



# A novel user authentication and key agreement scheme for heterogeneous ad hoc wireless sensor networks, based on the Internet of Things notion

Muhammed Turkanović\*, Boštjan Brumen, Marko Hölbl

University of Maribor, Faculty of Electrical Engineering and Computer Science, Maribor, Slovenia

## ARTICLE INFO

### Article history:

Received 19 June 2013

Received in revised form 23 December 2013

Accepted 31 March 2014

Available online 13 April 2014

### Keywords:

Wireless sensor network

Ad hoc

Mutual authentication

Key agreement

Smart card

Internet of Things

## ABSTRACT

The idea of the Internet of Things (IoT) notion is that everything within the global network is accessible and interconnected. As such Wireless Sensor Networks (WSN) play a vital role in such an environment, since they cover a wide application field. Such interconnection can be seen from the aspect of a remote user who can access a single desired sensor node from the WSN without the necessity of firstly connecting with a gateway node (GWN). This paper focuses on such an environment and proposes a novel user authentication and key agreement scheme for heterogeneous ad hoc wireless sensor networks. The proposed scheme enables a remote user to securely negotiate a session key with a general sensor node, using a lightweight key agreement protocol. The proposed scheme ensures mutual authentication between the user, sensor node, and the gateway node (GWN), although the GWN is never contacted by the user. The proposed scheme has been adapted to the resource-constrained architecture of the WSN, thus it uses only simple hash and XOR computations. Our proposed scheme tackles these risks and the challenges posed by the IoT, by ensuring high security and performance features.

© 2014 Elsevier B.V. All rights reserved.

## 1. Introduction

We are evermore surrounded by ubiquitous, intelligent interconnected objects (i.e. smart objects) which engage us with new applicative perspectives on our everyday lives, like RFID, smartphones, semantic web, wireless sensors, etc. This can be seen as the Internet of Things (IoT) notion, which was announced a decade ago [1]. The idea was that in the future everything (i.e. including live objects) would be accessible, sensed, and interconnected inside the global, dynamic, living structure of the Internet. Wireless sensor networks (WSN) play an important role regarding this

notion, since they cover a wide application field by collecting various environmental information.

In the early years WSNs started as simple and ambivalent research projects mainly motivated and funded by the military, e.g. DARPA (Defense Advanced Research Projects Agency) [2]. These early military-funded WSN research projects, according to their type of applications, presented a definition of the WSN as a large-scale, wireless, ad hoc, multi-hop, un-partitioned network of homogeneous, tiny, mostly immobile sensor nodes which are randomly deployed within a particular area of interest [2]. Such a definition does not apply to all of today's WSN applications, since WSN applications can also be heterogeneous, single-hop, infrastructure-based (i.e. non ad hoc), have mobile sensor nodes, etc. At that time the WSN research papers focused only on the theoretical and broad applicative uses of WSNs (e.g. military, environment, healthcare),

\* Corresponding author. Tel.: +386 40 303 874.

E-mail addresses: [muhammed.turkanovic@gmail.com](mailto:muhammed.turkanovic@gmail.com) (M. Turkanović), [boštjan.brumen@um.si](mailto:boštjan.brumen@um.si) (B. Brumen), [marko.holbl@um.si](mailto:marko.holbl@um.si) (M. Hölbl).

but the research and applicative use of WSN has significantly increased over the last few years. Today we talk about the use of WSN for traffic monitoring [3], pipeline monitoring [4], landslide detection [5], methane leak detection [6], border patrol [7], precision agriculture [8], rehabilitation applications [9], laboratory tutoring [10], asset tracking [11], real-time soccer playing monitoring [12] and many more [13–16]. A list of real-life applicative WSN projects was summed up by Romer and Mattern [2].

In the beginning it was thought that WSN would only be homogeneous, consisting only of equal sensor nodes. Today we also talk of heterogeneous WSNs since sensor networks can be built with different types of nodes, some more computationally-powerful than others (e.g. gateway nodes). In view of the IOT notion, the heterogeneity of a WSN is not the only thing rapidly adapting, hence the infrastructure has moved from mainly infrastructure-based networks, where nodes can only communicate directly with the base station, to ad hoc networks whereby nodes can also communicate directly with each other and with rest of the world.

