
Benfilali Mostefa, Gafour Abdelkader, and Belghachi Mohamed

Abstract—Recently, the numerous academic papers have been published on Authentication and Key Agreement (AKA) schemes for securing wireless sensor networks (WSN) in the context of the Internet of Things (IoT). The goal of these schemes is to protect external users’ access to data collected by WSNs. Due to limited resources and the wireless communication medium, the sensor nodes are vulnerable to multiple attacks performed by malicious user’s. Unfortunately, most of the proposed schemes are insecure and require higher storage, communication and computing costs. This paper presents a User Authentication protocol to secure WSNs Access in the IoT context (UAWSNA-IoT). The BAN-Logic and Real-Or-Random (ROR) models are used to demonstrate the reliability of UAWSNA-IoT in meeting all requirements for mutual authentication and session key security, respectively. In addition, UAWSNA-IoT offers high security with low computational, storage and communication overhead, making it an ideal choice for resource-constrained IoT devices such as WSNs.

Index Terms—Internet of Things, Wireless Sensor Networks, Mutual Authentication, Authentication and Key Agreement, User Authentication, Session Key, BAN-Logic, ROR model.

I. INTRODUCTION

THE Internet of Things (IoT) is a global information society infrastructure that enables the delivery of advanced services by connecting physical and virtual objects. These heterogeneous objects are interconnected, locatable, addressable, and readable in the Internet world [1]. IoT covers almost all areas of Information Technology (IT) such as smart cities, machine-to-machine systems, connected vehicles, and Wireless Sensor Networks (WSN), etc. WSN’s are a centrepiece of the success of the IoT, because they use small intelligent objects that are generally limited in terms of computational, storage, and energy capabilities. WSN consists of hundreds or thousands of sensor nodes deployed randomly or manually to monitor the hostile areas [4]. The primary goal of WSN is to detect and gather data on physical phenomena such as pressure, temperature, humidity, and blood pressure, and other [37]. Ultimately, this gathered data is transmitted to end users through wireless communication, facilitated by the GateWay Node (GWN) overseeing and coordinating the process. Furthermore, the Gateway Node (GWN) serves as a link between the WSN and the outside world (Internet), as all incoming and outgoing network data must pass through it [2]. In contrast to sensor nodes, the GWN has greater capabilities in terms of computing power, energy reserves and memory size. In addition, the wireless medium used in the various communications among network entities provides a suitable environment for attackers to carry out various attacks [6]. As a result, attackers can intercept, insert, delete, modify and redirect messages exchanged between legitimate parties. In such a scenario, message integrity, confidentiality and authentication are crucial to ensure the security and reliability of information exchanged between the entities [7]. Therefore, we require an efficient authentication protocol that is better suited to the resource-constrained environment of WSNs to secure the network and prevent attacks. In the literature, there are several user authentication protocols available for accessing wireless sensor networks [13], [24]. These protocols aim to ensure secure user access to data collected by sensor nodes. One commonly used protocol is mutual authentication, which incorporates the Authentication and Key Agreement (AKA) technique. In this protocol, a trusted third party called the gateway validates user’s identities and provides them with authorization to access data collected by the sensor nodes. The authentication techniques can be divided into three categories: single-factor, two-factor, and three-factor [5], [18]. The AKA technology enables the creation of session keys shared between users and sensor nodes, which secure future communications between them. This ensures that only legitimate users with session keys are authorized to access to WSN.

Our paper’s contribution addresses the following main points in light of the aforementioned challenges:

• We propose a novel lightweight and efficient authentication scheme named “UAWSNA-IoT” to secure the wireless sensor networks access from unauthorized users by using only a secure one-way hash function and bitwise XOR operations.

• Due to lack of sensor node resources such as memory storage space, we aim to reduce the numbers of authen-
tication parameters stored in the sensor node’s memory.

- Through formal analysis employing BAN-Logic and Real-Or-Random models, we demonstrated that UAWSNA-IoT guarantees both mutual authentication and session key security respectively. Additionally, informal assessment verifies its ability to withstand various known attacks.

- UAWSNA-IoT is capable of integrating the additional sensor nodes as needed, ensuring scalability to meet growing service demands.

- UAWSNA-IoT achieves higher security with acceptable computational, storage space and communication cost compared to related schemes [27], [14], [20], [25], [6], [35], and [38].

The remainder of this paper is organized as follows: We review some related literature on existing schemes in section II. In section III, the system model is presented to provides the information about the network model and the threat assumptions against UAWSNA-IoT. In Section IV, we presents an explanation of our scheme UAWSNA-IoT, which includes four phases such as: registration phase, Login/Authentication phase, changing passwords phase, and adding new sensor nodes phase. We provide a formal and informal security analysis of UAWSNA-IoT in section V. Finally, section VI conducts a comparative evaluation of UAWSNA-IoT and related protocols [27], [14], [20], [25], [6], [35] and [38], considering the costs of computational, storage and communication.

II. RELATED WORKS

This section discusses recent works focusing on authentication schemes to protect a Wireless Sensor Network (WSN) from unauthorized access.

The use of passwords for remote authentication was initially proposed by authors in [10]. This technique relies on one-way hash functions and authentication through session keys and signatures. The utilization of session keys, signatures, and location privacy plays a significant role in addressing specific security vulnerabilities [9, 11, and 36]. In [12] authors introduce the user-authentication protocol in Wireless Sensor Networks (WSNs) using lightweight hash functions and symmetric cryptosystems. However, the authors in [3] identified vulnerabilities in this protocol, including stolen verifier, replay, and forgery attacks. To address these issues, the authors propose a new user-authentication scheme that utilized passwords managed and controlled by a gateway. This approach gained popularity and became widely adopted in authentication systems. However, it lacked mutual authentication and session key security. The mutual authentication is important for verifying the legitimacy of the sender’s identity during a current session. Recent research has focused on protecting user identity by using user anonymity to hide their real identities [14], [15], [17], [18]. Several techniques have been explored, including the use of randomly selected strings as pseudo-identities for users [16]. However, these methods are vulnerable to user tracking attacks, especially when multiple sessions use the same pseudo-identity. To improve security, it is recommended that each new session generates a fresh and random string to verify the user’s true identity. In 2013, the authors in [19] propose an authentication method for WSNs, where temporary credentials are hashed using a one-way hash function. The temporary credentials used in this approach serve as reference information and include a timestamp and user-identity. In 2015, the authors in [16] pointed out the vulnerabilities present in the protocol introduced in [19], highlighting that the scheme is susceptible to attacks user-tracking and identity-guessing attacks. Subsequently, in 2017, researchers in [17] declared that the two-factor authentication mechanism proposed in [16] is exposed to offline guessing and desynchronization attacks, leading to its lack of security. In 2021, the authors in [27] proposed a three-factor authentication scheme for wireless sensor networks in the context of IoT. Nevertheless, the authors in [14] highlighted that this scheme is susceptible to stolen-verifier attacks and lacking perfect forward secrecy. Alternatively, the authors in [14] introduce a secure anonymous three-factor authentication system utilizing elliptic curve cryptography. Regrettably, in this scheme the compromise of a sensor node by an adversary can lead to the retrieval of the user’s identity. In broad terms, protocols that do not employ the Diffie-Hellman key exchange algorithm for session key generation are generally unable to attain perfect forward secrecy [23]. Recently, the authors in [25] introduced a three-factor authentication scheme for wireless sensor networks, utilizing elliptic curve cryptography. Nevertheless, their scheme is susceptible to replay attacks, sensor node capture attacks, and off-line password guessing attacks. Additionally, it lacks the capability to uphold session key secrecy, perfect forward secrecy, anonymity, and unlinkability. In 2021, the authors in [35] introduce an authentication scheme for wireless sensor networks in smart cities. This protocol aimed to resolve various existing flaws in scheme proposed in [42], including vulnerability to offline password guessing attacks and impersonation attacks, along with the absence of session key secrecy, identity unlinkability, and perfect forward secrecy. In 2022, the authors in [22] propose a security-enhanced two-factor authentication scheme for WSN in IoT environment based on ECC, and apply the formal verification using “ProVerif tool” to prove the security of the proposed scheme. In 2023, the author in [38] proposes a wireless sensor network authentication and key-agreement scheme for IoT that uses multiple gateways. However, the scheme has potential vulnerabilities, such as susceptibility to replay and man-in-the-middle attacks.

