A}$ computes $SK_{ij} = h(h(h(RID_{Sj} \| K_j) \| T_3) \| h(RID_{Ui} \| TC_{Ui} \| r_1) \| RID_{Sj} \| r_3 \| T_3)$.

The time complexity: $\mathcal{O}(10T_H + 4T_{XOR})$. As in the above attack, adversaries can obtain the session key of any previous session. The protocol cannot achieve forward secrecy.

D. Node Capture Attack

Since sensor nodes are usually unattended, the adversary can use physical attacks [6], [7] to obtain the secret information of the sensor node. Under this assumption, we find that Chaudhry et al.'s scheme [25] cannot resist node capture attacks. Then the adversary can obtain all previous session keys. The specific attack steps are as follows.

Step 1. $\mathcal{A}$ computes $r_2 = M_6 \oplus h(TC_{Sj} \| RID_{Sj})$.
Step 2. $\mathcal{A}$ computes the hash value $h(RID_{Ui} \| TC_{Ui} \| r_1) = h(TC_{Sj} \| T_2) \oplus M_7$.
Step 3. $\mathcal{A}$ uses the value of Step 2 to compute $r_3 = M_{10} \oplus h(h(RID_{Ui} \| TC_{Ui} \| r_1) \| T_3)$.
Step 4. $\mathcal{A}$ computes $SK_{ij} = h(h(h(RID_{Sj} \| K_j) \| T_3) \| h(RID_{Ui} \| TC_{Ui} \| r_1) \| RID_{Sj} \| r_3 \| T_3)$.

The time complexity: $\mathcal{O}(6T_H + 3T_{XOR})$. The adversary can calculate the session key by the above attack in polynomial time, making the security of the protocol invalid.

V. Review of Jabbari-Mohasef's Scheme

In 2022, Jabbari and Mohasef [26] claimed that it is necessary to ensure the security and privacy of the information collected by the end devices. It will facilitate the use of LoRaWAN and gain users' trust. However, no suitable authentication scheme could be used in LoRaWAN networks. Thus, Jabbari and Mohasef propose a new multi-factor authentication scheme for the LoRaWAN network. In this section, a brief review of their proposed scheme [26] is provided. The intuitive notations and abbreviations are listed in Table I.

A. Registration Phase

1) User Registration

Step 1. $U_i \Rightarrow NS$:$\{MID_i\}$.
U_i chooses the identity ID_i and password PW_i. She computes MID_i =$aes128\_cmac$ $(\pi_i, ID_i|\rho_i)$ and sends it to the network server NS.
Step 2. $NS \Rightarrow U_i : \{A_i, TID_i\}$.

NS chooses a random identity TID_i for U_i and computes $A_i = aes128\_cmac(s, TID_i) \oplus MID_i$, where s is the long-term secret key of NS. Then NS computes $CMID_i{}' = MID_i \oplus s$, $CMID_i = aes128\_cmac(s, CMID_i{}')$ and generates a counter $SUCnt_i = 0$. It keeps $SUCnt_i$ and $CMID_i$ in its database for auditing and sends A_i, TID_i to U_i.

Step 3. When receiving the message, U_i generates a counter $USCnt_i = 0$. Furthermore, U_i imprints her biometrics B_i at SMD_i. Then she calculates $Gen(B_i) = (\beta_i, \alpha_i)$, $K_i = aes128\_cmac\ (\beta_i, ID_i|\ PW_i)$ and $C_i = aes128\_encrypt(K_i, \pi_i|\rho_i|MID_i||\ ID_i|\ A_i|TID_i|USCnt_i)$. SMD_i stores C_i and α_i.

2) Offline End-Device Registration Phase

The one who could connect and configure E_j, computes $K_{ij} = aes128\_cmac(\pi_i, DevEUI_j|\rho_i)$ and load it to E_j offline. E_j generates a counter $EUCnt_j = 0$. Then the E_j stores K_{ij} and $EUCnt_j$ in its memory securely. At the end, the user generates a new number $UECnt_j = 0$ and appends $UECnt_j$ to the value of C_i in her SMD_i.

B. Login and Authentication Phase

Step 1. $U_i \rightarrow NS : M_1 = \{TID_i', C_{in}\}$.
First, U_i inputs ID_i, PW_i and biometrics B_i. SMD_i computes $\beta_i = Rep(B_i, \alpha_i)$ and $K_i = aes128\_cmac\ (\beta_i,\ ID_i|\ PW_i)$. Then it computes $aes128\_decrypt\ \ (K_i, C_i) = (\pi_i'|\rho_i'|MID_i'|ID_i'|\ A_i'|TID_i'|USCnt_i'|UECnt_i')$. And it checks if $ID_i' = ID_i$. If so, SMD_i computes $K_{ij} = aes128\_cmac(\pi_i', DevEUI_j|\rho_i')$. Then SMD_i computes $C_{ij}^1 = aes128\_decrypt(K_{ij}, \lambda_i)$. It gets $UECnt_i' = UECnt_i' + 1$, $ECnt_j = UECnt_i'$ and $C_{ij}^2 = aes128\_decrypt\ \ (\lambda_i,\ DevEUI_j|ECnt_j|\ pad_{16})$. Afterward, U_i computes $USCnt_i' = USCnt_i' + 1$, $\ Cnt_i = USCnt_i'$, $\ K_{in} = A_i' \oplus MID_i'$, and $C_{in} = aes128\_encrypt(K_{in}, MID_i'|\ Cnt_i|C_{ij}^1|C_{ij}^2|DevEUI_j)$. Due to the Cipher Block Chaining (CBC) mode of the AES encryption, the user needs to update the values of $UECnt_i'$ and $USCnt_i'$. And the counter Cnt_i is used to resist the replay attack. Then U_i delivers M_1 to NS.

Step 2. $NS \rightarrow E_j : M_2 = \{C_{nj} : KeyEstablishReq\}$.
Upon receiving M_1, NS employs s and TID_i' to get $K_{ni} = aes128\_cmac(s, TID_i')$. Subsequently, it decrypts C_{in} as $(MID_i'|Cnt_i'|C_{ij}'^1|\ C_{ij}'^2|DevEUI_j') = aes128\_decrypt(K_{ni}, C_{in})$, $CMID_i'' = MID_i' \oplus s$, $CMID_i' = aes128\_cmac\ (s, CMID_i'')$ and checks if $CMID_i'$ is equal to the stored $CMID_i$. If it does not hold, NS rejects the request. Otherwise, it checks $Cnt_i' > SUCnt_i$. Then it selects a new random number, λ_n, and use $NwkSEncKey_{nj}$ to encrypts $(CID|C_{ij}'^1|C_{ij}'^2|\ DevEUI_j'|\ \lambda_n)$, which is the key shared between NS and E_j. Then it generates a Message Integrity Code (MIC) for this message to provide the integrity of the message. Finally, NS produces the result C_{nj} and sends the message M_2 to E_j.

Step 3. $E_j \rightarrow NS : M_3 = \{C_{jn} : KeyEstablishAns\}$.
The end-device E_j uses the key $NwkSEncKey_{jn}$ to get $(CID'|C_{ij}'^1|C_{ij}'^2|DevEUI_j'|\lambda_n')$. Then, it use K_{ji} to decrypt $C_{ij}'^1$ as $(\lambda_i') = aes128\_encrypt(K_{ji}, C_{ij}'^1)$. It also computes $C_{ij}'^2$ as $(DevRUI_j''|ECnt_j'|pad_{16}) = aes128\_encrypt(\lambda_i', C_{ij}'^2)$. It gets the $DevEUI_j''$ and checks if $DevEUI_j'' = DevEUI_j' = DevEUI_j$ and $ECnt_j' > EUCnt_j$. If it holds, this request is initiated by U_i. Then the E_j uses random number λ_j to computes $EUCnt_j = ECnt_j'$, $SK_{ji} = \ \lambda_i' \ \oplus\lambda_j$, $\ C_{ji}^1 = aes128\_encrypt(K_{ji} \oplus \lambda_i', \lambda_j)$ and $C_{ij}^2 = aes128\_encrypt(\lambda_j, SK_{ji})$. It encrypts $(C_{ji}^1|C_{ji}^2|\lambda_n')$ by using $NwkSEncKey_{jn}$, computes the message's MIC value and gets the result C_{jn}. Furthermore, E_j sends $M_3 = \{C_{jn} : KeyEstablishAns\}$ to NS.

Step 4. $NS \rightarrow U_i : M_4 = \{C_{ni}\}$.
NS uses $NwkSEncKey_{nj}$ to decrypt the message and check the MIC value. Thus, NS gets $(C_{ji}^1|C_{ji}^2|\lambda_n')$. Then, it verifies if λ_n'=λ_n. If so, NS selects a new temporary identity TID_i^{new} and gets $A_i^{new} = aes128\_cmac(s, TID_i^{new}) \oplus MID_i$, $SUCnt_i = Cnt_i'$ and $C_{ni} = aes128\_encrypt\ (K_{ni}, C_{ji}^1|\ C_{ji}^2|\ TID_i^{new}|\ A_i^{new}|Cnt_i')$. NS sends the message $M_4 = \{C_{ni}\}$ to U_i.

Step 5. The user U_i computes $aes128\_decrypt(K_{in}, C_{ni}) = (C_{ji}'^1|\ C_{ji}'^2|TID_i^{'new}|A_i^{'new}|Cnt_i')$ and checks whether $Cnt_i' = USCnt_i'$. If it holds, she decrypts $(\lambda_j') = aes128\_decrypt(K_{ij} \oplus \lambda_i, C_{ji}'^1)$ and $(SK_{ji}') = aes128\_decrypt(\lambda_j', C_{ji}'^2)$. Then she computes $SK_{ij} = \lambda_i \oplus \lambda_j'$ and checks whether $SK_{ij} = SK_{ji}'$. Then user establishes a session key with E_j. U_i computes $C_i^{new} = aes128\_encrypt(K_i, \pi_i'|\ \rho_i'|MID_i'|ID_i'|A_i^{'new}|\ TID_i^{'new}|USCnt_i'|UECnt_i')$ and replaces C_i with C_i^{new}.

C. Biometric and Password Update Phase

Step 1. U_i first inputs ID_i, PW_i and B_i. Then SMD_i computes $\beta_i = Rep(B_i, \alpha_i)$ and $K_i = aes128\_cmac\ (\beta_i, ID_i|PW_i)$, and decrypts $(\pi_i'|\rho_i'|MID_i'|ID_i'|\ A_i'|\ TID_i'|\ USCnt_i'|\ UECnt_j') = aes128\_decrypt\ (K_i, C_i)$, and verifies whether $ID_i' = ID_i$ and $MID_i' = aes128\_cmac\ (\pi_i', ID_i|\rho_i')$.