WSNs are becoming evermore numerous and interconnected with the IOT, thereby presenting new opportunities but also challenges which need to be addressed. An example of such would be a remote user who wants to access a particular sensor node of the WSN. Such a user needs to be authorized and, if done positively, allowed to gather data from or send commands to the sensor node. Since the most important and distinct characteristic of WSNs is their resource-constrained architecture (i.e., limited computational and communicational capabilities), a lightweight security solution is required, thus urging the security design to be more prudent.

A key challenge is how to enable the establishment of a shared cryptographic key in a secure and lightweight manner, between the sensor node and the user outside the network. Mutual authentication is also needed for such a scenario, and is highly important since all parties need to be sure of the legitimacies of all the entities involved. Numerous cryptographic schemes for the security of the WSN have been proposed but very few have addressed the aforementioned scenarios, challenges, and requirements [17–27]. This was the motivation for us to develop and propose a challenge-specific cryptographic scheme, which uses a rare four-step authentication model that we believe is the most appropriate for the mentioned scenario, when a remote user wants to connect to a sensor node inside a WSN. Since our scheme uses smart cards for users to authenticate, the security architecture is also smart-card oriented. The proposed scheme is also lightweight and resource-friendly, hence it is based only on symmetric cryptography, whereby using only hash and XOR computation.

The remainder of the paper is organized as follows. Section 2 for better understanding of the topic below, provides some brief preliminaries. We present an overview of the existing work in Section 3. Our proposed mutual authentication and key agreement scheme for heterogeneous wireless sensor ad hoc networks is presented in Section 4, followed by the security and performance evaluations in Sections 5 and 6. Finally, Section 7 concludes the paper.

## 2. Preliminaries

Heterogeneous WSN consists of multiple, simple, low-cost sensor nodes that have limited computational and communicational capabilities and of at least one sink node, also called gateway node (GWN) [28]. GWN are bigger, more secure and have more computational and communicational capabilities. Since they are more powerful, they are being used to act as proxies between the sensor network and the outside world. Their functionalities are sometimes gathering and processing data from each sensor node before forwarding non-redundant information, sending commands to the sensor network, facilitating authentication schemes, etc. [29]. By facilitating with authenticated schemes, the GWN help to process mutual authentication and key agreement protocols by playing the lightweight role of a trusted third party entity [27].

According to Xue et al. [27] there are five basic authentication models for WSNs, and to our knowledge, this is the first time that authentication models for the WSN, based on the GWN, have been presented and depicted. Each of the five presented authentication models needs four messages to complete key agreement and mutual authentication. Since WSNs are resource constrained it is very important to reduce communication and transmission to the minimum, since the cost of transmitting 1 Kb of information is 3 J or equal to that of 80,000–600,000 instructions [30,31]. In four of the five models, the user initiates the key agreement scheme between him/her and the sensor node, by firstly contacting the GWN. The further key agreement communication depends on the specific model. The end result is always mutual authentication between all parties and a successful key agreement between the user and a specific sensor node.

When developing our novel mutual authentication and key agreement scheme for heterogeneous wireless sensor ad hoc networks, we focused on the resource constrained architecture of WSN, on the security requirements, and on the IOT-notional environment. According to our specifications and the mentioned requirements in the previous section for WSNs based on the IOT notion, we used the fifth authentication model (Fig. 1), described by Xue et al. [27]. This model is the only one which initiates the authentication and key agreement protocol by firstly contacting the specific sensor node. This approach has already been presented and used by Yeh et al. [23] in their secured authentication protocol for WSN using Elliptic Curve Cryptography (ECC). Using this authentication model, after successful registration, the remote user can reach a specific sensor node directly through the Internet and does not need to first connect with the GWN, thereby ensuring a more straightforward approach.