III. THE SYSTEM MODEL

In this section, we present the network and adversary model. The notations used in UAWSNA-IoT are defined in the table I.

A. Network Model

The network model comprise four participants : System Administrator (SA), Gateway, User, and Sensor Nodes. The user and sensor nodes undergo registration with the gateway via a secure channel. After the registration phase, a process of mutual authentication is initiated among the entities: User,
Gateway, and Sensor Node, respectively. Once the authentication phase has been successfully completed, the communication between users and sensor is established over the public channel using the shared session key. The network model is visually represented in Figure 1.

- **System Administrator (SA):** SA is responsible for generating the confidential parameters, registration, and updating the gateway (GW\(N_j\)). Additionally, SA is responsible to registering the new U\(i\)'s and SN\(i\)'s after the network deployment.
- **User (U\(i\)):** During the registration phase, the user (U\(i\)) with a Smart Card (SC\(i\)) receives their secret parameters from the gateway (GW\(N_j\)). U\(i\) must first be verified by the gateway before being able to access and communicate with a sensor node (SN\(i\)).
- **Gateway (GW\(N_j\)):** GW\(N_j\) is considered a trusted entity responsible for registering every user and sensor node. GW\(N_j\) is responsible to generate the secret parameters for each user (U\(i\)) and sensor node (SN\(i\)) based on their respective identities.
- **Sensor Node (SN\(i\)):** During registration, SN\(i\) receives its secret key from GW\(N_j\). After confirming the legitimacy of U\(i\) through GW\(N_j\), SN\(i\) and U\(i\) establish a session key (SK\(ik\)) to ensure the security of future communications.

### B. Adversary Model

In accordance with the attack model suggested in [26], the adversary “A” model against our protocol “UAWSNA-IoT” is delineated as follows:

- “A” has the capability to intercept all transmitted messages, he/she enable to capturing, replaying, modifying, and rerouting of messages.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA</td>
<td>System Administrator</td>
</tr>
<tr>
<td>SN(i)</td>
<td>(i^{th}) Sensor Node</td>
</tr>
<tr>
<td>U(i)</td>
<td>(i^{th}) User</td>
</tr>
<tr>
<td>GW(N_j)</td>
<td>(j^{th}) Gateway</td>
</tr>
<tr>
<td>ID(gwn)</td>
<td>Identities of the (j^{th}) Gateway</td>
</tr>
<tr>
<td>IDS(i)</td>
<td>Identities of the (i^{th}) Sensor Node</td>
</tr>
<tr>
<td>ID(U)</td>
<td>Identities of the (k^{th}) User</td>
</tr>
<tr>
<td>(\sigma_{G_j})</td>
<td>160 bits Master Private key of (j^{th}) GW(N_j)</td>
</tr>
<tr>
<td>(\alpha_{G_j})</td>
<td>160 bits Mask Key of (j^{th}) GW(N_j)</td>
</tr>
<tr>
<td>(\rho)</td>
<td>160 bits public parameter chosen by SA</td>
</tr>
<tr>
<td>PID(S)</td>
<td>The pseudo-identity of SN(i)</td>
</tr>
<tr>
<td>PID(U)</td>
<td>The pseudo-identity of U(i)</td>
</tr>
<tr>
<td>PID(GW(N_j))</td>
<td>The pseudo-identity of GW(N_j)</td>
</tr>
<tr>
<td>SC(G)</td>
<td>Smart Card of (k^{th}) User</td>
</tr>
<tr>
<td>(p_i)</td>
<td>private key shared with (SN_i) and GW(N_j)</td>
</tr>
<tr>
<td>(p_k)</td>
<td>private key shared with (U_i) and GW(N_j)</td>
</tr>
<tr>
<td>(N_1)</td>
<td>160 bits random numbers generated by (U_k)</td>
</tr>
<tr>
<td>(N_2)</td>
<td>160 bits random numbers generated by (SN_i)</td>
</tr>
<tr>
<td>(ST)</td>
<td>Current timestamp</td>
</tr>
<tr>
<td>(\Delta T)</td>
<td>Maximum time threshold of accepting messages</td>
</tr>
<tr>
<td>PW(G)</td>
<td>K/2&quot;Password associated to user (U_k)</td>
</tr>
<tr>
<td>h(.)</td>
<td>one-way hash function, where h: ({0, 1}^*\rightarrow Z_n)</td>
</tr>
<tr>
<td>SK(i_k)</td>
<td>Session Key shared between (SN_i) and (U_k)</td>
</tr>
</tbody>
</table>

**IV. PROPOSED AUTHENTICATION SCHEME**

In this section, we provide a comprehensive explanation of UAWSNA-IoT. UAWSNA-IoT consists of five phases namely the System setup phase, users/sensor nodes registration phase, login/authentication phase, password renewal phase, and new sensor node addition phase.

#### A. System Setup Phase

The System Administrator (SA) initiates the generation of essential parameters for setup of the gateway (GW\(N_j\)). The steps of this phase are described below:

- **Step 1:** SA chooses the Master private key \(\sigma_{G_j}\), mask key \(\alpha_{G_j}\), and the public system parameter \(n\), each with a size of 160 bits.
- **Step 2:** SA chooses a secure one-way hash function \(h: \{0, 1\}^* \rightarrow Z_n\), and selects an identity (ID\(gwn\)) for the specific gateway (GW\(N_j\)) and proceeds to calculate its pseudo-identity, such as: \(PID_{GW\(N_j\)} = h(ID_{GW\(N_j\)} \parallel \alpha_{G_j})\).
- **Step 3:** SA stores this information \((\sigma_{G_j}, \alpha_{G_j}, ID_{GW\(N_j\)}, h(.) , n)\) in its database (DB\(SA\)) and sends this information \((\sigma_{G_j}, \alpha_{G_j}, PID_{GW\(N_j\)}, h(.) , n)\) to GW\(N_j\) to storing them secretly in its database (DB\(GW\(N_j\)).
- **Step 4:** Afterward, GW\(N_j\) publishes these information \((h(.) , n)\) to sensor nodes and users in registration phase. Finally, SA and GW\(N_j\) use a shared Master Private Key \(\sigma_{G_j}\) as symmetric key for the future communication.