Step 2. U_i enters new PW_i^{new} and B_i^{new}. SMD_i calculates $Gen(B_i^{new}) = (\beta_i^{new}, \alpha_i^{new})$, $K_i^{new} = aes128\_cmac\ (\beta_i^{new}, ID_i|\ PW_i^{new})$ and C_i^{new}. Finally, SMD_i replaces C_i, α_i with C_i^{new}, α_i^{new}.

VI. Analysis of Jabbari-Mohasef's Scheme

Jabbari and Mohasef [26] claimed that their protocol could achieve the coexistence of efficiency and security. However, we reveal that their scheme [26] fails to meet the fundamental goals of a secure multi-factor authentication protocol.

A. No Truly Multi-Factor Security

We reveal that Jabbari-Mohasef's scheme [26] fails to achieve truly multi-factor security. The password can be guessed if an adversary obtains the user's biometric information and smart card. The specific attack steps are as follows.

Step 1. $\mathcal{A}$ computes $Gen(B_i) = (\beta_i, \alpha_i)$.
Step 2. $\mathcal{A}$ chooses ID_i^*, PW_i^* from $\mathcal{D}_{id} \times \mathcal{D}_{pw}$.
Step 3. $\mathcal{A}$ computes $K_i = aes\_cmac(\beta_i, ID_i^* | PW_i^*)$.
Step 4. $\mathcal{A}$ further computes $(\pi_i' | \rho_i' | MID_i' | ID_i^* | A_i' | TID_i' | USCnt_i' | UECnt_i') = aes\_decrypt(K_i, C_i)$.
Step 5. Repeat step 1∼4 until ID_i^* meets $(ID_i^* = ID_i)$.

The time complexity: $\mathcal{O}((T_B + T_{aesdec} + T_{aesmac}) \times |\mathcal{D}_{ID}| \times |\mathcal{D}_{PW}|)$, where T_{aesmac} is the time of AES-CMAC algorithm, and T_{aesdec} is the time of AES decryption algorithm. Similar to Section IV-A, the offline password guessing attack can obtain the password in polynomial time, thereby breaking the multi-factor security of Jabbari-Mohasef's scheme.

B. Node Capture Attack

We also perform a node capture attack and demonstrate that Jabbari-Mohasef's scheme [26] fails to resist such an attack. Specifically, an adversary can compute session keys for all previous communications between the user and the sensor node. The specific attack details are as follows.

Step 1. $\mathcal{A}$ decrypts $C_{nj} = (CID | C_{ij}'^1 | C_{ij}'^2 | DevEUI_j' | \lambda_n')$ using $NwkSEncKey_{nj}$.
Step 2. $\mathcal{A}$ computes $\lambda_i' = aes\_encrypt(K_{ji}, C_{ij}'^1)$.
Step 3. $\mathcal{A}$ decrypts $C_{jn} = (C_{ji}'^1 | C_{ji}'^2 | \lambda_n')$.
Step 4. $\mathcal{A}$ decrypts $(\lambda_j') = aes\_decrypt(K_{ij} \oplus \lambda_i, C_{ji}'^1)$.
Step 5. $\mathcal{A}$ decrypts $(SK_{ji}') = aes\_decrypt(\lambda_j', C_{ji}'^2)$.

The time complexity: $\mathcal{O}(4T_{aesdec} + T_{aesenc})$, where T_{aesenc} is the time of AES encryption. It is clear that when the nodes are attacked, the protocol cannot maintain forward security.

C. Denial of Service Attack

Jabbari-Mohasef's scheme [26] allows users to locally update their passwords or biometrics, providing a convenient feature for users. However, we reveal that this functionality makes the scheme vulnerable to denial of service (DoS) attacks. The specific attack details are as follows.

Step 1. $\mathcal{A}$ computes $Gen(B_i) = (\beta_i, \alpha_i)$.
Step 2. $\mathcal{A}$ chooses ID_i^*, PW_i^* from $\mathcal{D}_{id} \times \mathcal{D}_{pw}$.
Step 3. $\mathcal{A}$ computes $K_i = aes\_cmac(\beta_i, ID_i^* | PW_i^*)$.
Step 4. $\mathcal{A}$ uses K_i to compute $aes\_decrypt(K_i, C_i) = (\pi_i' | \rho_i' | MID_i' | ID_i | A_i' | TID_i' | USCnt_i' | UECnt_i')$.
Step 5. Repeat step 1∼4 until ID_i^* meets $(ID_i^* = ID_i)$.
Step 6. $\mathcal{A}$ inpus the parameters PW_i, ID_i, B_i and initiates an update request.
Step 7. $\mathcal{A}$ inputs B_i^{new} and PW_i^{new}.

The time complexity: $\mathcal{O}((T_B + T_{aesdec} + T_{aesmac}) \times |\mathcal{D}_{ID}| \times |\mathcal{D}_{PW}|)$. An adversary can update a user's unknown password and biometric information, causing the smart device to deny service to the legitimate user. This is a common vulnerability in protocols that allow local password updates.

VII. Our Proposed Scheme

After analyzing the two protocols [25], [26], we first categorize the identified issues. It is worth noting that security issues are not limited to the analyzed protocols but are also prevalent in other recent protocols. Therefore, we analyze their fundamental causes and propose effective countermeasures. Subsequently, we re-examine the common issues observed in these failed protocols and propose a new two-factor password-based authentication protocol for WSNs.

A. Countermeasures of Existing Defects

After analysis, we find both Chaudhry et al.'s [25] and Jabbari-Mohasef's scheme [26] have the following defects.

1) Vulnerable to Offline Password Guessing Attacks: Multi-factor security is defined such that when the adversary obtains $n-1$ factors of an n-factor authentication protocol, the protocol remains secure [16]. Therefore, it is reasonable to assume that the adversary can obtain all factors except the password. Under this assumption, both schemes [25], [26] fail to resist offline password guessing attacks (see Sections IV-A, IV-B, and VI-A) and cannot achieve truly multi-factor security. As we know, an offline password guessing attack refers to a scenario where the adversary can gain a verifier in which the only two unknown parameters are the password and the identity number. All other parameters are either known or can be deduced from the password, identity, and additional known parameters [49]. Using the value of the verifier, the adversary can determine the correctness of guessed passwords. Since the attack process is offline, it requires no additional equipment and remains undetectable to others. From the protocol design perspective, adversaries can obtain the verifier through different methods to launch attacks. Attacks on Chaudhry et al.'s [25] and Jabbari-Mohasef's [26] schemes can be categorized into two types.

For the first one, to enable functionalities like "local password update" or "timely typo detection", the smart card must store the verifier to authenticate the user identity. However, protocols that store the verifier in the smart card are vulnerable to offline password guessing attacks. Adversaries can obtain the verifier (e.g., through a side channel attack on the smart card [50]) and verify the correctness of the guessed password. As demonstrated in our analysis, Chaudhry et al. [25] and Jabbari-Mohasef [26] neither consider this problem nor employ effective countermeasures to prevent such attacks. Thus, their schemes fail to resist this type of attack (see Sections IV-A and VI-A).

To reconcile the need for local password updates with the risk of offline password guessing attacks caused by smart card storage, Wang et al. [16] proposed the "fuzzy-verifier" technique, which can be employed to solve the above offline guessing attack. Specifically, for Chaudhry et al.'s protocol [25], the verifier $F_i^* = h(RPW_i^* \parallel \sigma_i^* \parallel R_i \parallel TC_{Ui})$. We reset $F_i^* = h(RPW_i^* \parallel \sigma_i^* \parallel R_i \parallel TC_{Ui}) \bmod n_0$, where n_0 is an integer and $n_0 \in (2^4, 2^8)$. The adversary can find $|\mathcal{D}_{\text{ID}} \times \mathcal{D}_{PW}|/n_0 \approx 2^{32}$ pairs of (ID_i^*, PW_i^*), which satisfy the equation $F_i = F_i^*$. However, there is only one correct pair of correct identity number and password. If the adversary desires to know whether the guessed password is correct, she must initiate an online request

to the gateway. The adversary's advantage is negligible, as she must sift through 2^{32} possible answers via online requests to the gateway. Thus, the "fuzzy-verifier" effectively solves this type of offline password guessing attack. Similarly, this approach can be applied to Jabbari-Mohasef's protocol [26]. Due to space limitations, it will not be repeated here.

For the second type of offline password guessing attack, the adversary can obtain the verifier by eavesdropping on information transmitted over public channels (see Section IV-B). In 1999, Halevi and Krawczyk [51] pointed out that traditional single-factor password authentication is not immune to offline password guessing attacks if public-key technology is not employed. Later, Ma et al. [24] proposed that all security parameters stored in a smart card could be compromised. Then, smart card based password authentication schemes are downgraded to traditional single-factor password authentication schemes. In such cases, the security of the scheme relies only on the password. To address this issue, Ma et al. proposed "The public-key principle", asserting that public-key cryptography is essential for two-factor authentication schemes to resist offline password guessing attacks. And they pointed out a large number of schemes as evidence. Furthermore, for n-factor authentication scheme, we assume that the adversary can get $n-1$ factors except for the password. Under this assumption, n-factor authentication schemes degrade to traditional single-factor password-based authentication schemes. Ma et al.'s [24] principle is therefore applicable to multi-factor authentication as well.

To address the issue in Chaudhry et al.'s scheme [25], public-key technology can be employed as a remedy. U_i first chooses a random number a_1 and uses GWN's public key $y = g^x$ to compute $X_1 = g^{a_1}, X_2 = y^{a_1}$. Then U_i generates the verifier $M_4' = h(ID_i \parallel RID_{Sj} \parallel TC_{Ui} \parallel r_1 \parallel T_1 \parallel X_2)$ and $Msg_1 = \{M_1, M_2, M_3, M_4, X_1, T_1\}$. In this way, in addition to the password and identity, X_2 remains unknown to the adversary. Since X_2 can only be computed by the gateway node GWN, the adversary cannot use M_4 to verify the correctness of the guessed password. Public-key encryption technology effectively prevents this type of offline password guessing attack, and it can be applied to similar protocols.

2) Vulnerable to Node Capture Attacks: We demonstrate that Chaudhry et al.'s [25] and Jabbari-Mohasef's [26] protocols are subject to node capture attacks (see Sections IV-D and VI-B). The adversary can obtain the secret key of the node through the node capture attack. Then she can use it to obtain previous session keys, which breaks the protocol's forward secrecy. The user usually sends a random number to the gateway to establish a session key. The gateway will make the random number XOR the node's secret parameters, and then send it to the node. After that, the node makes its random number XOR the secret and sends it to the gateway. In this process, the gateway can obtain all the parameters to calculate the session key. It is known that the gateway and the node only use the shared secret to protect the parameter. The gateway is legal and credible, but the adversary can obtain the shared secret by breaking the sensor node. If an adversary captures a sensor node, she can extract the shared secret and use it to derive all parameters related to the session key, as described in [7]. To mitigate this vulnerability, it is essential to ensure that neither the gateway nor the adversary can compute the secret parameters solely based on the shared secret.