There are three types of cryptographic techniques for the WSN developed so far, i.e. symmetric, asymmetric, and hybrid [32]. Since WSNs are resource-constrained, the more lightweight techniques like the symmetric ones are the more appropriate. Even though symmetric based cryptographic schemes require less computational power, they need more memory for saving all the keys involved in the network (i.e. between all the nodes and users). Regardless to this, its efficiency still prevails in relation

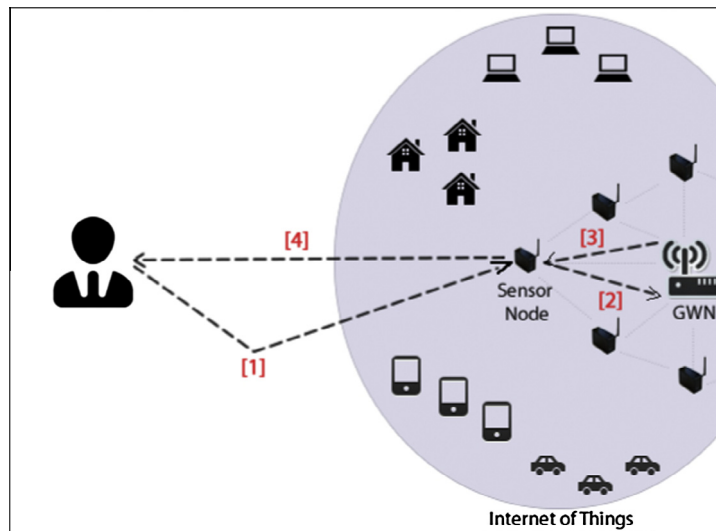


Fig. 1. Authentication model of the proposed user authentication scheme.

to the asymmetric-based cryptographic schemes, if the network size is within a certain limit. We have developed our protocol based on symmetric cryptography, using only XOR and hash computations, and have solved the problem of saving keys by delegating key-saving to more powerful and resource rich GWN nodes.

WSNs need to be secured, since they are mostly unattended and sometimes deployed within hostile environments, hence the remote user has to be allowed to access data from the network, requiring him/her to register first. Moreover, the sensor node and the GWN have to be completely sure of the legitimacy of the user, since he/she can access data from the nodes or maybe send commands to the nodes. In contrast, the user also has to be sure that the sensor node and GWN are legitimate, hence an adversary cannot impersonate the nodes and send false data. In order for a WSN scheme to be suitable, mutual authentication between all the parties involved is of crucial importance. Although mutual authentication is typically ensured by asymmetric encryption, our scheme can achieve mutual authentication by using only symmetric techniques.

Certain requirements need to be satisfied, whilst designing a user authentication scheme for WSN, like mutual authentication, but also single registration, user friendliness, low communication and computation complexities and high security (i.e. resilience against most common attacks).

### 3. Related work

This section briefly discusses some existing and related user authentication schemes for WSNs.

Numerous user authentication schemes have been proposed for the security of the WSN. Most schemes concentrate on the establishment of a cryptographic key between the user and the base station or the GWN. Asym-

metric schemes represent good foundations for the secure exchange of keys within a network but require much more computational and communicational powers. Such cryptographic architecture is unsuitable for resource constrained WSN. In 2004, Watro et al. [33] proposed a highly-cited, user authentication scheme based on asymmetric cryptography, called TinyPK. It was later shown that this scheme was vulnerable to several attacks (e.g. man-in-the-middle attack) and therefore unsuitable for implementation [24]. Additional asymmetric user authentication schemes based on the Diffie and Helman [34] key agreement, which provided mutual authentication were presented by Xu et al. [35] and Song [36]. The problem with these schemes was their memory overhead, since each node had to store all the public keys of other nodes and users. Another asymmetric ECC-based user authentication scheme was presented by Yeh et al. [23] but had the same problems as the aforementioned asymmetric schemes.

In 2006, Wong et al. [37] proposed a user authentication scheme for WSN based on symmetric encryption. They used only hash based computation, thus providing a lightweight architecture. However, it was later shown that this scheme was also vulnerable to multiple attacks (e.g., many logged-in users with the same login-id attack, stolen verifier attack, etc.). Das [24] improved the security of Wong et al.'s scheme and proposed an efficient password-based user authentication with the help of the GWN, that used temporal credentials (i.e. timestamps) for verification. The drawbacks of this scheme were that it did not ensure mutual authentication and user anonymity. It also did not provide for a key agreement and was vulnerable to several attacks (e.g. denial-of-service attack, node capture attack, etc.). Some improvements to Das's scheme were later presented by [17,18,38]. Although the presented schemes ensured some security enhancements, they incompletely removed the security flaws. In 2010, Khan and Alghathbar [21] also presented an improvement of Das's scheme. They solved the mutual-authentication and

unsecured password problems by introducing pre-shared keys and hashed passwords. However Vaidya et al. [19], later showed that Khan and Alghathbar's scheme was also vulnerable to several attacks and proposed an improved version of the scheme. Another improvement of Das's scheme came from Chen and Shih [22]. Their scheme provided mutual authentication between all parties involved in the key agreement process but was later shown as being vulnerable to several attacks (e.g. replay attack, forgery attack, etc.).