#### B. Registration Phase

This phase is divided into two parts: The Sensor node registration and User registration, as depicted in Figure 2 and Figure 3 respectively. The Both registrations take place over a secure channel.

- **B.1. Sensor Node registration**
C. Login/Authentication Phase

The login/authentication phase between the User, gateway, and Sensor Node is illustrated in Figure 4. This phase can be described as follows:

Step 1: $U_k$ inserts his Smart Card ($SC_k$) into the card reader and inputs its $(IDU_k, PW_k)$; $SC_k$ compute: $\rho_k = h(IDU_k \oplus PW_k)$ and choose a random number $N_1 \in z^*_n$ to compute: $V_1 = vu_k \parallel N_1; V_2 = h(IDU_k || PID_{GW,N_j}) \parallel PIDS_i; V_3 = h(IDU_k || PID_{GW,N_j} || \rho_k || N_1)$.

Finally, $SC_k$ selects current $TS_1$ and sends $Messg_1$ ($V_1, V_2, V_3, PID_{U,k}, TS_1$) to $GW_{N_j}$ over a public channel.

Step 2: Upon receiving $Messg_1$, $GW_{N_j}$ checks $TS_1$ (Time $- TS_1 \leq \Delta T$; Time: represents the current time at which a message is received). If not true, the login request is ignored. Otherwise, $GW_{N_j}$ retrieves $(\rho_k, vu_k)$ associated with $PID_{U_k}$, stored in its database $(DB_{GW,N_j})$.

After that, $GW_{N_j}$ computes: $N_1' = V_1 \parallel vu_k; PIDS_{N_j}' = V_2 \oplus h(PID_{U_k} || PID_{GW,N_j} || \rho_k || N_1')$ and $V_3' = h(PID_{U_k} || PIDS_i || PID_{GW,N_j} || \rho_k || N_1'); V_4 = h(PIDS_i || PID_{GW,N} || N_1 || PID_{U,k})$.

Finally, $GW_{N_j}$ chooses the current $(TS_2)$ and sends $Messg_2$ ($V_4, V_5, V_6, PID_{U,k}, TS_2$) to $SN_i$ over a public channel.

Step 3: Once $SN_i$ receives $Messg_2$, $SN_i$ checks $(Time - TS_2 \leq \Delta T ?)$, if not true, $SN_i$ rejects the request message. Otherwise, $SN_i$ compute: $h^*(PIDS_i || PID_{U,k} || PID_{GW,N_j} || N_1') = V_4 \parallel \rho_k; N_1' = V_5 \oplus h^*(PIDS_i || PID_{U,k} || PID_{GW,N_j} || N_1')$ and $V_6' = h(V_6 \parallel V_6')$.

After that, $SN_i$ checks the legality ($V_6' = V_6$), if not equal ($V_6' \neq V_6$), $SN_i$ rejects the request message. Otherwise, $SN_i$ randomly chooses a random number $N_2 \in z^*_n$ and current $TS_3$. After, $SN_i$ compute: $V_2 = \rho_k \oplus N_2; \gamma = N_1' \parallel N_2; \rho_{new} = h(\rho_k || N_2); V_8 = h(h^*(PIDS_i || PID_{U,k} || PID_{GW,N_j} || N_1') || N_2 || \gamma); SK_{dk} = h(h^*(PIDS_i || PID_{U,k} || PID_{GW,N_j} || N_1') || N_2 || \gamma)$.

Finally, $SN_i$ replaces $\rho_k$ with $\rho_{new}$ in its memory, chooses the current $TS_3$, and sends $Messg_3$ ($V_7, V_8, TS_3$) to $GW_{N_j}$.

Step 4: Upon receiving $Messg_3$, $GW_{N_j}$ checks $(Time - TS_3 \leq \Delta T ?)$, if not true, $GW_{N_j}$ aborts the session.
Otherwise, $GW_{N_j}$ compute: $N'_j = V_t \oplus \rho_i; \gamma' = N_1 \oplus N'_j; V'_9 = h^{\gamma'}(PIDS'_k || PIDS'_k || PIDS_{GW_{N_j}} || N'_j || N_2 || \gamma')$.

- After that, $GW_{N_j}$ checks equality ($V'_9 = V_8$ ?), if not equal, then $GW_{N_j}$ rejects the request message. Otherwise, $GW_{N_j}$ compute $\rho_i^{\text{new}} = h(\rho_i || N_2); SK_{ik} = h(h^{\gamma}(PIDS'_k || PIDS'_k || PIDS_{GW_{N_j}} || N'_j || N_2 || \gamma'); V_9 = N'_j \oplus h^{\gamma}(PIDS'_k || PIDS_{GW_{N_j}} || N'_j); PIDS_{GW_{N_j}}^{\text{new}} = h(PIDS_{GW_{N_j}} || N'_2), and replaces $PIDS_{GW_{N_j}}$ with $PIDS_{GW_{N_j}}^{\text{new}}$ and $\rho_i$ associated to $PIDS'_k$ with $\rho_i^{\text{new}}$ in its $DB_{GW_{N_j}}$.

- Finally, $GW_{N_j}$ chooses current $TS_4$ and sends $Messg_4(V_5, V_6, V_7, TS_4)$ to $U_k$.

Step 5: Upon receiving $Messg_4$, $SC_k$ checks ($Time - TS_4 \leq \Delta T$ ?), if not true, the request message is ignored. Otherwise, $SC_k$ compute: $h^{\gamma}(PIDS'_k || PIDS_{GW_{N_j}} || PIDS_{GW_{N_j}} || N'_j || \gamma = N_1 \oplus N'_2; V'_9 = h^{\gamma}(PIDS'_k || PIDS_{GW_{N_j}} || PIDS_{GW_{N_j}} || N'_j || N_1 || N'_2 || \gamma')$.

- After, $SC_k$ Checks ($V'_9 = V_8$ ?), if not equal, then $SC_k$ rejects the request message. Otherwise, $SC_k$ computes: $SK_k = h^{\gamma}(PIDS'_k || PIDS_{GW_{N_j}} || PIDS_{GW_{N_j}} || N'_j || \gamma); PIDS_{GW_{N_j}}^{\text{new}} = h(PIDS_{GW_{N_j}} || N'_2)$ and replaces $PIDS_{GW_{N_j}}$ with $PIDS_{GW_{N_j}}^{\text{new}}$ in its memory; $\sigma$.

D. Password Renewal Phase

The user ($U_k$) has the autonomy to change their password ($PW_k$) at regular intervals, without involving $GW_{N_j}$ or $SA$. This process is solely between $U_k$ and their own $SC_k$ and is optional. The following steps are involved in this phase:

1) $U_k$ inserts its $SC_k$ into the card reader, then he/she enters his $ID_{U_k}$ and old password $PW_k$. $SC_k$ needs to compute the private key $\rho_k$ such as: $\rho_k = \rho_k^s \oplus h(ID_{U_k} \parallel PW_k)$ to compute the mask: $MPW = h(ID_{U_k} \parallel PW_k \parallel \rho_k)$.