Thus, we propose a design where the gateway verifies session key-related content without being able to compute the session key itself. Public-key cryptography can help us realize this design. Specifically, the user chooses a random number r_1 and computes $X_1 = r_1 \cdot P$, where P is a point on the elliptic curve. Similarly, the node chooses a new random number r_2 and computes $X_2 = r_2 \cdot P$. They send X_1, X_2 to each other through the gateway. The session key is then constructed as $X_3 = r_1 \cdot r_2 \cdot P$. The gateway only verifies identity and data integrity. Importantly, because of the hardness of the elliptic curve discrete logarithm problem, neither the adversary nor the gateway can derive X_3 from X_1 and X_2. As a result, this approach effectively resists node capture attacks and ensures the forward secrecy of the protocol. Besides, we find that node capture attacks can compromise user anonymity [52]. Thus, the gateway should avoid transmitting messages containing user identity information when forwarding user requests to the node.

3) Synchronization Mechanism: Both protocols employ the synchronization mechanism to achieve user anonymity. While time synchronization is a commonly employed technique, it is not recommended due to its inherent vulnerabilities. As Wang et al. highlighted in [36], synchronization mechanisms may lead to de-synchronization attacks or other problems (see Luo et al.'s attack [53] on Gope et al.'s scheme [54]). In addition to synchronization mechanisms, public-key cryptography can also be employed to achieve user anonymity [20]. Beyond anonymity, public-key technology can resist offline password guessing attacks and ensure forward secrecy, as demonstrated in the aforementioned methods. Therefore, public-key cryptography can serve as a direct solution for achieving anonymity. Besides, Chaudhry et al. [25] use a time synchronization mechanism to ensure the freshness of the scheme. Although it is effective, it will increase the complexity and energy consumption of the protocol. A more efficient alternative is the use of random numbers to ensure message freshness and prevent replay attacks, which can be employed in almost any protocol.

Building on the previous analysis, we propose a new protocol that adopts the elliptic curve public-key algorithm. To resist offline dictionary guessing attacks, the protocol integrates the "fuzzy-verifier" mechanism [16] and the "honeywords" approach [55]. Furthermore, we implement elliptic curve multiplication on the sensor node and the user side to achieve forward secrecy and user anonymity. The specific details are as follows. The intuitive notations and abbreviations are listed in Table III. For a better understanding of our scheme, the user registration phase and the login and authentication phase are illustrated in Fig. 2 and Fig. 3, respectively.

B. Initialization Phase

The protocol includes three types of participants: user U_i, gateway GWN, and sensor node S_j. Our proposed scheme is built on an elliptic curve E (generated by a base point P with a large prime order p) defined over a prime finite field $\mathbb{F}_p$. The gateway chooses its long-term secret key X_{GWN}, which

TABLE III
NOTATIONS AND ABBREVIATIONS IN OUR PROTOCOL

Symbol	Description	Symbol	Description
U_i	the i^{th} user	S_j	the j^{th} sensor node
GWN	the gateway	SC_i	smart card of the U_i
$\mathcal{A}$	the adversary	$HoneyList$	a list recording forged keys
ID_i	identity of U_i	DID_i	dynamic identity of U_i
SID_j	identity of S_j	$h(\cdot)$	one-way hash function
PW_i	password of U_i	X_{GWN}	GWN's long-term secret key
SK	the session key	Y	GWN's public key
$\rightarrow$	an insecure channel	$\|$	concatenation operation
$\Longrightarrow$	a secure channel	$\oplus$	bitwise XOR operation

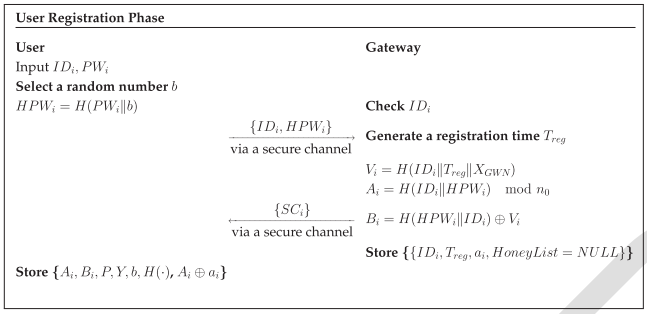

Fig. 2. The user registration phase of our proposed scheme.

is known only to GWN. P is shared with all servers. The GWN computes its public key as $Y = P \cdot X_{GWN}$, which is made publicly available. Each node S_j has been registered and shared the secret $P_j = H(SID_j \| X_{GWN})$ with GWN.

C. Registration Phase

R1. $U_i \Rightarrow GWN$:$\{ID_i, HPW_i\}$.
First, the user U_i chooses an identity and password pair $\{ID_i, PW_i\}$ and generates a random number b. Then, user calculates $HPW_i = H(PW_i||b)$ and sends the register message $\{ID_i, HPW_i\}$ to the gateway GWN.

R2. $GWN \Rightarrow U_i$: SC_i:$\{A_i, B_i, P, Y, H(\cdot), A_i \oplus a_i\}$.
The gateway GWN first checks whether the ID_i is in the database. Then, it computes $V_i = H(ID_i||T_{reg}||X_{GWN})$, where T_{reg} is the registration time of the user and X_{GWN} is the long-term secret key of the GWN. And it selects a random number a_i and computes $A_i = H(ID_i||HPW_i) \bmod n_0$, $B_i = H(HPW_i||ID_i) \oplus V_i$. The GWN stores $\{A_i, B_i, P, Y, H(\cdot), A_i \oplus a_i\}$ in smart card SC_i and sends it to user. The GWN stores $\{ID_i, T_{reg}, a_i, HoneyList = NULL\}$ into the database, where $HoneyList$ is a list stored in *GWN* specifically designed to record forged keys generated in response to potential attacks.

R3. The user receives SC_i: $\{A_i, B_i, P, Y, H(\cdot), A_i \oplus a_i\}$ and stores b into the smart card SC_i.

D. Login and Authentication Phase

L1. $U_i \rightarrow GWN$:$\{C_1, C_2, X_1, DID_i\}$.
U_i first enters her identity ID_i and password PW_i. Then the smart card SC_i computes $HPW_i^* = H(PW_i^*||b)$ and $A_i^* = H\ (ID_i||HPW_i^*) \bmod n_0$. The SC_i verifies the user by checking whether $A_i^* = A_i$. If $A_i^* \neq A_i$, the smart card SC_i will reject the user's login request. Otherwise, SC_i selects the corresponding sensor nodes SID_j and two random numbers r_1, r_2. It computes $a_i = A_i \oplus A_i \oplus a_i$, $X_1 = r_1 \cdot P$, $X_2 = r_1 \cdot Y$, $V_i = B_i \oplus H(HPW_i||ID_i)$, $DID_i = ID_i \oplus H(X_2)$, $C_1 = (SID_j||r_2||a_i) \oplus H(X_2||X_1)$, $C_2 = H(C_1||DID_i||X_2||V_i||SID_j)$, sends $\{C_1, C_2, DID_i, X_1\}$ to GWN.

L2. $GWN \rightarrow S_j$:$\{C_3, C_4, X_1\}$.
When the GWN receives the login request, it first computes $X_2^* = X_1 \cdot X_{GWN}$, $ID_i^* = DID_i \oplus H(X_2)$, $V_i^* = H(ID_i^*||T_{reg}||X_{GWN})$, $(SID_j^*||r_2^*||a_i^*) = C_1 \oplus H(X_2^*||X_1)$. Then GWN first checks if $a_i^* = a_i$. If it is not equal, GWN rejects this session. Otherwise, GWN computes $C_2^* = H(C_1||DID_i||X_2^*||V_i^*||SID_j^*)$ and checks whether $C_2^* = C_2$. If it is not, GWN will reject this session and insert C_2^* into $HoneyList$ when there are fewer than N_i (e.g., N_i=10) items in $HoneyList$. If there are N_i items in $HoneyList$, GWN will suspend U_i's smart card SC_i until U_i re-registers. If $C_2^* = C_2$, GWN uses X_{GWN} to get $P_j = H(SID_j||X_{GWN})$. Then it computes $C_3 = r_2 \oplus H(SID_j||P_j)$, $C_4 = H(r_2||P_j||\ SID_j||X_1)$ and transmits $\{C_3, C_4, X_1\}$ to S_j.

L3. $S_j \rightarrow GWN$:$\{C_5, C_6, X_3\}$.
On receiving the message, S_j first uses P_j, SID_j to get $r_2^* = H(SID_j||P_j) \oplus C_3$. Then it computes $C_4^* = H(\ r_2||P_j||SID_j||X_1)$ and compares whether $C_4^* = C_4$. The node will reject the request if it is not equal. Otherwise, it chooses a random number r_3 and computes $X_3 = r_3 \cdot P$, $X_4 = r_3 \cdot X_1$, $SK = H(X_1||\ X_3||X_4||r_2||\ SID_j)$, $C_5 = H(SK||r_2||X_1||X_4)$, $C_6 = H(r_2||X_3||\ C_5||X_1||SID_j)$ and sends message $\{C_5, C_6, X_3\}$ to the GWN.

L4. $GWN \rightarrow U_i$:$\{C_5, C_7, X_3\}$.
The GWN first authenticates the sensor node S_j by checking whether $C_6 = H(r_2||X_3||C_5||X_1||SID_j)$. If so, it computes $C_7 = H(r_2||X_2||X_3||V_i||SID_j||C_5)$ and sends $\{C_5, C_7,\ X_3\}$ to the user. Otherwise, the gateway node GWN will end the session.

L5. When receiving the reply from gateway, U_i first calculates $C_7^* = H(r_2||X_2||X_3||V_i||SID_j||C_5)$ to authenticate the GWN by checking whether $C_7^* = C_7$. If not, U_i will terminate the conversation. Otherwise, U_i computes $X_4^* = X_3 \cdot r_1$, $SK = H(X_1||X_3||X_4^*||r_2||SID_j)$. If $C_5^* = H(SK||r_2||X_1||X_4)$, U_i and S_j successfully authenticate each other and establish a session key SK to ensure the security of the subsequence conversation.

E. Password Change Phase

P1: If U_i desires to update a new password, she inputs ID_i, the previous PW_i and the new password PW_i^{new}.