More recently Das et al. [26] and Xue et al. [27] proposed two user authentication and key agreement schemes for WSN using smart cards. They are both designed to support a remote user to effectively and securely connect to the nodes of a WSN. Both schemes provide security features like mutual authentication, password protection, key agreement, resilience against several attacks, and a dynamic node addition phase. Both schemes use hash and XOR computations and are therefore lightweight and highly appropriate for WSN. It was later shown by Xu and Wang [39], and Turkanović and Hölbl [40] that Das et al.'s scheme has imperfections and is infeasible for implementations. Both have proposed an improved version of Das et al.'s scheme. Similar to Das et al.'s scheme, it was also shown that Xue et al.'s scheme has some vulnerabilities, which were presented and corrected by Li et al. [41] and Turkanović and Hölbl [42]. Although Das et al.'s scheme was developed for hierarchical WSNs (i.e. part of the heterogeneous group of WSN) and the key agreement runs between the user, base-station (BS), and cluster head (CH), and Xue et al.'s key agreement runs between the user, GWN, and the sensor node, both use the same authentication models as described in [27, Fig. 1d]. Xue et al. argue

that such a model is efficient because it runs the last two communications, acknowledgement for the BS or GWN and the user, simultaneously. However since both communications have to be run it is insignificant regarding efficiency hence, as mentioned before in the previous section, communication is highly energy-demanding, resulting only in faster protocol-ending.

#### 4. Proposed scheme

This section presents our new lightweight mutual authentication scheme for heterogeneous ad hoc wireless sensor networks. The scheme is designed to work in correlation with the IOT notion. We have developed this scheme based on a rare four-step authentication model (Fig. 1), where a remote user initiates the authentication phase by firstly connecting with a sensor node of interest. The scheme ensures important features like mutual authentication, password security, single registration, key agreement, high security, and low computational costs.



Prior to describing this newly proposed scheme, we have firstly summarized in Table 1, the notations used within the protocol. We have then described all the phases relating to our scheme in the subsections, i.e. pre-deployment phase, registration phase, login phase, authentication phase, and the dynamic node addition phase.

##### 4.1. Pre-deployment phase

Heterogeneous WSN consist of two or more different node types. The tiny low-cost, resource constrained sensor nodes, counting in tens or hundreds, and a proportional

**Table 1**

Notations used in the paper.

	Sent via SECURE network
	Sent via PUBLIC/UNSECURE network
$U_i$	User
$SC$	Smart card
$S_j$	Sensor node
$X_{GWN}$	Secure password known only to the GWN
$X_{GWN-U_i}$	Secure password shared with the user. Initiated during user's registration phase
$X_{GWN-S_j}$	Secure password shared with sensor node. Initiated in the pre-deployment phase
$ID_i$	User's identity
$PW_i$	User's password
$r_i, r_j$	User's and sensor node's secret random nonces used for computing user's masked password and sensor node's masked identity
$MP_i$	User's masked password
$MI_i$	User's masked identity
$SID_j$	Regular sensor node's identity
$MN_j$	Masked nonce of the sensor node
$RMP_j$	Masked password of sensor node XOR-ed with the masked nonce
$T_x$	Current timestamps
$\Delta T$	Time interval for the allowed transmission delay
$N_i$	Computed by the user. Sent to GWN over sensor node. Used by GWN to check if the user is a legitimate user
$Z_i$	Computed by the user. Sent to sensor node. Used to mask the user's session-key part
$A_j$	Used to mask $x_j$ of sensor so the GWN can use it to check whether the sensor is a legitimate one
$F_{ij}$	Computed by the GWN and used by the sensor node to get the $K_i$ from $Z_i$
$H_j$	Used by the sensor to check whether GWN is legitimate and not an impersonator
$S_i$	Computed by the GWN and sent to the user over the sensor node. It is used by the user to check if the sensor node and GWN are legitimate
$R_{ij}$	Computed by the sensor node and sent to the user. Used to mask the sensor node's session-key part
$SK$	Separately computed session key with private information of both user and sensor node
$\parallel, \oplus, h()$	Concatenation, XOR and hash operation

number of more powerful GWNs. Since WSNs are only used within a particular field of interest, they need to be firstly deployed. The pre-deployment phase, which is the setup phase, runs offline and is done by a network administrator using a setup server.