2) After that, $SC_k$ check the equality ($MPW = MPW$ ?), if not equal, then $SC_k$ rejects the password change request. Otherwise, $SC_k$ will ask $U_k$ to enter the new password $PW_k^{\text{new}}$ according to its choices.

3) After $U_k$ enters his new password $PW_k^{\text{new}}$, then $SC_k$ calculates the new mask $MPW^{\text{new}}$ according to $PW_k^{\text{new}}$ such as: $MPW^{\text{new}} = h(ID_{U_k} \parallel PW_k^{\text{new}} \parallel \rho_k)$.

4) Afterwards, $SC_k$ computes: $\rho_k^{\text{new}}$ and $vu_k^{\text{new}}$ according to the new password $PW_k^{\text{new}}$ such that: $\rho_k^{\text{new}} = h(ID_{U_k} \parallel PW_k^{\text{new}} \parallel \rho_k); vu_k^{\text{new}} = h(ID_{U_k} \parallel PW_k^{\text{new}} \parallel vu_k^{\text{new}})$.

5) Finally, $SC_k$ stores these new values ($\rho_k^{\text{new}}, vu_k^{\text{new}}, MPW^{\text{new}}$) in its memory secretly by replacing the old values.

E. Sensor Node Addition phase

To achieve scalability, UAWSNA-IoT needs to be adapted to dynamically integrate the new sensor nodes $SN_i^{\text{new}}$. In this phase, The System Administrator (SA) is responsible for registering new sensor node $SN_i^{\text{new}}$ even in the absence of gateway ($GW_{N_j}$). The steps followed by SA are as follows:

- SA chooses the appropriate $GW_{N_j}$, where $SN_i^{\text{new}}$ will be deployed in order to calculate their confidential parameters.

- Subsequently, both $SA$ and $SN_i^{\text{new}}$ follow the identical steps as outlined in the sensor node registration phase mentioned above in the section IV.B.1.

- Upon the registration phase of $SN_i^{\text{new}}$ is completed, $SA$ utilizes the shared symmetric key: $K_{e_{\text{GW}_{N_j}}} = \sigma_{ij}$ to encrypt these information ($PIDS_i^{\text{new}}$, and $\rho_i^{\text{new}}$) associated with new $SN_i^{\text{new}}$, and transmits them to the corresponding $GW_{N_j}$.

- Upon receiving this information, $GW_{N_j}$ decrypt them and store its in $DB_{GW_{N_j}}$. Ultimately, $SA$ deploys $SN_i^{\text{new}}$ in the chosen capture area. Following this deployment $U_k$ is promptly notified about this new addition to including the pseudo-identity ($PIDS_i^{\text{new}}$) associated with $SN_i^{\text{new}}$ in his smart card $SC_k$.

V. Security Analysis

In this section, we conduct a comprehensive security evaluation of UAWSNA-IoT using both formal and informal analyses.

A. Formal Security Analysis using (ROR) Model

The Real-Or-Random (ROR) model [8] is employed to evaluate the session key security in UAWSNA IoT protocol’s. In this model, the network is vulnerable to various attacks conducted by an adversary “A”, including eavesdropping, capturing, inserting, and deleting messages [26]. In the security analysis, we use symbols $\prod_{i=1}^{t}, \prod_{i=1}^{t}, \prod_{i=1}^{t}$ to represent specific instances denoted by $t, u$ and $v$, respectively which act as oracles in the system. We apply the principles of the ROR model to UAWSNA-IoT scheme, where “A” possesses the capability to execute different attacks, as indicated by the following queries:

- Execute($\prod_{i=1}^{t}, \prod_{i=1}^{t}, \prod_{i=1}^{t}$): “A” performs this type of query in order to intercept the messages exchanged between the oracles of legitimate participants. This query models a passive-type attack.

- Send ($\prod_{i=1}^{t}, M$): The goal of this query is to simulate an active attack. By executing this query, “A” is capable of sending a message $M$ to a participating instance $\prod_{i=1}^{t}$ and receiving a response message in return.

- Test ($x, i$): If the oracle accepted and has the session key ($SK_{ik}$), then a bit $b$ is chosen randomly. If $b = 1$, then “A” gets the freshly $SK_{ik}$, else, if $b = 0$, it means “A” gets a random $SK_{ik}$. However, if $b = 0$ and $b = 1$, “A” gets a NULL value. This query is used to model an attacker’s ability to distinguish between a real and a random $SK_{ik}$. To ensure the security of $SK_{ik}$ in UAWSNA-IoT, “A” can never distinguish between a random and the real session key ($SK_{ik}$) generated as a result.

- Reveal ($x, i$): If the oracle $\prod_{i=1}^{t}$ is accepted and has a session key ($SK_{ik}$), then we give $SK_{ik}$ to “A”. This model simulates the robustness of UAWSNA-IoT, i.e. that is disclosing a $SK_{ik}$ affects only the current session.
CorruptSC (\(\Pi_{SC}^{j}\)): The goal of this query is to model the smart card (SC\(_{k}\)) loss attack. This query produces the output to indicate that “A” extracts all parameters stored in SC\(_{k}\).

- Theorem 1

Let A, try to get the session key of our UAWSNA-IoT protocol in polynomial time \(t\) by following Real-Or-Random (RoR) model. Let \(Adv^{(UAWSNA-IoT)}(A)\), represents the probability that the adversary “A” will be successful in breaking the session key. Let \(q_{hash}\), \(q_{send}\), HASH, represent the number of hash requests, number of send requests and hash function space domain \(h(\cdot)\) respectively. The parameters \(s\) and \(C\) are the Zipf’s parameters defined in [28].

- The formal proof

Following the proof technique used in [29], [30], we perform four game rounds called Game\(_{j}\), where \(j\) is \([0, 3]\). Let Succ\(_{(A, game_{j})}\) be an event where “A” can guess the correct bit \(b\) in the game Game\(_{j}\) with a probability equal to \(Pr[Succ_{(A, game_{j})}]\). The game starts with Game\(_{1}\) which is a real attack, and ends with Game\(_{3}\). We can accomplish Game\(_{j}\) as follows with these parameters:

- **Game\(_{0}\)**: This game models a real attack performed by adversary “A” against UAWSNA-IoT based on RoR model. Initially, Game\(_{0}\) randomly chooses a bit \(b\) which we can derive as follows:

\[
Adv^{(UAWSNA-IoT)}(A) = 2Pr[Succ_{(A, game_{0})}] - 1 \quad (1)
\]