P2: The smart card SC will compute $HPW_i = H(PW_i||b)$ and $A_i^* = H(ID_i||HPW_i)$ to check the user identity.

P3: If $A_i^* = A_i$, the smart card SC_i computes $HPW_i^{new} = H(PW_i^{new}||b)$, $A_i^{new} = H(ID_i||HPW_i^{new})$, and $B_i^{new} = B_i \oplus H(HPW_i||ID_i) \oplus H(HPW_i^{new}\ ||ID_i)$

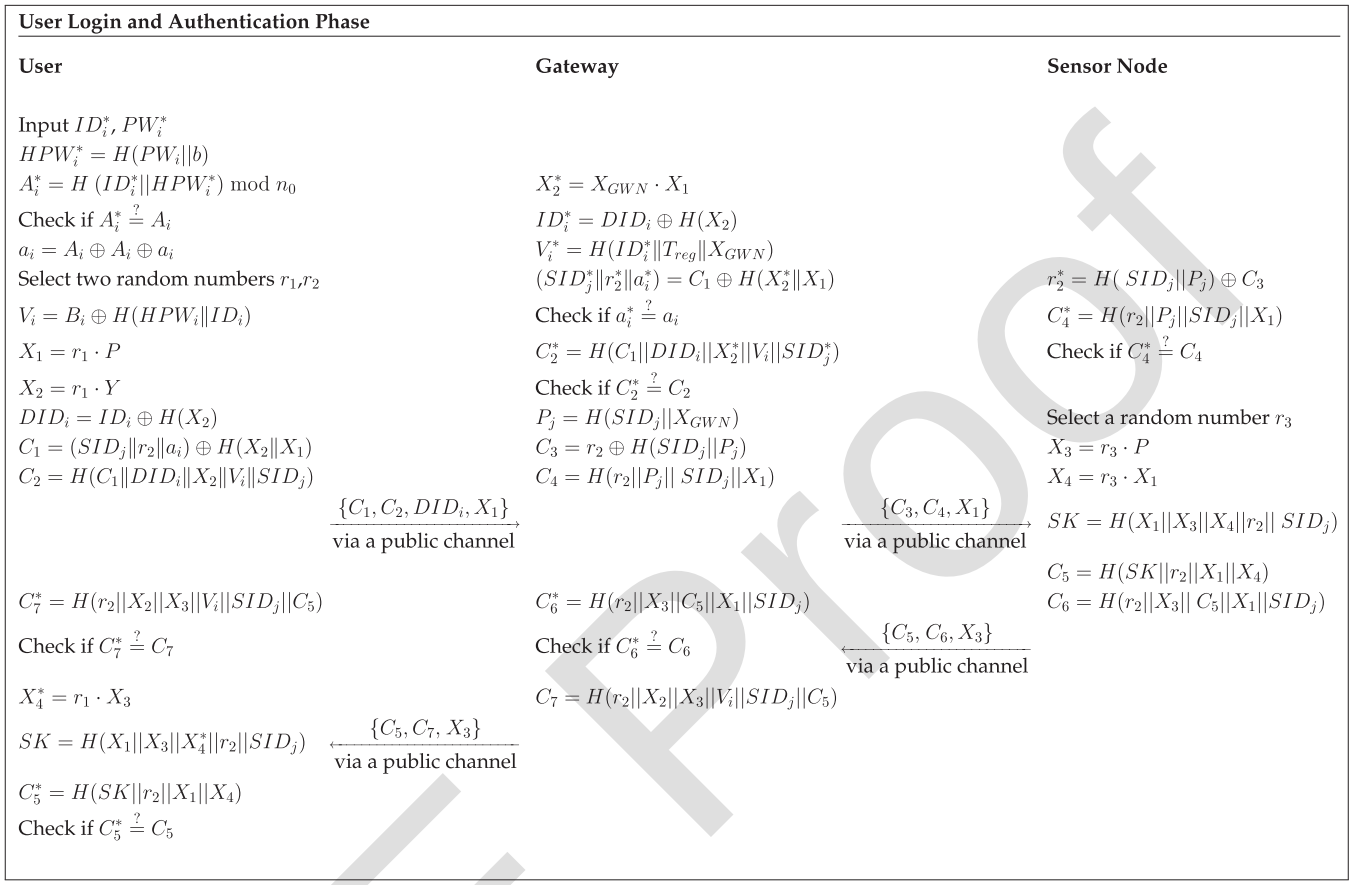

Fig. 3. The login and authentication phase of our proposed scheme.

and replaces $\{A_i, B_i\}$ with $\{A_i^{new}, B_i^{new}\}$. Otherwise, the smart card will reject this request.

F. Dynamic Node Addition Phase

D1. The new sensor node sends the registration request to the gateway node GWN.

D2. The GWN chooses a new SID_{new} for new sensor node. Then GWN uses X_{GWN} to compute $P_{new} = H(SID_{new} || X_{GWN})$ and send $\{SID_{new}, P_{new}\}$ to the new sensor node S_{new}.

D3. The new sensor node will keep P_{new} as its private key.

VIII. Security Analysis

In this section, we define the security model and prove the security of the proposed protocol under the ROM.

Participants. In our protocol $\mathcal{P}$, we have three types of participants : the user $U \in$ **User**, the $GWN \in$ **Gateway** and the sensor node $S \in$ **Sensor˜node**. We use U^i to denote the ith user. Similarly, GWN denotes the Gateway and S^j denotes the jth sensor node. Let I denote any instances, which means $I \in (U \cup GWN \cup S)$.

Long-term keys. Before authentication phase, there will be an initialization and registration phase to establish public parameters and security foundations. The GWN owns a long-term secret key X_{GWN} and the corresponding public key Y. The sensor node S owns an identity SID_S and a long-lived key P_S. The user U owns an identity ID_U and a password PW_U. Specifically, each PW_U is independently sampled from the password space $D(\lambda)$ according to the Zipf's law [47].

Queries. The adversary $\mathcal{A}$ can use oracle queries to interact with the participants of the protocol. The capabilities of the adversary $\mathcal{A}$ in the real attack are captured by oracle queries. The query types are as follows:

- **Execute** (U^i, GWN, S^j). This query models a passive attack (eavesdropping). The output of this query is the transcript of an honest execution among the three instances the user U^i, the gateway *GWN*, the sensor node S^j.
- **Send** (I, m). This query models an active attack of $\mathcal{A}$. It means that $\mathcal{A}$ sends the message m to the instance I. On receiving the message m from **Send** query, I executes according to the protocol and sends the response to $\mathcal{A}$.
- **Test** (I). This query does not model the adversary $\mathcal{A}$'s capability but defines semantic security of the session key, which is only valid for "fresh" sessions. If the session key of the instance U^i is not defined, the answer is $\perp$. Otherwise, the oracle randomly selects a bit b. If $b = 1$, it returns the session key to the adversary $\mathcal{A}$; If $b = 0$, it returns a random string of the same length to $\mathcal{A}$ as the session key. This query is allowed only once.

– **Reveal** (I). This query is used to model the known-key attack where I is user U or sensor node S. If the instance I has negotiated a session key SK and not asked by **Test** (I), it outputs SK to $\mathcal{A}$. Otherwise, it outputs $\perp$.
– **Corrupt** $(U, 1)$. According to the capability C7 in Section II, $\mathcal{A}$ can break one factor of the two-factor protocol. This query is used to model that $\mathcal{A}$ can obtain the U's password. The output is the user U's password PW_U.
– **Corrupt** $(U, 2)$. Based on the capability C7, the adversary can also break another factor of the protocol. This query is used to model that $\mathcal{A}$ can obtain the smart card SC and the parameters stored in SC. The output of this query is parameters in the user's smart card.
– **Corrupt** $(GWN, 1)$. This query models the adversary's capability C5. The output of this query is GWN's long-term secret key X_{GWN}.
– **Corrupt** $(S, 1)$. This query models the adversary's capability C6. The output of this query is the shared secret P_S of the sensor node S.

Partnering. We use the session identifier sid and partner identifier pid to define partnering. We say that the two instances U^i and S^j are partnering if all the following conditions are met. (1) Both instances have accepted; (2) The session identifier $sid_{U^i} = sid_{S^j}$; (3) The partner identifier $Pid_{U^i} = S^i$ and $Pid_{S^j} = U^i$. The sid is denoted the order concatenation of all messages sent and received by the instance U^i (or S^j).

Freshness. Assuming that $\mathcal{A}$ can execute **Test** (I) and **Reveal** (I) queries at the same time, then $\mathcal{A}$ can easily obtain the session key and break the semantic security of the protocol. The notion of freshness characterizes the fact that an adversary cannot easily obtain the session key. The instance I is fresh if: (1) I has been accepted and had a session key. (2) I and its partner have not been asked for **Reveal** (I) query. (3) The adversary is allowed to ask one kind of **Corrupt**-query for U or its partner S and no **Corrupt** $(GWN, 1)$ query.

Correctness. If U^i and S^j are partnered and accepted, they will share a common session key.

Semanticsecurity. One of the main purposes of the authentication protocol is to protect the session key from being obtained by the adversary. In a protocol $\mathcal{P}$, $\mathcal{A}$ can ask a polynomial time of Execute, Send, Reveal queries. It can also ask for one time Test query. At the end of the game, the adversary tries to guess and output a bit b' for the value of b from **Test**-query. We definite that $\mathcal{A}$ wins the game if $b' = b$ and use $Succ$ to denote this event. The advantage of $\mathcal{A}$ in breaking semantic security of $\mathcal{P}$ is:

$$\mathrm{Adv}_{p}^{\mathrm{ake}}(\mathcal{A}) = 2\Pr[\mathrm{Succ}(\mathcal{A})] - 1 = 2\Pr[b' = b] - 1$$

For a secure smart card based password authentication protocol, online password guessing attacks should be the best strategy adopted by the adversary to break the protocol. Therefore, if the advantage of the $\mathcal{A}$ who can make at most q_{send} online attacks satisfied:

$$\mathrm{Adv}_{\mathcal{P},\mathcal{D}}^{\mathrm{ake}}(\mathcal{A}) \leq C' \cdot q_{\mathrm{send}}^{s'} + \epsilon(\ell)$$

Then we say that the protocol is semantic secure. The $\mathcal{D}$ is the password space whose frequency distribution follows a Zipf's law [47] and C', s' are the parameters of Zipf, $\epsilon(\cdot)$ is a negligible function, and ℓ is a system security parameter. Before the security analysis, we introduce the computational assumption required for the formal security proof:

EllipticCurveGapDiffie − Hellman(ECGDH) problem: A (t, ϵ)-ECGDH attacker in $\mathcal{G}$ is a PPT machine Δ running in time t such that

$$\mathrm{Adv}_{g,\mathcal{G}}^{\mathrm{ECGDH}}(\Delta) = \Pr_{x,y}[\Delta(xP, yP) = xyP]$$
$$\mathrm{Adv}_{g,\mathcal{G}}^{\mathrm{ECGDH}}(t) = \max_{\Delta}\left\{\mathrm{Adv}_{g,\mathcal{G}}^{\mathrm{ECGDH}}(\Delta)\right\}$$

the probability is taken over the random values x and y. The ECGDH-Assumption states that $\mathrm{Adv}_{g,\mathcal{G}}^{\mathrm{ECGDH}}(t) \leq \epsilon$, Where ϵ is a negligible value.