During the pre-deployment phase, each regular sensor node  $\{S_j | 1 \leq j \leq m\}$  is predefined with its identity  $SID_j$  and a randomly-generated secure password-key  $X_{GWN-S_j}$ , which is shared with the GWN and stored in the memory of the node. The value  $m$  represents the number of sensor nodes within the network that will be deployed in the field of interest. Since sensor nodes are resource-constrained they have limited memory, but in the case of our proposed scheme they only need to store a small amount of data, as described above. Additionally the GWN is predefined with a randomly-generated highly secure password-key  $X_{GWN}$  which is known only to the GWN and is secretly stored in the memory of the GWN. Furthermore, the GWN stores the corresponding secret password-key shared with each sensor node  $\{X_{GWN-S_j} | 1 \leq j \leq m\}$ . As mentioned earlier, GWNs are more resource rich having much more memory space, and hence can store all the shared password-keys of each of the sensor node  $\{S_j | 1 \leq j \leq m\}$ .

#### 4.2. Registration phase

Two separate registration phases are needed after the deployment. The first is a registration phase between the GWN and a corresponding regular sensor node  $S_j$ , and is run after the deployment. The second registration phase is between a user  $U_i$  and the GWN, and is initiated by the user on demand. The procedure for both registration phases is depicted in Fig. 2.

We now provide a detailed description of both phases. Firstly we introduce the  $U_i$  – GWN registration phase. The aim of this phase is for the user to register with the WSN. The registration is done by using a smart card, which is personalized by the GWN at the end of the phase. When the user is registered with the WSN, he/she can then

connect with each regular sensor node on demand and run the authentication phase, whereby the result is a shared session key used for secure communication between them. Furthermore, after successfully registering, the user can mutually authenticate with the WSN and the sensor nodes within the network. The following steps are required for the user  $U_i$  to successfully register with the GWN and with WSN:

- Step 1: The user  $U_i$  selects his/her identity  $ID_i$ , password  $PW_i$  and chooses a random number  $r_i$ .
- Step 2:  $U_i$  computes a masked password  $MP_i = h(r_i \parallel PW_i)$  using the random number  $r_i$  and his/her password  $PW_i$ .
- Step 4:  $U_i$  computes a masked identity  $MI_i = h(r_i \parallel ID_i)$  using the random number  $r_i$  and his/her  $ID_i$ .
- Step 4: The masked password  $MP_i$  and masked identity  $MI_i$  are then sent to the GWN via a secure channel [23].
- Step 5: After receiving  $MI_i$  and  $MP_i$  from  $U_i$ , the GWN computes  $f_i = h(MI_i \parallel X_{GWN})$ , using his/her secret password-key  $X_{GWN}$ .
- Step 6: The GWN then randomly chooses a secret password-key  $X_{GWN-U_i}$  for the  $U_i$  and computes  $x_i = h(MP_i \parallel X_{GWN-U_i})$  using  $U_i$ 's masked password  $MP_i$ .
- Step 7: After having computed  $f_i$  and  $x_i$ , the GWN computes  $e_i = f_i \oplus x_i$ .
- Step 8: The GWN personalizes  $U_i$ 's smart card (SC) with the following parameters: (i)  $MI_i$ , (ii)  $e_i$ , (iii)  $f_i$  and (iv)  $X_{GWN-U_i}$ .
- Step 9: The GWN stores user's masked identity  $MI_i$  and secret shared password  $X_{GWN-U_i}$  to its memory.
- Step 10: At the end, the  $U_i$ , as the only one knowing the secret random number  $r_i$ , stores it into the smart card.