- **Game\(_{1}\)**: We assume that “A” intercepts the messages Message\(_{1}\)\([V_{1}, V_{2}, V_{3}, PIDU_{k}, TS_{1}]\), Message\(_{2}\)\([V_{4}, V_{5}, V_{6}, PIDU_{k}, TS_{2}]\), Message\(_{3}\)\([V_{4}, V_{5}, V_{6}, TS_{3}]\), and Message\(_{4}\)\([V_{5}, V_{6}, V_{9}, TS_{4}]\) using Execute (\(\Pi_{U}^{1}, \Pi_{G}^{1}, \Pi_{SN}^{1}\)) query. Then “A” executes Test() and Reveal() queries to obtain SK\(_{ik}\). SK\(_{ik}\) is computed using the following secret parameters: PIDU\(_{k}\), PID\(_{i}\), PIDGWN\(_{j}\) and \(\gamma = N_{1} \oplus N_{2}\) such as SK\(_{ik}\) = \(h^{(h(PIDU_{i}) \parallel PIDU_{k}) \parallel PIDGWN_{j}) \parallel N_{1}) \parallel \gamma}\). So “A” needs the pseudo-identity’s PIDGWN\(_{j}\) and PID\(_{i}\) associated with GWN\(_{j}\) and SN\(_{i}\) respectively. In addition, the random numbers \(N_{1}\) and \(N_{2}\) as the main parameters to calculate SK\(_{ik}\). “A” is unable to compute a real SK\(_{ik}\) without knowledge of these secret parameters. This means Game\(_{0}\) and Game\(_{1}\) are indistinguishable. Therefore, the probability that “A” is a winner of Game\(_{1}\) remains similar to Game\(_{0}\):

\[
Pr[Succ_{(A, game_{1})}] = Pr[Succ_{(A, game_{0})}] \quad (2)
\]

- **Game\(_{2}\)**: In this game, “A” perform the Send() and HASH() queries which are an active attack. So, “A” uses Message\(_{1}\), Message\(_{2}\), Message\(_{3}\) and Message\(_{4}\) exchanged to get SK\(_{ik}\). But, these messages contain the values which are embedded in HASH() query. More precisely, the values \([V_{2}, V_{3}, V_{4}, V_{5}, V_{6}]\) are calculated according to random number \(N_{1}\). Additionally, the values \(V_{7}, V_{8}\) and \(V_{9}\) are computed using \(N_{1}\) and \(N_{2}\). We use random num-b
bers \( N_1 \) and \( N_2 \) to prevent collision between the sessions. In addition, UAWSNA-IoT use the timestamps \( (T_S_1, T_S_2, T_S_3, \text{ and } T_S_4) \) associated with the exchanged messages. Therefore, “A” cannot get the collision cases in the hash function according to the birthday paradox [34], we can get the following equation:

\[
| Pr[Succ(A, game_2)] - Pr[Succ(A, game_1)] | \leq q^2_h | HASH |
\]

(3)

**Game_3:** In this game, Adversary “A” employs the CorruptSC(i) query to simulate stolen smart card (SC_k) attacks. “A” has the capability of obtaining the following information: \( MPW, \rho^i_k, \nu^i_k, PIDS_i, \text{ and } PID_{GW_i,j} \) stored in SC_k by using the power analysis attack. Assuming that the password \( PW_k \) has low entropy, “A” tries to use a brute-force attack with an online dictionary, exploiting the information extracted from SC_k. However, we assume that our system limits the number of attempts to enter the correct PW_k. Therefore, “A” cannot obtain the required secret information \( \rho^i, \text{ and } PW_k \) from the parameters \( \rho^i_k, \nu^i_k, \text{ and } MPW \) extracted from the SC_k. This difficulty lies in the adversary inability to know \( IDU_k \) and \( PW_k \) at the same time, since they are embedded in the MPW. Therefore, “A” cannot distinguish between Game_2 and Game_3, without knowing \( IDU_k \) and \( PW_k \), which is an impossible task. For this reason, we get the result according to Zipf’s law [28].

\[
| Pr[Succ(A, game_3)] - Pr[Succ(A, game_1)] | \leq C' q^i_{send}
\]

(4)

Afterwards, “A” acquires the guessed bit \( b \), because the games are over.

\[
Pr[Succ(A, game_3)] = \frac{1}{2}
\]

(5)

From equations (1) and (2), we deduce the following result:

\[
\frac{1}{2} Ado^{(UAWSNA−IoT)}(A) = | Pr[Succ(A, game_1)] - \frac{1}{2} |
\]

(6)

We apply equations (5) and (6) to easily obtain the following equation:

\[
\frac{1}{2} Ado^{(UAWSNA−IoT)}(A) = | Pr[Succ(A, game_1)] - Pr[Succ(A, game_2)] |
\]

(7)

We apply the triangular inequality, we easily obtain the following result:

\[
\frac{1}{2} Ado^{(UAWSNA−IoT)}(A) = | Pr[Succ(A, game_1)] - Pr[Succ(A, game_2)] | \leq \frac{q^2_h | HASH |}{2} + C' q^i_{send}
\]

\[
| Pr[Succ(A, game_1)] - Pr[Succ(A, game_2)] | \leq \frac{q^2_h | HASH |}{2} + C' q^i_{send}
\]

Finally, by multiplying the formula (8) by 2, we will obtain the following equation:

\[
Ado^{(UAWSNA−IoT)}(A) \leq \frac{q^2_h | HASH |}{2} + 2 C' q^i_{send}
\]

**B. Formal Security Analysis Using BAN Logic**

In this section we prove that UAWSNA-IoT fulfills all conditions to achieve mutual authentication using BAN-logic analysis [39]. The exchange messages:

- **Messg_1:** \( U_k \rightarrow GW_N_j : \{ PIDS_1, N_1, TS_1 \}_{PK} \);
- **Messg_2:** \( GW_N_j \rightarrow SN_i : \{ N_2, h(PIDS_1') \parallel PIDU_k' \parallel PID_{GW_i,j}, \| N_1 \}, TS_2 \}_{PK} \);
- **Messg_3:** \( SN_i \rightarrow GW_N_j : \{ N_2, TS_3 \}_{PK} \);
- **Messg_4:** \( GW_N_j \rightarrow U_k : \{ N_2, h(PIDS_i' \parallel PIDU_k' \parallel PID_{GW_i,j}, \| N_1 \}, TS_2 \}_{PK} \).

- The forms idealized of the exchanged messages:

- **Messg_1:** \( U_k \rightarrow GW_N_j : \{ PIDS_1, N_1, TS_1 \}_{PK} \);
- **Messg_2:** \( GW_N_j \rightarrow SN_i : \{ N_1, h(PIDS_1') \parallel PIDU_k' \parallel PID_{GW_i,j}, \| N_1 \}, TS_2 \}_{PK} \);
- **Messg_3:** \( SN_i \rightarrow GW_N_j : \{ N_2, TS_3 \}_{PK} \);
- **Messg_4:** \( GW_N_j \rightarrow U_k : \{ N_2, h(PIDS_i' \parallel PIDU_k' \parallel PID_{GW_i,j}, \| N_1 \}, TS_2 \}_{PK} \).