Theorem 1: Let $\mathcal{P}$ denote the protocol we proposed in Section VII, $|\mathcal{D}|$ is the space of the password. Let $\mathcal{A}$ denote the adversary who against the semantic security of the protocol within a time bound t, with q_{send} times **Send**-queries, q_{exe} times **Execute**-queries and q_h times **hash**-queries. Then the advantage of $\mathcal{A}$ to break the protocol $\mathcal{P}$ is

$$\mathrm{Adv}_{\mathcal{P}}^{\mathrm{ake}}(\mathcal{A}) \leq C' \cdot q_{\mathrm{send}}^{s'} + 6q_h\,\mathrm{Adv}_{\mathcal{P}}^{\mathrm{ECGDH}}(t') + \frac{q_h^2 + 6q_{\mathrm{send}}}{2^l} + \frac{(q_{\mathrm{send}} + q_{\mathrm{exe}})^2}{p}$$

where $t' \leqslant t + (q_{send} + q_{exe} + 1) \cdot T_m$, T_m is time for scaler multiplication operation in $\mathcal{G}$, C', s' are the parameters of the Zipf's law [47]. The specific proof are as follows.

Proof: We prove this theorem by a series of hybrid games, starting with the real attack Game G_0 and ending up with a game where the advantage of $\mathcal{A}$ is 0. In all games, the oracle handles queries as described in the protocol. We have defined the following events for each game $\mathbf{G_n}$, where $n = (0, 1, 2, \ldots, 8)$.

– $\mathbf{Succ_n}$: The adversary $\mathcal{A}$ can successfully guess the bit b in Test-query.
– $\mathbf{AskPara_n}$: The adversary $\mathcal{A}$ can correctly compute the parameter V_i by asking a hash query H(·) on $(ID_i||T_{reg}||X_{GWN})$ or $(HPW_i||ID_i)$.
– $\mathbf{AskAuth_n}$: The adversary $\mathcal{A}$ can correctly compute the parameter V_i and ask a hash H(·) on $(C_1||DID_i||X_2||V_i||SID_j)$ or $(r_2||X_2||X_3||V_i||SID_j||C_5)$.
– $\mathbf{AshH_n}$: On the basis of the above hash-queries, the adversary $\mathcal{A}$ successfully asks a hash query on $(X_1||X_3||X_4||r_2||SID_j)$ to compute the session key SK.

Game $\mathbf{G_0}$. This game is a real attack under the random oracle model, thus, there are

$$\mathrm{Adv}_{\mathcal{P}}^{ake}(A) = 2\Pr[Succ_0] - 1 \quad (1)$$

Game $\mathbf{G_1}$. In this game, we simulate hash-query $H(\cdot)$ and $H'(\cdot)$ that appear in Game G_6 later. The system maintains two hash lists $\Lambda_{\mathcal{H}}$ and $\Lambda_{\mathcal{A}}$. The list $\Lambda_{\mathcal{H}}$ is used to store the record of the hash queries. For example, for a hash query $H(x)$ with x as input, the oracle first checks whether the list $\Lambda_{\mathcal{H}}$ contains a record (x, y). If so, the answer is y. Otherwise, the oracle chooses a random number $y \in \{0, 1\}^l$ as output and stores the record (x, y) in the list $\Lambda_{\mathcal{H}}$. The list $\Lambda_{\mathcal{A}}$ is used to store the hash-queries record asked by adversary $\mathcal{A}$. For other queries,

we also simulate and keep the same as the real players would do. Therefore, this game is no different from a real attack. Then we have

$$|\Pr[\text{Succ}_1] - \Pr[\text{Succ}_0]| \leq \epsilon(l) \quad (2)$$

Game $\mathbf{G_2}$. In this game, all oracles remain the same as Game G_1. To facilitate subsequent analysis, we need to remove possible collisions.

The collisions include the following two types:

-The collisions on the output of hash queries.

-The collisions on the message $\{\{C_1, C_2, X_1, DID_i\}, \{C_3, C_4, X_1\}, \{C_5, C_6, X_3\}, \{C_5, C_7, X_3\}\}$.

According to the birthday paradox principle [56], we have

$$|\Pr[\text{Succ}_2] - \Pr[\text{Succ}_1]| \leq \frac{(q_{\text{send}} + q_{\text{exe}})^2}{2p} + \frac{q_{\text{h}}^2}{2^{l+1}} \quad (3)$$

Game $\mathbf{G_3}$. In this game, we consider the situation that if the adversary is lucky to guess the correct verifier directly $\{C_2, C_4, C_6, C_7\}$ in the protocol (without asking the hash query $H(\cdot)$), then we have

$$|\Pr[Succ_3] - \Pr[Succ_2]| \leq \frac{q_{send}}{2^l} \quad (4)$$

Game $\mathbf{G_4}$. In this game, we abort the game if the adversary is lucky to directly guess the V_i correctly (without asking the hash query $H(\cdot)$. Then we have

$$|\Pr[\text{Succ}_4] - \Pr[\text{Succ}_3]| \leq \frac{q_{\text{send}}}{2^l} \quad (5)$$

Game $\mathbf{G_5}$. In this game, we abort the game if the $\mathcal{A}$ can compute the correct V_i by asking the corresponding hash query. In the case where the $AskPara_5$ event does not occur, we cannot distinguish G_4 and G_5. Then we have

$$|\Pr[\text{Succ}_5] - \Pr[\text{Succ}_4]| \leq \Pr[AskPara_5] \quad (6)$$

Since the case of directly guessing V_i's value has been ruled out before, the $AskPara_5$ event can be divided into the following two cases:

-The adversary asks a $Corrupt(U, 1)$ query to compute the value of V_i.

-The adversary asks a $Corrupt(U, 2)$ query to compute the value of V_i.

We define these two events as $AskPara_5 withCorrupt_1$ and $AskPara_5 withCorrupt_2$, respectively. We first consider the probability of the event $AskPara_5 withCorrupt_1$. The adversary uses a $Corrupt(U, 1)$ query to obtain user's password PW_U. Since $\mathcal{A}$ does not know other parameters, the probability of this event is the same as direct guessing. Then we have

$$\Pr[AskPara_5 withCorrupt_1] \leq \frac{q_{\text{send}}}{2^l} \quad (7)$$

Subsequently, $\mathcal{A}$ can use $Corrupt(U, 2)$ query to obtain the parameters of the smart card, then she can compute the value of V_i by offline guessing user's password PW_U whose frequency distribution follows the Zipf's law [47]. Then we have

$$\Pr[AskPara_5 withCorrupt_2] \leq \frac{1}{2} c' \cdot q_{\text{send}}^{s'} \quad (8)$$

According to the above equations, we have

$$|\Pr[\text{Succ}_5] - \Pr[\text{Succ}_4]| \leq \frac{1}{2} c' \cdot q_{\text{send}}^{s'} + \frac{q_{\text{send}}}{2^l} \quad (9)$$

Game $\mathbf{G_6}$. In this game, we abort the game if the $\mathcal{A}$ can compute the correct authenticator C_2, C_7 by asking corresponding hash query and impersonating as a user or the sensor node. The G_5 and G_6 are indistinguishable unless event $AskAuth_6$ occurs, then we have

$$|\Pr[\text{Succ}_6] - \Pr[\text{Succ}_5]| \leq \Pr[AskAuth_6] \quad (10)$$

$$\Pr[AskPara_6] - \Pr[AskPara_5] \leq \Pr[AskAuth_6] \quad (11)$$

Game $\mathbf{G_7}$. In this game, we use private hash $H'(\cdot)$ to replace the hash $H(\cdot)$ in the protocol. The G_6 and G_7 are indistinguishable unless event $AskH_7$ occurs, then we have

$$|\Pr[\text{Succ}_7] - \Pr[\text{Succ}_6]| \leq \Pr[AskH_7] \quad (12)$$

$$|\Pr[AskAuth_7] - \Pr[AshAuth_6]| \leq \Pr[AskH_7] \quad (13)$$

Besides, since $\mathcal{A}$ cannot use the private hash $H'(\cdot)$, then

$$\Pr[\text{Succ}_7] = \frac{1}{2} \quad (14)$$

Game $\mathbf{G_8}$. In this game, we use the random self-reducibility of the Elliptic Curve Gap Diffie-Hellman problem to simulate the executions. Since the G_8 and G_7 are indistinguishable, then

$$\Pr[AskH_8] = \Pr[AskH_7] \quad (15)$$

The event $AskH_8$ occurs means that $\mathcal{A}$ had queried the random oracles $H(\cdot)$ on $H(X_1||X_3||ECGDH(X_1, X_3)||r_2||\ SID_j)$. Then we have

$$|\Pr[AskH_8]| \leq q_h \cdot \text{Adv}_{\mathcal{P}}^{\text{ECGDH}}(t') \quad (16)$$

where $t' \leq t + (q_{send} + q_{exe} + 1) \cdot T_m$. First, from the (1)–(5), we have

$$|\Pr[\text{Succ}_4] - \Pr[\text{Succ}_0]| \leq \frac{(q_{send} + q_{exe})^2}{2p} + \frac{q_h^2}{2^{l+1}} + \frac{2q_{send}}{2^l}$$

Then, from (6)–(12), we get

$$|\Pr[\text{Succ}_7] - \Pr[\text{Succ}_4]| \leq$$
$$\Pr[AskPara_5] + \Pr[AskAuth_6] + \Pr[AskH_7]$$

Lastly, from (13)–(16), we can get

$$\text{Adv}_{\mathcal{P}}^{\text{ake}}(\mathcal{A}) = 2\Pr[\text{Succ}_7] - 1 + 2(\Pr[\text{Succ}_0] - \Pr[\text{Succ}_7])$$
$$\leq C' \cdot q_{\text{send}}^{s'} + 6q_h \text{Adv}_{\mathcal{P}}^{\text{ECGDH}}(t') + \frac{q_h^2 + 6q_{\text{send}}}{2^l}$$
$$+ \frac{(q_{\text{send}} + q_{\text{exe}})^2}{p}$$

We prove that the advantage of the adversary is gradually decreasing to zero via a series of hybrid games. Thus, the adversary cannot break the semantic security of the protocol. We formally prove the security of our protocol.