Secondly, we provide the details of the  $S_j$  – GWN registration phase. The aim of this phase is to register all the deployed regular sensor nodes with the GWN, and with the network. After successful registration, the GWN and the regular sensor node can mutually authenticate each

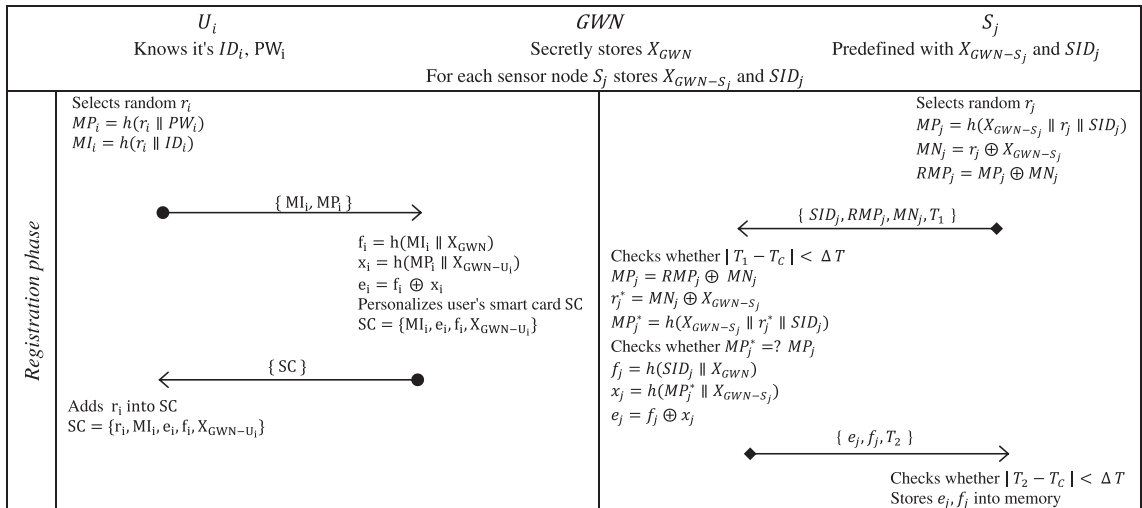


Fig. 2. Registration phase of the proposed scheme.



other and thus be certain of each other's legitimacy. The  $S_j$ –GWN registration phase is also necessary for the dynamic node addition phase. Since the network can be dynamic and grow, it should be able to add additional nodes on-demand. Moreover, in a case where a node fails to operate (e.g. battery drain, hardware failure, etc.), replacement should also be possible. The following steps are required for a sensor node  $S_j$  to successfully register with the GWN and with the WSN:

- Step 1: The sensor node  $S_j$  chooses a random number  $r_j$ .
- Step 2: Using its identity  $SID_j$ , the stored password key  $X_{GWN-S_j}$  and the chosen random number  $r_j$ , the  $S_j$  computes  $MP_j = h(X_{GWN-S_j} || r_j || SID_j)$ .
- Step 3: Next the  $S_j$  computes the masked nonce  $MN_j = r_j \oplus X_{GWN-S_j}$ , by XOR-ing the stored password key  $X_{GWN-S_j}$  and the chosen random number  $r_j$ .
- Step 4: Using the previously computed  $MP_j$  and  $MN_j$ , the  $S_j$  computes  $RMP_j = MP_j \oplus MN_j$ .
- Step 5: After computing all the necessary values, the  $S_j$  sends  $SID_j$ ,  $RMP_j$ ,  $MN_j$  and  $T_1$  to the GWN via an unsecure channel, whereby  $T_1$  represents the current timestamp value of the  $S_j$  and is used to prevent a replay attack.
- Step 6: After receiving the values from the  $S_j$ , the GWN firstly checks  $|T_1 - T_C| < \Delta T$ .  $T_1$  is the received timestamp from the  $S_j$  and  $T_C$  is the current timestamp of the GWN. By computing the equation, the GWN checks whether the transmission delay was within the allowed time interval  $\Delta T$ . If the equation does not hold, the GWN aborts any further operation and sends a rejection message to the  $S_j$ .
- Step 7: If the equation in step 6 holds, the GWN computes the original version of the  $MP_j = RMP_j \oplus MN_j$ , by XOR-ing the received values  $RMP_j$  and  $MN_j$ .
- Step 8: The GWN chooses the appropriate shared password-key  $X_{GWN-S_j}$  according to the received identity  $SID_j$  of the sensor node  $S_j$ . Having found the shared password-key  $X_{GWN-S_j}$ , the GWN computes its own version of the nonce  $r_j^* = MN_j \oplus X_{GWN-S_j}$ , using the received  $MN_j$  and found shared password key  $X_{GWN-S_j}$ .
- Step 9: Using the newly computed version of the nonce  $r_j^*$ , the GWN further computes its own version of the masked password  $MP_j^* = h(X_{GWN-S_j} || r_j^* || SID_j)$ .
- Step 10: GWN now checks whether the original  $MP_j$  and the computed  $MP_j^*$  are equal. If they are equal, the GWN is certain of  $S_j$ 's legitimacy. Otherwise GWN aborts any further operation and sends a rejection message to the  $S_j$ .
- Step 11: Using its secret password-key  $X_{GWN}$  and  $S_j$ 's identity  $SID_j$ , the GWN computes  $f_j = h(SID_j || X_{GWN})$ .
- Step 12: Using the shared password-key  $X_{GWN-S_j}$  and the early computed  $MP_j^*$ , the GWN further computes  $x_j = h(MP_j^* || X_{GWN-S_j})$ .
- Step 13: The GWN finally computes  $e_j = f_j \oplus x_j$  and sends  $e_j, f_j$  and  $T_2$  via an unsecure channel to the  $S_j$ , whereby  $T_2$  is the GWN's current timestamp.
- Step 14: After receiving the values from the GWN, the  $S_j$  firstly checks  $|T_2 - T_C| < \Delta T$ . If the equation does not hold the  $S_j$  sends a rejection message to GWN and aborts any further action.