**- BAN-logic rules:**

\- **Rule 1:** Message-meaning rule: \( A \models \{ \text{Message meaning} \} \)
\- **Rule 2:** Nonce-verification rule: \( A \models \{ \text{Nonce verification} \} \)
\- **Rule 3:** Jurisdiction rule: \( A \models \{ \text{Jurisdiction} \} \)
\- **Rule 4:** Belief rule: \( A \models \{ \text{Belief} \} \)
\- **Rule 5:** Freshness rule: \( A \models \{ \text{Freshness} \} \)

**- Proof**

We define the following goals:

- **Goal (1):** \( U_k \models U_k^{SK_{IK}} GW_N_j \)
- **Goal (2):** \( U_k \models GW_N_j \models U_k^{SK_{IK}} SN_i \)
- **Goal (3):** \( GW_N_j \models U_k^{SK_{IK}} GW_N_j \)
- **Goal (4):** \( GW_N_j \models U_k^{SK_{IK}} SN_i \)

**TABLE II**

<table>
<thead>
<tr>
<th>NOTATION USED IN BAN-LOGIC PROOF</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SIGNIFICATION</strong></td>
</tr>
<tr>
<td>A, B</td>
</tr>
<tr>
<td>X, Y</td>
</tr>
<tr>
<td>SK_{IK}</td>
</tr>
<tr>
<td>A \models B</td>
</tr>
<tr>
<td>#(B)</td>
</tr>
<tr>
<td>A \leftarrow B</td>
</tr>
<tr>
<td>A \models { X }</td>
</tr>
<tr>
<td>A \models X</td>
</tr>
<tr>
<td>{ X }_{key}</td>
</tr>
<tr>
<td>A \models B</td>
</tr>
</tbody>
</table>
• Goal (5): \( SN_i \equiv SN_i^{SK_{k_i}} GW \); 
• Goal (6): \( SN_i \equiv GW_j \equiv SN_i^{SK_{k_j}} GW \); 
• Goal (7): \( GW_j \equiv SN_i^{SK_{k_j}} GW \); 
• Goal (8): \( GW_j \equiv SN_i \equiv SN_i^{SK_{k_j}} GW \);
we will define the following assumptions:
• A(1): \( GW_j \equiv \#(N_1) \); 
• A(2): \( GW_j \equiv \#(N_2) \); 
• A(3): \( U_k \equiv \#(N_2) \); 
• A(4): \( SN_i \equiv \#(h(PIDS_i || PIDU_k || PID_{GW_j} || N_1)) \); 
• A(5): \( U_k \equiv GW_j \Rightarrow (U_k^{SK_{k_j}} GW) \); 
• A(6): \( GW_j \equiv U_k \Rightarrow (U_k^{SK_{k_j}} GW) \); 
• A(7): \( SN_i \equiv GW_j \Rightarrow (SN_i^{SK_{k_j}} GW) \); 
• A(8): \( GW_j \equiv SN_i \Rightarrow (SN_i^{SK_{k_j}} GW) \); 
• A(9): \( U_k \equiv (U_k^{\#(N_1)} GW_j) \); 
• A(10): \( GW_j \equiv (U_k^{\#(N_1)} GW_j) \); 
• A(11): \( SN_i \equiv (SN_i^{\#(N_1)} GW_j) \); 
• A(12): \( GW_j \equiv (SN_i^{\#(N_1)} GW_j) \).

-Proof postulates

Step1 : \( D_1 \) can be acquired from \( \text{Msg} q_1 \):
\( D_1 : GW_j \& \{ PIDS_i, N_1, T_{S1} \} \).

Step2 : \( D_2 \) can be derived by applying Rule 1 using \( D_1 \) and \( A(10) : D_2 : GW_j \equiv U_k \equiv (PIDS_i, N_1, T_{S1}). \)

Step3 : \( D_3 \) is induced by applying the Rule 5 using \( A(1) : D_3 : GW_j \equiv \#(PIDS_i, N_1, T_{S1}). \)

Step4 : \( D_4 \) is induced by applying the Rule 2 using \( D_2 \) and \( D_3 : D_4 : GW_j \equiv U_k \equiv (PIDS_i, N_1, T_{S1}). \)

Step5 : \( D_5 \) is induced from rule 4 using \( D_4 : D_5 : GW_j \equiv U_k \equiv (N_1). \)

Step6 : \( D_6 \) is obtained from \( \text{Msg} q_2 : D_6 : SN_i \mid \#(N_1, h(PIDS_i || PIDU_k || PID_{GW_j} || N_1), T_{S2}). \)

Step7 : \( D_7 \) is deduced from Rule 1 using \( D_6 \) and \( A(11) : D_7 : SN_i \equiv GW_j \equiv (N_1, h(PIDS_i || PIDU_k || PID_{GW_j} || N_1), T_{S2}). \)

Step8 : \( D_8 \) is obtained from applying Rule 5 using \( A(4) : D_8 : SN_i \equiv \#(N_1, h(PIDS_i || PIDU_k || PID_{GW_j} || N_1), T_{S2}). \)

Step9 : \( D_9 \) can be derived by applying Rule 2 using \( D_7 \) and \( D_8 : D_9 : SN_i \equiv GW_j \equiv (N_1, h(PIDS_i || PIDU_k || PID_{GW_j} || N_1), T_{S2}). \)

Step10 : \( D_{10} \) is deduced by applying Rule 4 using \( D_9 \), we obtain: \( D_{10} : SN_i \equiv GW_j \equiv (N_1, h(PIDS_i || PIDU_k || PID_{GW_j} || N_1)). \)

Step11 : \( D_{11} \) is obtained from \( \text{Msg} q_3 : D_{11} : GW_j \equiv \{ N_2, T_{S3_1} \}. \)

Step12 : \( D_{12} \) is deduced by applying Rule 1 using \( D_{11} \) and \( A(12) : D_{12} : GW_j \equiv SN_i \equiv (N_2, T_{S3}). \)

Step13 : \( D_{13} \) is obtained by applying Rule 5 and Rule 2 using \( A(2) \) and \( D_{12} \) respectively: \( D_{13} : GW_j \equiv SN_i \equiv (N_2, T_{S3}). \)

Step14 : \( D_{14} \) is deduced by applying Rule 4 using \( D_{13} : D_{14} : GW_j \equiv SN_i \equiv (N_2). \)

Step15 : \( D_{15} \) and \( D_{16} \) are deduced from \( D_{10} \) and \( D_{14} : SN_i \) and \( GW_j \) compute the session key \( SK_{k_i} = h'(h(PIDS_i || PIDU_k || PID_{GW_j} || N_1 || \gamma)) \) such as \( \gamma = N_1 \oplus N_2. \)

\( D_{15} : GW_j \equiv SN_i \equiv SN_i^{SK_{k_i}} GW_j \) (Goal 8).

\( D_{16} : SN_i \equiv GW_j \equiv SN_i^{SK_{k_i}} GW_j \) (Goal 6).

Step16 : \( D_{17} \) and \( D_{18} \) are obtained applying Rule 3 using \( D_{15} \) and \( A(8) \), and \( D_{16} \) and \( A(7) \) respectively.

\( D_{17} : GW_j \equiv SN_i \equiv SN_i^{SK_{k_i}} GW_j \) (Goal 5).

\( D_{18} : SN_i \equiv SN_i^{SK_{k_i}} GW_j \) (Goal 7).