IX. Security Analysis of Our Scheme

Due to the limitations of formal methods [36], [57], many realistic attacks are difficult to capture. In this section, we use heuristic analysis to analyze our protocol.

A. Offline Password Guessing Attack

Generally, the adversary can obtain all the information in the public channel. And she can also obtain the parameters stored in the smart card. We assume that the adversary uses the above information to offline guess the user's ID_i^* and PW_i^*. To check the correctness of (ID_i^*, PW_i^*), the adversary can compute $HPW_i^* = H(PW_i||b)$, $A_i^* = H(ID_i^*||HPW_i^*)$ mod n_0 and check whether $A_i^* = A_i$. However, even if it is equal, the adversary also cannot be sure that ID_i^* and PW_i^* are the same as the user's ID_i and PW_i. To resist offline password guessing attacks, we use fuzzy-verifier technology. Specifically, $\mathcal{A}$ can find $|D_{ID} \cdot D_{pw}|/n_0$ pairs of (ID_i, PW_i) to make $A_i^* = A_i$. Therefore, the adversary can only launch an online attack to test whether they are correct. To prevent the adversary from unlimited online testing, we use "honeywords" to detect this attack. Each time the adversary fails, the $HoneyList$'s value will add one until the value of $HoneyList$ exceeds the predetermined threshold (such as 10). Then the GWN suspends the use of the user's smart card. Thus, our proposed scheme can resist this attack. Besides, the adversary can also execute offline password guessing attacks with another method. The adversary can compute $V_i^* = B_i \oplus H(HPW_i^*||ID_i^*)$ and verify V_i^* by computing the parameter C_2 or C_7. However, the adversary cannot obtain the GWN's long-term secret key X_{GWN}, she cannot compute the parameter X_2. Our scheme can resist offline password guessing attacks.

B. Smart Card Loss Attack

Generally, if the adversary impersonates the user using the smart card, the password is required. However, the above analysis has demonstrated that our protocol effectively resists offline password guessing attacks by employing "fuzzy-verifier" and "honeywords". Even if the adversary possesses the smart card along with its stored secret parameters, she cannot retrieve the password or any other password-related information. Furthermore, the adversary cannot use the parameters stored in the smart card to compute the session key or any other secret value. Our protocol can resist smart card loss attacks.

C. User Anonymity

User anonymity [20] usually has two requirements aspects: 1) Anonymity: The user identity cannot be known by communication entities other than GWN; 2) Indistinguishability: The adversary cannot distinguish whether the communication data between entities is sent by the same user. In our scheme, we do not directly transmit the user identity in the public channel. Thus, the adversary cannot obtain the user identity. We set user's temporary identity $DID_i = ID_i \oplus H(X_2)$. Since the user has to choose a random number for every authentication, the DID_i is different every time. Meanwhile, the adversary cannot obtain the GWN's long-term secret key X_{GWN} to compute X_2. The adversary can neither obtain the user identity nor distinguish whether messages comes from the same user. Our scheme can achieve user anonymity.

D. Gateway Spoofing Attack

In our scheme, if the adversary wants to impersonate the gateway, she needs to compute $V_i = H(ID_i\|T_{reg}\|X_{GWN})$. Since the adversary cannot know the long-term secret key X_{GWN} of GWN, our scheme can resist this attack.

E. Replay Attack

To resist replay attacks, we use random numbers to ensure the freshness of information in our protocol. When the participants of this scheme receive a message, they will check whether the random number is valid. Moreover, each time the protocol is executed, the number is randomly generated and not the same, so the attacker cannot launch replay attacks.

F. User Impersonation Attack

The first method for adversary is obtaining user's ID_i and PW_i. However, the adversary cannot use offline password guessing to get the correct ID_i and PW_i. The other method is that the adversary gets the message $C_2 = H(C_1||DID_i||X_2||V_i||SID_j)$ and impersonates the user. Unfortunately, the value of C_2 is related to V_i, which can be computed by $V_i = B_i \oplus H(HPW_i||ID_i)$ or $V_i = H(ID_i||T_{reg}||X_{GWN})$. Since the adversary cannot get V_i, she cannot launch the user impersonation attack in our scheme.

G. Forward Secrecy

When considering the forward secrecy of the scheme, we assume that the adversary can obtain the long-term secret key X_{GWN}. If the adversary wants to obtain the session key, she needs to get X_4 from X_1, X_3. However, it is impossible for the adversary to crack the elliptic curve encryption algorithm, which is a hard problem in polynomial time. Thus, our scheme can achieve forward secrecy.

H. Sensor Node Capture Attack

The sensor nodes are deployed in an unsupervised environment for a long time, and their security parameters can be easily acquired by the adversary [7]. Therefore, to prevent the node capture attack of the adversary, in our protocol, GWN only sends the random number selected by the user to the sensor node S. Even if the adversary captures the sensor node and the random numbers of the user, it is also impossible to get the value of V_i or the session key. Therefore, our protocol can resist the sensor node capture attack.

TABLE IV
PERFORMANCE COMPARISON AMONG RELEVANT SCHEMES

Related Protocols	Computational cost (ms)*			The evaluation criteria in Table 2 [6]											
	User	Gateway	Sensor node	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12
Our scheme	$2T_P$+$9T_H$≈1.022	T_P+$6T_H$≈0.512	$2T_P$+$4T_H$≈1.019	√	√	√	√	√	√	√	√	√	√	√	√
Xie et al. (2024) [40]	T_S+$8T_H$≈0.006	T_S+$6T_H$≈0.005	T_S+$2T_H$≈0.002	√	×	√	×	×	×	√	×	×	√	×	×
Bai et al. (2023) [33]	T_E+$6T_H$≈1.173	T_E+$6T_H$≈1.173	$3T_H$≈0.002	√	√	√	×	×	×	√	×	√	√	√	×
Xu et al.(2023) [35]	T_B+$7T_H$+$5T_P$≈1002.544	T_P+$4T_H$≈0.511	$6T_P$+$6T_H$≈3.052	√	√	√	×	×	√	√	×	√	√	×	√
Jabbari-Mohasef (2022) [26]	T_B + $10T_S$≈1000.005	$6T_S$ + T_H≈0.004	$7T_S$ + TH≈0.005	√	√	√	×	×	×	√	×	√	×	×	×
Raque et al. (2022) [38]	T_B + T_S + $10T_H$≈1000.007	$4T_S$ + $8T_H$≈0.008	$2T_S$ + $4TH$≈0.005	√	√	√	×	×	√	√	×	√	√	×	×
Zhang et al. (2022) [34]	T_B + $2T_S$ + $5T_H$≈1000.005	$6T_S$ + $4T_H$≈0.004	$2T_S$ + $2T_H$≈0.003	√	√	√	×	×	×	√	√	√	√	√	×
Li et al. (2022) [58]	T_B + $11T_H$≈1000.008	$13T_H$≈0.009	$6T_H$≈0.004	√	√	√	×	×	√	√	×	√	√	×	×
Chaudhry et al. (2021) [25]	T_B + $16T_H$≈1000.011	$14T_H$≈0.010	$7T_H$≈0.004	√	√	√	×	×	×	√	×	√	√	√	×
Wazid et al. (2020) [27]	T_B + $13T_H$≈1000.009	$8T_H$≈0.006	$8T_H$≈0.006	√	√	√	×	×	√	√	×	√	√	√	×
Li et al. (2020) [28]	$10T_H$≈0.007	$9T_H$≈0.006	$7T_H$≈0.005	√	√	√	×	×	√	√	×	√	×	×	×
Wang et al. (2020) [52]	T_{be}+T_{se}+$9T_H$≈9.514	T_{be}+T_S+$6T_H$≈9.357	T_{se}+T_S+$4T_H$≈0.157	√	√	√	√	×	√	√	√	√	√	√	×
Xu et al. (2019) [29]	$5T_H$+T_B≈1000.003	$12T_H$≈0.008	$4T_H$ ≈ 0.002	√	×	×	×	×	×	√	√	×	√	√	×
Guo et al. (2019) [30]	$6T_H$+T_B≈1000.004	$16T_H$≈0.011	$6T_H$≈0.004	√	√	√	×	×	×	√	×	×	√	√	×
Srinivas et al.(2018) [39]	T_B+$2T_C$+$15T_H$≈1002.624	$10T_H$≈0.007	$2T_C$+$6T_H$≈2.618	√	√	√	×	×	√	√	×	√	√	×	√
Wazid et al.(2018) [31]	T_B+$2T_S$+$13T_H$≈1000.010	$4T_S$+$5T_H$≈0.006	$2T_S$+$4T_H$≈0.004	√	√	√	×	×	√	√	×	√	√	×	×
Li et al.(2018) [59]	T_B+$2T_P$+$8T_H$≈1001.022	T_P+$9T_H$≈0.514	$4T_H$≈0.003	√	√	√	×	×	×	√	×	√	√	×	×
Li et al.(2018) [60]	T_B+$3T_P$+$7T_H$≈1001.529	T_P+$7T_H$≈0.513	$2T_P$+$4T_H$≈1.019	√	√	√	×	√	×	√	√	√	√	√	√
Wang et al.(2017) [61]	T_B+$3T_P$+$10T_H$ ≈ 1001.531	T_P+$11T_H$≈0.516	$2T_P$+$4T_H$≈1.019	√	√	√	√	√	×	√	√	√	√	√	√

* $T_S, T_H, T_E, T_P, T_C, T_B, T_{be}, T_{se}$ denote the time of symmetric encryption/decryption, hash computation, modular exponentiation operation, scalar multiplication on elliptic curve, Chebyshev polynomial operation, fuzzy extracting biometric info, big-exponent modular exponentiation of RSA, small-exponent modular exponentiation of RSA. According to [36], T_E ≈1.169 ms (when we set the length of modular $|n| = 512$), T_P≈0.508 ms, T_H≈0.693μs, T_S≈0.541μs. The T_{bc} ≈1.169*8=9.352 ms and $T_{se} \approx \frac{17}{1024} * T_{bc}$ =0.156 ms. And the T_B≈1000ms. [52].

I. Privileged Insider Attack

For our scheme, the gateway does not store any parameters related to user passwords in memory. Even if insiders obtain the parameters of the smart card, the parameters in the public channel, and the information stored in the gateway, they cannot calculate the user's password. Since we use "fuzzy-verifier" [16] to resist offline password guessing attacks, adversaries cannot find a verifier to check the password. Meanwhile, due to the use of public-key technology, insiders also cannot obtain the session key. Therefore, our scheme can resist insider attacks.

J. Parallel Attack

Parallel attack [62] means that the adversary uses the information obtained in one protocol progress to reply to another protocol progress. Then the adversary can obtain the session key by cheating the gateway or others. In our protocol, we use different random numbers each time the protocol is executed, so the information of each protocol is different. Thus, in our scheme, the adversary cannot use the information in the public channel to launch parallel attacks.

K. De-Synchronization Attack

Compared with other protocols, our protocol has no parameters that require participants to update simultaneously in the protocol process. If the user desires to update the password, she should authenticate her identity first. Meanwhile, we do not use clock synchronization in our scheme. Our scheme can resist de-synchronization attacks.

L. Mutual Authentication

Our proposed protocol provides (1) mutual authentication between the gateway and the user. The user and gateway authenticate each other by computing C_2, C_7 to check the value of V_i. (2) mutual authentication between the sensor node and gateway. The gateway and the sensor node authenticate each other by computing C_4, C_6 to check the sensor node's secret key $P_j = H(SID_j||X_{GWN})$. Obviously, our protocol can achieve mutual authentication with the help of the gateway.

X. PERFORMANCE ANALYSIS

Wang et al.'s evaluation framework [6] is concrete, systematic and comprehensive, and has been widely accepted (see [7], [45], [63]). The specific details of the evaluation criteria can be seen in Section II-B. In this section, we will use it to compare and analyze our scheme with 18 state-of-the-art related protocols in terms of security and performance. From the comparison results, only our protocol fully satisfies all 12 criteria. For the performance analysis, the primary focus is on the computational cost of each participant in the protocol. The comparison results are presented in Table IV.

It is evident from the analysis that protocols employing symmetric encryption or hash functions [27], [28], [29], [30], [31], [33], [34], [38], [40], [58] generally exhibit better performance compared to those based on public-key cryptography [35], [39], [60], [61]. However, without exception, these symmetric encryption-based protocols fail to achieve forward secrecy and multi-factor security. This observation aligns with Ma et al. [24], which states that public-key cryptography is an essential component for achieving forward secrecy in MFA protocols.

It is worth emphasizing that in security-critical WSN environments, the security of authentication protocols should be considered at least as important as, if not more important than, their efficiency. Consequently, critical security properties must not be compromised for the sake of improving efficiency. Among the protocols analyzed, only the scheme proposed by Wang et al. [61] achieves a security level comparable to ours. The security of the remaining public-key-based protocols is weaker than that of our protocol. Furthermore, our two-factor authentication protocol meets the security goals of the three-factor authentication protocol proposed by Wang et al. [61]. Since our protocol does not employ the fuzzy extractor, it offers significantly higher

user-side efficiency compared to [61] and most other three-factor authentication schemes.

Although our scheme is less efficient in sensor nodes than most existing protocols, it is the most efficient scheme among those capable of resisting node capture attacks and achieving forward secrecy. The comprehensive comparison result demonstrates that our scheme outperforms its foremost counterparts and performs well in WSNs.

XI. Conclusion

In this work, we revisited two leading multi-factor authentication protocols [25], [26] for WSNs, and pointed out that neither of them can achieve the claimed security. Besides reporting security defects, we revealed the fundamental causes of their protocol problems and proposed corresponding effective countermeasures to help protocol designers avoid making the same mistakes. By incorporating countermeasures (e.g., public-key cryptography and fuzzy-verifier) into multi-factor authentication design, we proposed a new two-factor authentication protocol for WSNs and formally proved its security. Meanwhile, we compared our protocol with 18 foremost counterparts, and the results show the superiority of our protocol. We hope that this work will facilitate the protocol designers to have a better understanding of the essential problems and difficulties in protocol design. Our protocol can not only be used for WSNs, but would also provide inspiration and solutions for multi-factor authentication protocols in other environments.

References

[1] F. Wang, J. Cui, Q. Zhang, D. He, and H. Zhong, "Blockchain-based secure cross-domain data sharing for edge-assisted industrial Internet of Things," *IEEE Trans. Inf. Forensics Secur.*, vol. 19, pp. 3892–3905, 2024.

[2] B. Li, Y. Chen, L. Zhang, L. Wang, and Y. Cheng, "HomeSentinel: Intelligent anti-fingerprinting for IoT traffic in smart homes," *IEEE Trans. Inf. Forensics Secur.*, vol. 19, pp. 4780–4793, 2024.

[3] Y. Wang, Z. Wang, J. A. Zhang, H. Zhang, and M. Xu, "Vital sign monitoring in dynamic environment via mmWave radar and camera fusion," *IEEE Trans. Mobile Comput.*, vol. 23, no. 5, pp. 4163–4180, May 2024.

[4] M. Wazid, A. K. Das, S. Shetty, and J. J. P. C. Rodrigues, "On the design of secure communication framework for blockchain-based internet of intelligent battlefield things environment," in *Proc. IEEE Conf. Comput. Commun. Workshops*, 2020, pp. 888–893.

[5] M. Mamdouh, A. I. Awad, A. A. M. Khalaf, and H. F. A. Hamed, "Authentication and identity management of IoHT devices: Achievements, challenges, and future directions," *Comput. Secur.*, vol. 111, 2021, Art. no. 102491.

[6] D. Wang, W. Li, and P. Wang, "Measuring two-factor authentication schemes for real-time data access in industrial wireless sensor networks," *IEEE Trans. Ind. Informat.*, vol. 14, no. 9, pp. 4081–4092, Sep. 2018.

[7] C. Wang, D. Wang, Y. Tu, G. Xu, and H. Wang, "Understanding node capture attacks in user authentication schemes for wireless sensor networks," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 1, pp. 507–523, Jan./Feb. 2022.

[8] J. Bonneau, C. Herley, P. C. van Oorschot, and F. Stajano, "The quest to replace passwords: A framework for comparative evaluation of web authentication schemes," in *Proc. IEEE Symp. Secur. Privacy*, 2012, pp. 553–567.

[9] L. H. Newman, "The password isn't dead yet, you need a hardware key," Dec. 2022. [Online]. Available: https://www.wired.com/story/hardware-security-key-passwords-passkeys/

[10] S. Jarecki, H. Krawczyk, and J. Xu, "OPAQUE: An asymmetric PAKE protocol secure against pre-computation attacks," in *Proc. Annu. Int. Conf. Theory Appl. Cryptographic Techn.*, 2018, pp. 456–486.

[11] B. F. D. Santos, Y. Gu, S. Jarecki, and H. Krawczyk, "Asymmetric PAKE with low computation and communication," in *Proc. Annu. Int. Conf. Theory Appl. Cryptographic Techn.*, 2022, pp. 127–156.

[12] Y. Gu, S. Jarecki, P. Kedzior, P. Nazarian, and J. Xu, "Threshold PAKE with security against compromise of all servers," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, 2024, pp. 66–100.

[13] Yahoo's 2013 Email hack actually compromised three billion accounts, Oct. 2013. [Online]. Available: https://www.wired.com/story/yahoo-breach-three-billion-accounts/

[14] F. Neil, List of data breaches and cyber attacks in 2023, 2023. [Online]. Available: https://www.itgovernance.co.uk/blog/list-of-data-breaches-and-cyber-attacks-in-2023

[15] D. Wang, Y. Zou, Z. Zhang, and K. Xiu, "Password guessing using random forest," in *Proc. USENIX Conf. Secur. Symp.*, 2023, pp. 965–982.

[16] D. Wang and P. Wang, "Two birds with one stone: Two-factor authentication with security beyond conventional bound," *IEEE Trans. Dependable Secure Comput.*, vol. 15, no. 4, pp. 708–722, Jul./Aug. 2018.

[17] M. L. Das, "Two-factor user authentication in wireless sensor networks," *IEEE Trans. Wireless Commun.*, vol. 8, no. 3, pp. 1086–1090, Mar. 2009.

[18] R. Fan, D. He, X. Pan, and L. Ping, "An efficient and DoS-resistant user authentication scheme for two-tiered wireless sensor networks," *J. Zhejinag Univ. Sci. C*, vol. 12, no. 7, pp. 550–560, 2011.

[19] D. He, Y. Gao, S. Chan, C. Chen, and J. Bu, "An enhanced two-factor user authentication scheme in wireless sensor networks," *Ad Hoc Sens. Wireless Netw.*, vol. 10, no. 4, pp. 361–371, 2010.

[20] D. Wang and P. Wang, "On the anonymity of two-factor authentication schemes for wireless sensor networks: Attacks, principle and solutions," *Comput. Netw.*, vol. 73, no. C, pp. 41–57, 2014.

[21] A. Das, S. Sharma, P. Chatterjee, and J. Sing, "A dynamic password-based user authentication scheme for hierarchical wireless sensor networks," *J. Netw. Comput. Appl.*, vol. 35, no. 5, pp. 1646–1656, 2012.

[22] C. T. Li, C. Y. Weng, and C. C. Lee, "An advanced temporal credential-based security scheme with mutual authentication and key agreement for wireless sensor networks," *Sensors*, vol. 13, no. 8, pp. 9589–9603, 2013.

[23] K. Xue, C. Ma, P. Hong, and R. Ding, "A temporal-credential-based mutual authentication and key agreement scheme for wireless sensor networks," *J. Netw. Comput. Appl.*, vol. 36, no. 1, pp. 316–323, 2013.

[24] C. Ma, D. Wang, and S. Zhao, "Security flaws in two improved remote user authentication schemes using smart cards," *Int. J. Commun. Syst.*, vol. 27, no. 10, pp. 2215–2227, 2012.

[25] S. A. Chaudhry, A. Irshad, K. Yahya, N. Kumar, M. Alazab, and Y. B. Zikria, "Rotating behind privacy: An improved lightweight authentication scheme for cloud-based IoT environment," *ACM Trans. Internet Technol.*, vol. 21, no. 3, pp. 1–19, 2021.

[26] A. Jabbari and J. B. Mohasef, "A secure and LoRaWAN compatible user authentication protocol for critical applications in the IoT environment," *IEEE Trans. Ind. Informat.*, vol. 18, no. 1, pp. 56–65, Jan. 2022.

[27] M. Wazid, A. K. Das, V. B. K, and A. V. Vasilakos, "LAM-CIoT: Lightweight authentication mechanism in cloud-based IoT environment," *J Netw. Comput. Appl.*, vol. 150, 2020, Art. no. 102496.

[28] J. Li, Z. Su, D. Guo, K. K. R. Choo, and Y. Ji, "PSL-MAAKA: Provably-secure and lightweight mutual authentication and key agreement protocol for fully public channels in internet of medical things," *IEEE Internet Things J.*, vol. 8, no. 17, pp. 13183–13195, Sep. 2021.