Step17 : \( D_{19} \) is obtained from \( \text{Msg} q_4 : D_{19} : U_k \equiv \{ N_2, h(PIDS_i || PIDU_k || PID_{GW_j} || N_1), T_{S4} \}. \)

Step18 : \( D_{20} \) is deduced from Rule 1 using \( D_{19} \) and \( A(9) : D_{20} : U_k \equiv GW_j \equiv (N_2, h(PIDS_i || PIDU_k || PID_{GW_j} || N_1), T_{S4}). \)

Step19 : \( D_{21} \) is obtained by applying Rule 5 using \( A(3) : D_{21} : U_k \equiv \#(N_2, h(PIDS_i || PIDU_k || PID_{GW_j} || N_1), T_{S4}). \)

Step20 : \( D_{22} \) is deduced by applying Rule 2 using \( D_{19} \) and \( D_{26} : D_{22} : U_k \equiv GW_j \equiv (N_2, h(PIDS_i || PIDU_k || PID_{GW_j} || N_1), T_{S4}). \)

Step21 : \( D_{23} \) is deduced applying Rule 4 using \( D_{22} : D_{23} : U_k \equiv GW_j \equiv (N_2, h(PIDS_i || PIDU_k || PID_{GW_j} || N_1)). \)

Step22 : \( D_{24} \) and \( D_{25} \) are deduced by applying \( D_5 \) and \( D_23 : U_k \) and \( GW_j \) can compute the Session Key \( SK_{k_i} = h(h(PIDS_i || PIDU_k || PID_{GW_j} || N_1) || \gamma) \) such as \( \gamma = N_1 \oplus N_2. \)

\( D_{24} : U_k \equiv GW_j \equiv U_k^{SK_{k_i}} GW_j. \) (Goal 2).

\( D_{25} : GW_j \equiv U_k \equiv U_k^{SK_{k_i}} GW_j. \) (Goal 4)

Step23 : \( D_{26} \) and \( D_{27} \) are deduced by applying Rule 3 using \( D_{24} \) and \( A(5) \), and \( D_{25} \) and \( A(6) \) respectively.

\( D_{26} : U_k \equiv U_k^{SK_{k_i}} GW_j. \) (Goal 1).

\( D_{27} : GW_j \equiv U_k^{SK_{k_i}} GW_j. \) (Goal 3).
C. Informal Security Analysis

This section provides an informal analysis of the performance and effectiveness of UAWSNA-IoT against some of the most know attacks.

1) Privileged Insider Attack: In the scenario where a privileged insider adversary “A”, intercepts the registration message $ID_U$ from a legitimate user $U_k$, “A” endeavours to calculate $U_k$’s session key using the messages specified in Login/Authentication phase. Nevertheless, “A” is unable to calculate $U_k$’s session key $SK_{ik}$. In order to calculate $SK_{ik} = h(h(PIDS_k^i || PIDU_k^i || PID_{GWN_j}) || N_j) || γ)$, such as $γ = N_1 ⊕ N_2$, “A” must be aware of the both random numbers $N_1$ and $N_2$ generated in each session. As “A” lacks information about the values of $N_1$ and $N_2$, it is unable to compute the correct $SK_{ik}$. As a result, UAWSNA-IoT has the resistance against the stolen verifier attacks.

2) Stolen smart card attack: If adversary “A” steals a legitimate user’s smart card (SCj) using a power analysis attack [40] to extract its stored data. This data comprises $PIDU_k$, $vU_k^s$, $ρ_k^s$, $PIDS_{j}$, and $MPW$ obtained by the following operations: $vU_k^s = h(IDU_k || PW_k) ⊕ vU_k^i$; $ρ_k^s = h(IDU_k || PW_k) ⊕ ρ_k$ and $MPW = h(IDU_k || PW_k)$ . While the adversary “A” may be able to make guesses about the user’s password $PW_k$, they lack the necessary knowledge about the user’s identity $ID_U$. As a result, UAWSNA-IoT effectively defends against stolen smart card attacks.

3) Offline password guessing attack: If adversary “A” succeeds in extracting the information ($ρ_k^s$, $vU_k^s$, $PIDU_k$, $PIDS_j$, and $MPW$) stored in the smart card (SCj) memory via a power analysis attack [40]. Then “A” tries to impersonate $U_k$ and tries to guess $IDU_k$ and $PW_k$ by extracting them from the knowing values $ρ_k^s$, $vU_k^s$ and $MPW$ that as: $vU_k^s = h(IDU_k || PW_k) ⊕ vU_k^i$; $ρ_k^s = h(IDU_k || PW_k) ⊕ ρ_k$ and $MPW = h(IDU_k || PW_k)$ . If “A” is unable to find $IDU_k$ and $PW_k$ only based on the data stored in SCj. As they are merged and hashed alongside other values within parameters such as $PIDU_k$, $ρ_k$, and $MPW$. Therefore, UAWSNA-IoT is secure against offline password guessing Attack.

Therefore, UAWSNA-IoT is secure against offline password guessing Attack.

4) Stolen Verifier Attack: Suppose an adversary “A”, illicitly acquires the database $D_{GWN_j}$ of $GWN_j$, which includes $α_{Gj}$, $α_{Gj}$, $ρ_k$, $PIDU_k$, $vU_k$, $ρ_k$, $PIDS_j$, $PID_{GWN_j}$. Nevertheless, “A” is unable to calculate the session key for the legitimate user $U_k$ using these parameters. To compute the session key $SK_{ik} = h(h^*(PIDS_k^i || PIDU_k^i || PID_{gw_{w_{i}}}) || N_j^i) || γ)$ such as $γ = N_1 ⊕ N_2$, “A” must be aware of the both random numbers $N_1$ and $N_2$ generated in each session. As “A” lacks information about the values of $N_1$ and $N_2$, it is unable to compute the correct $SK_{ik}$. As a result, UAWSNA-IoT has the resistance against the stolen verifier attacks.
compute the correct session key unless in possession of the current random numbers. Consequently, UAWSNA-IoT is secure against known session key attacks.

9) Perfect Forward Secrecy: In UAWSNA-IoT, the session key is computed using the expression $SK_{ik} = h( h^*(PIDS_i) \| PIDS_j \| PID_{GWNI} \| N_i \| \gamma_i^* )$ with $\gamma = N_1 \oplus N_2$. The values $N_1$ and $N_2$ represent random numbers generated by $U_k$ and $SN_i$, respectively. Despite that the adversary “$A$” know all the keys ($\rho_k, \rho_i$), including the password $PMS_k$ and the shared secret parameters $PID_{GWNI}, PIDS_i, \text{and } PID_{U_i}$. “$A$” cannot determine the current or past session keys without knowing the values of the random numbers $N_1, N_2$. As a result, it is evident that UAWSNA-IoT effectively guarantees perfect forward secrecy.

10) Sensor node capture Attack: We assume a scenario where an adversary “$A$”, takes control of a particular sensor node $SN_i$ and obtains shared key $(\rho_i)$ from $SN_i$’s memory by employing a power analysis attack[40]. Subsequently, “$A$” has the capability to authenticate with gateway $GWNI$ and user $U_k$. Nevertheless, “$A$” does not pose a threat to other sensor nodes. As the shared secret key $(\rho_i)$ is determined by the formula $\rho_i = h(IDSN_i \| \sigma_G)$, “$A$” is limited to authenticating solely with the particular sensor node $SN_i$. “$A$” is incapable of computing any information pertaining to other sensor nodes. Thus, UAWSNA-IoT effectively withstands sensor node capture attacks.

11) Man-in-middle attack: During the login/authentication phase, $GWNI$ verifies the authenticity of $U_k$ by validating the shared secret key $(\rho_i)$ and the associated value $v_u$ in its $DB_{GWNI}$. Similarly, $SN_i$ can authenticate $GWNI$ by leveraging its knowledge of $SN_i$’s secret key $(\rho_i)$. Furthermore, $GWNI$ can identify $SN_i$ through his $h^*(PIDS_i) \| PID_{U_k} \| PID_{GWNI} \| N_i^1$) knowledge. Finally, $U_k$ authenticates $GWNI$ through his $N_i$ knowledge. As a result, all participants are able to mutually authenticate each other. This robust authentication mechanism makes UAWSNA-IoT resistant to man-in-the-middle attacks.