[29] L. Xu and F. Wu, "A lightweight authentication scheme for multi-gateway wireless sensor networks under IoT conception," *Arab. J. Sci. Eng.*, vol. 44, no. 4, pp. 3977–3993, 2019.

[30] H. Guo, Y. Gao, T. Xu, X. Zhang, and J. Ye, "A secure and efficient three-factor multi-gateway authentication protocol for wireless sensor networks," *Ad Hoc Netw.*, vol. 95, 2019, Art. no. 101965.

[31] M. Wazid, A. K. Das, V. Odelu, N. Kumar, M. Conti, and M. Jo, "Design of secure user authenticated key management protocol for generic IoT networks," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 269–282, Feb. 2018.

[32] L. Zhu and D. Wang, "Robust multi-factor authentication for WSNs with dynamic password recovery," *IEEE Trans. Inf. Forensics Secur.*, vol. 19, pp. 8398–8413, 2024.

[33] L. Bai, C. Hsu, L. Harn, J. Cui, and Z. Zhao, "A practical lightweight anonymous authentication and key establishment scheme for resource-asymmetric smart environments," *IEEE Trans. Dependable Secure Comput.*, vol. 20, no. 4, pp. 3535–3545, Jul./Aug. 2023.

[34] L. Zhang, Y. Zhu, W. Ren, Y. Zhang, and K. R. Choo, "Privacy-preserving fast three-factor authentication and key agreement for IoT-based e-health systems," *IEEE Trans. Serv. Comput.*, vol. 16, no. 2, pp. 1324–1333, Mar./Apr. 2023.

[35] H. Xu, C. Hsu, L. Harn, J. Cui, Z. Zhao, and Z. Zhang, "Three-factor anonymous authentication and key agreement based on fuzzy biological extraction for industrial Internet of Things," *IEEE Trans. Serv. Comput.*, vol. 16, no. 4, pp. 3000–3013, Jul./Aug. 2023.

[36] D. Wang, D. He, P. Wang, and C. Chu, "Anonymous two-factor authentication in distributed systems: Certain goals are beyond attainment," *IEEE Trans. Dependable Secure Comput.*, vol. 12, no. 4, pp. 428–442, Jul./Aug. 2015.
[37] X. Huang, Y. Xiang, A. Chonka, J. Zhou, and R. H. Deng, "A generic framework for three-factor authentication: Preserving security and privacy in distributed systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 22, no. 8, pp. 1390–1397, Aug. 2011.
[38] F. Rafique, M. S. Obaidat, K. Mahmood, M. F. Ayub, J. Ferzund, and S. A. Chaudhry, "An efficient and provably secure certificateless protocol for industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 18, no. 11, pp. 8039–8046, Nov. 2022.
[39] J. Srinivas, A. K. Das, M. Wazid, and N. Kumar, "Anonymous lightweight chaotic map-based authenticated key agreement protocol for industrial Internet of Things," *IEEE Trans. Dependable Secure Comput.*, vol. 17, no. 6, pp. 1133–1146, Nov./Dec. 2020.
[40] D. Xie, J. Yang, B. Wu, W. Bian, F. Chen, and T. Wang, "An effectively applicable to resource constrained devices and semi-trusted servers authenticated key agreement scheme," *IEEE Trans. Inf. Forensics Secur.*, vol. 19, pp. 3451–3464, 2024.
[41] Q. Wang and D. Wang, "Understanding failures in security proofs of multi-factor authentication for mobile devices," *IEEE Trans. Inf. Forensics Secur.*, vol. 18, pp. 597–612, 2023.
[42] K. Hussain, N. Z. Jhanjhi, H. Mati-ur-Rahman, J. Hussain, and M. H. Islam, "Using a systematic framework to critically analyze proposed smart card based two factor authentication schemes," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 33, no. 4, pp. 417–425, 2021.
[43] H. Xiong, Z. Kang, J. Chen, J. Tao, C. Yuan, and S. Kumari, "A novel multiserver authentication scheme using proxy resignature with scalability and strong user anonymity," *IEEE Syst. J.*, vol. 15, no. 2, pp. 2156–2167, Jun. 2021.
[44] V. Sureshkumar, S. Anandhi, R. Amin, N. Selvarajan, and R. Madhumathi, "Design of robust mutual authentication and key establishment security protocol for cloud-enabled smart grid communication," *IEEE Syst. J.*, vol. 15, no. 3, pp. 3565–3572, Sep. 2021.
[45] P. Gope, A. K. Das, N. Kumar, and Y. Cheng, "Lightweight and physically secure anonymous mutual authentication protocol for real-time data access in industrial wireless sensor networks," *IEEE Trans. Ind. Informat.*, vol. 15, no. 9, pp. 4957–4968, Sep. 2019.
[46] E. Erdem and M. T. Sandikkaya, "OTPaaS-one time password as a service," *IEEE Trans. Inf. Forensics Secur.*, vol. 14, no. 3, pp. 743–756, Mar. 2019.
[47] D. Wang, H. Cheng, P. Wang, X. Huang, and G. Jian, "Zipf's law in passwords," *IEEE Trans. Inf. Forensics Secur.*, vol. 12, no. 11, pp. 2776–2791, Nov. 2017.
[48] D. Dolev and A. C. Yao, "On the security of public key protocols," *IEEE Trans. Inf. Theory*, vol. 29, no. 2, pp. 198–207, Mar. 1983.
[49] X. Huang, X. Chen, J. Li, Y. Xiang, and L. Xu, "Further observations on smart-card-based password-authenticated key agreement in distributed systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 7, pp. 1767–1775, Jul. 2014.
[50] M. Carbone et al., "Deep learning to evaluate secure RSA implementations," *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, vol. 2019, no. 2, pp. 132–161, 2019.
[51] S. Halevi and H. Krawczyk, "Public-key cryptography and password protocols," *ACM Trans. Inf. Syst. Secur.*, vol. 2, no. 3, pp. 230–268, 1999.
[52] D. Wang, P. Wang, and C. Wang, "Efficient multi-factor user authentication protocol with forward secrecy for real-time data access in WSNs," *ACM Trans. Cyber Phys. Syst.*, vol. 4, no. 3, pp. 30:1–30:26, 2020.
[53] H. Luo, G. Wen, and J. Su, "Lightweight three factor scheme for real-time data access in wireless sensor networks," *Wireless Netw.*, vol. 26, no. 2, pp. 955–970, 2020.
[54] P. Gope and T. Hwang, "A realistic lightweight anonymous authentication protocol for securing real-time application data access in wireless sensor networks," *IEEE Trans. Ind. Electron.*, vol. 63, no. 11, pp. 7124–7132, Nov. 2016.
[55] D. Wang, H. Cheng, P. Wang, J. Yan, and X. Huang, "A security analysis of honeywords," in *Proc. Annu. Netw. Distrib. Syst. Secur. Symp.*, 2018, pp. 1–16.
[56] K. Suzuki, D. Tonien, K. Kurosawa, and K. Toyota, "Birthday paradox for multi-collisions," in *Proc. Int. Conf. Inf. Secur. Cryptol.*, 2006, pp. 29–40.
[57] A. Menezes, "Another look at provable security," in *Proc. Annu. Int. Conf. Theory Appl. Cryptographic Techn.*, Springer, 2012, Art. no. 8.
[58] Y. Li and Y. Tian, "A lightweight and secure three-factor authentication protocol with adaptive privacy-preserving property for wireless sensor networks," *IEEE Syst. J.*, vol. 16, no. 4, pp. 6197–6208, Dec. 2022.
[59] X. Li, J. Niu, S. Kumari, F. Wu, A. K. Sangaiah, and K. K. R. Choo, "A three-factor anonymous authentication scheme for wireless sensor networks in Internet of Things environments," *J. Netw. Comput. Appl.*, vol. 103, pp. 194–204, 2018.
[60] X. Li, J. Niu, M. Z. A. Bhuiyan, F. Wu, M. Karuppiah, and S. Kumari, "A robust ECC based provable secure authentication protocol with privacy protection for industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 14, no. 8, pp. 3599–3609, Aug. 2018.
[61] C. Wang, G. Xu, and J. Sun, "An enhanced three-factor user authentication scheme using elliptic curve cryptosystem for wireless sensor networks," *Sensors*, vol. 17, no. 12, pp. 2946:1–2946:20, 2017.
[62] F. Hao, R. Metere, S. F. Shahandashti, and C. Dong, "Analysing and patching SPEKE in ISO/IEC," *IEEE Trans. Inf. Forensics Secur.*, vol. 13, no. 11, pp. 2844–2855, Nov. 2018.
[63] S. Qiu, D. Wang, G. Xu, and S. Kumari, "Practical and provably secure three-factor authentication protocol based on extended chaotic-maps for mobile lightweight devices," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 2, pp. 1338–1351, Mar./Apr. 2022.

Meijia Xu received the MS degree from the College of Cryptology and Cyber Science, Nankai University, Tianjin, P. R. China, in June 2023. She is currently working toward the PhD degree from the College of Cryptology and Cyber Science, Nankai University, Tianjin, P. R. China. She has published papers with venues like *Journal of Systems Architecture* and *Wireless Communications and Mobile Computing*. Her research interests include applied cryptography and password-based authentication.

Ding Wang received the PhD degree in information security from Peking University, in 2017, and was supported by the "Boya Postdoctoral Fellowship" in Peking University from 2017 to 2019. Currently, he is a full professor with Nankai University. As the first author (or corresponding author), he has published more than 100 papers with venues like IEEE S&P, ACM CCS, NDSS, Usenix Security, *IEEE Transactions on Dependable and Secure Computing* and *IEEE Transactions on Information Forensics and Security*. His research has been reported by more than 200 media like *The Wall Street Journal*, *Daily Mail*, *Forbes*, *IEEE Spectrum*, and *Communications of the ACM*, appeared in the Elsevier 2017 "Article Selection Celebrating Computer Science Research in China", and resulted in the revision of the authentication guideline NIST SP800-63-2. He has been involved in the community as a PC chair/TPC member for more than 60 international conferences such as NDSS 2023–2025, ACM CCS 2022, PETS 2022–2024, ACSAC 2020–2024, RAID 2024/2023, IEEE EuroS&P 2025, FC 2025, ACM AsiaCCS 2025/2022/2021, ICICS 2018–2025, SPNCE 2020–2022. He has received the "ACM China Outstanding Doctoral Dissertation Award", the Best Paper Award with INSCRYPT 2018, the First Prize of Cryptologic Innovation Award of the China Association for Cryptologic Research, and the First Prize of Natural Science Award of Ministry of Education. His main research interests focus on passwords, authentication, and provable security.