![TABLE III

<table>
<thead>
<tr>
<th>ATTACK</th>
<th>[27]</th>
<th>[14]</th>
<th>[20]</th>
<th>[25]</th>
<th>[6]</th>
<th>[35]</th>
<th>[38]</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
</tr>
<tr>
<td>A2</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
</tr>
<tr>
<td>A3</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
</tr>
<tr>
<td>A4</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
</tr>
<tr>
<td>A5</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
</tr>
<tr>
<td>A6</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
</tr>
<tr>
<td>A7</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
</tr>
<tr>
<td>A8</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
</tr>
<tr>
<td>A9</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
</tr>
<tr>
<td>A10</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
</tr>
<tr>
<td>A11</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
</tr>
<tr>
<td>A12</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
</tr>
</tbody>
</table>

A1: Privileged Insider Attack; A2: Stolen smart card attack; A3: Offline password guessing Attack; A4: Stolen Verifier Attack; A5: Mutual Authentication; A6: Replay Attacks; A7: Anonymity; A8: unlinkability; A9: Known session key attack; A10: Perfect Forward Secrecy; A11: Sensor node capture Attack; A12: Man-in-middle attack o The protocol s secure x: The protocol is not secure.

VI. PERFORMANCE ANALYSIS

In this section, we evaluate the proposed scheme by contrasting it with various related schemes ([27], [14], [20], [25], [6], [35], and [38]) concerning security performances and computational, communication, and storage costs. The corresponding results are presented in Tables III, IV, and V respectively.

A. Security Performance Evaluation

We compare the security performance of UAWSNA-IoT with related protocols ([27], [14], [20], [25], [6], [35], and [38]) against various well-known attacks. As shown in Table III, UAWSNA-IoT provides superior security performance when compared to related protocols.

B. Computational Cost

We assess the computational costs of UAWSNA-IoT and compare its performance to the related protocols ([27], [14], [20], [25], [6], [35], and [38]) as is presented in Table IV. In our study, we symbolize $T_h, T_{Ecomm}$ and $T_{E/D}$, which respectively represent the time to execute the hash function 0.068 millisecond (ms), ECC points multiplication 2.501 ms, and symmetric encryption/decryption 0.56 ms respectively [30]. In our study, we did not take into account the computational cost of the XOR operation as it is considered negligible. According to our study, the computational cost of UAWSNA-IoT is lower compared to the related protocols ([27], [14], [25], [6], [35], and [38]) at both the sensor node and network levels. However, it does a higher computational cost than the protocol ([20]). Unfortunately, the protocol [20] is susceptible to several attacks, as shown in Table III.

C. Communication Cost

In this section, we will evaluate the communication cost of our protocol UAWSNA-IoT in comparison to protocols [27], [14], [20], [25], [6], [35], and [38]. We will assume that the output size of hash function $h(\cdot)$, random number, and timestamp ($TS_i$ such as $i\in[1,4]$) are 160, 160, and 32 bits, respectively. In order to compute the communication costs of UAWSNA-IoT, we consider the values ($V_i, i = 1,..,9$) that go into the messages calculation used in the authentication phase, which have a size of 160 bits. We also calculate the size of the messages sent by each entity ($U_k, GWNI, and SN_i$) separately in authentication phase. The size of the messages exchanged are: $Messq_{\{V_i, V_j, PID_{U_i}, TS_1\}}$ requires (4 $\times$ 160 $+$ 32 $=$ 672 bits), $Messq_{\{V_i, V_j, PID_{U_i}, TS_2\}}$ requires (4 $\times$ 160 $+$ 32 $=$ 672 bits), $Messq_{\{V_i, V_j, TS_3\}}$, and $Messq_{\{V_i, V_j, V_k, TS_4\}}$, requires (2 $\times$ 160 $+$ 32 $=$ 352 bits) and (3 $\times$ 160 $+$ 32 $=$ 512 bits) respectively. Thus, the total communication cost for the three entities $U_k, GWNI, and SN_i$ are 672 + 672 + 352 + 512 $=$ 2 208 bits. The Table V shows the communication costs comparison of UAWSNA-IoT and related schemes ([27], [14], [20], [25], [6], [35], and [38]).
It can be concluded that UAWSNA-IoT is more efficient in terms of total communication costs than the related schemes ([27], [20], [25], [6], [35], and [38]). However, UAWSNA-IoT has a higher total communication cost than the protocol ([14]). Unfortunately, the scheme ([14]) is insecure due to its vulnerability to sensor node capture attack. Moreover, at the sensor node level, UAWSNA-IoT boasting a reduced message size compared to protocols ([14], [20], [25], [6], [35], and [38]), while higher then protocol [27]. But the protocol [27], which is not secure against some attacks as presented in Table III.

VII. Conclusion

In this paper, we presented a new user authentication protocol to secure the wireless sensor networks access using two factors called “UAWSNA-IoT”. UAWSNA-IoT also mitigates network congestion by decreasing the size of authentication messages during the authentication phase. Furthermore, UAWSNA-IoT allows users access to data in WSN after their authentication process conducted by gateway via internet. UAWSNA-IoT is designed to be adaptable, allowing easy addition of sensors nodes as needed, ensuring scalability to meet growing services demands. Our protocol provides robust security measures against several attacks. It ensures anonymity, offers complete mutual authentication among all authentication entities, and maintains perfect forward secrecy during the authentication phase. Efficiency, lightweight design, and impressive performance of UAWSNA-IoT, as demonstrated in Section VI, making it an ideal choice for resource-constrained IoT devices like WSN.

REFERENCES


D. Storage Cost

In this part, we focus on the storage space cost study in the sensor node level. To simplify the analysis, the storage space required for hash functions is excluded. Table VI outlines the storage space cost at the sensor node level in the UAWSNA-IoT scheme, compared to protocols ([27], [14], [20], [25], [6], [35], and [38]). In the UAWSNA-IoT protocol, the authentication parameters stored in sensor node is solely required, necessitating only 160 bits of storage space. Table VI outlines the storage space cost at the sensor node level in the UAWSNA-IoT scheme, compared to protocols ([27], [14], [20], [25], [6], [35], and [38]). Furthermore, the storage cost at sensor node in UAWSNA-IoT is the same as one in [6], and it is less than

\[ \begin{align*}
\text{Schemes} & \quad U_k & \quad GW/N_c & \quad SN_c & \quad \text{total Cost} & \quad \text{NumMessages} \\
[27] & 512 & 544 & 192 & 2912 & 4 \\
[14] & 512 & 512 & 384 & 1408 & 5 \\
[20] & 640 & 1440 & 480 & 2560 & 4 \\
[25] & 800 & 1440 & 480 & 2720 & 4 \\
[35] & 736 & 864 & 704 & 2304 & 4 \\
[38] & 672 & 2784 & 480 & 3936 & 7 \\
\text{Our} & 672 & 1184 & 352 & 2208 & 4
\end{align*} \]