- Step 15: If the above equation holds, the  $S_j$  accepts the message and stores values  $e_j$  and  $f_j$  into the memory, thereby successfully ending the registration phase.

#### 4.3. Login phase

After the registration phase, the user  $U_i$  can connect to a desired sensor node of the network by initiating an authentication phase. In order for the authentication phase to be initiated, the user has to first login. As aforementioned, our scheme uses a smart card (SC) for the user to register and authenticate. Any other security system for securely saving a user's data could be used instead. An illustration of the login phase is presented in Fig. 3. Detailed steps of the phase are provided below as follows:

- Step 1: The user  $U_i$  inserts his/her SC into a terminal and inputs his/her password  $PW_i^*$ .
- Step 2: The SC uses the inputted password  $PW_i^*$  and the stored secret random number  $r_i$  to compute  $MP_i^* = h(r_i || PW_i^*)$ .
- Step 3: Using the newly computed  $MP_i^*$  and the stored shared password-key  $X_{GWN-U_i}$ , the SC computes its own version of  $x_i^* = h(MP_i^* || X_{GWN-U_i})$ .
- Step 4: To compute the original  $x_i = f_i \oplus e_i$ , the SC uses the stored  $e_i$  and  $f_i$ .
- Step 5: The SC then checks whether the original  $x_i$  and the computed one  $x_i^*$  are equal. If they are unequal, the user did not input the correct password and the login process is aborted.
- Step 6: If the password was correct, the SC further computes  $N_i = h(x_i || X_{GWN-U_i} || T_1)$ , using the stored values  $x_i$  and  $X_{GWN-U_i}$ , and the current timestamp  $T_1$ .
- Step 7: Furthermore, the SC randomly chooses a nonce  $K_i$ , and computes  $Z_i = K_i \oplus f_i$ , by using the stored value  $f_i$ . The nonce  $K_i$  will be used for constructing a shared session key with the chosen sensor node. The value  $Z_i$  will be sent to the  $S_j$  and is used to mask the  $U_i$ 's session-key part.
- Step 8: Finally, the user  $U_i$  chooses a  $S_j$  and sends an authentication message via an unsecure channel to the node with the following values  $Mi$ ,  $e_i$ ,  $Z_i$ ,  $N_i$  and  $T_1$ .

#### 4.4. Authentication phase

We now provide the details of the authentication phase. This phase is initiated by the user with an authentication message at the end of a successfully executed login phase. Since our scheme is developed to optimally adapt to the IOT notion, the user sends the authentication message to a desired sensor node of the network and not the GWN. The aim of this phase is to negotiate a secret session key between the user and the sensor node in a way that both will individually contribute to the session key with a secretly chosen nonce. After successfully negotiating the session key, they can use it to securely communicate in any encrypted matter. In order to achieve the secure session key negotiation, a lightweight key agreement method is proposed which involves mutual authentication between all parties (i.e. user, GWN and sensor node). The mutual authentication at this part is highly necessary because a

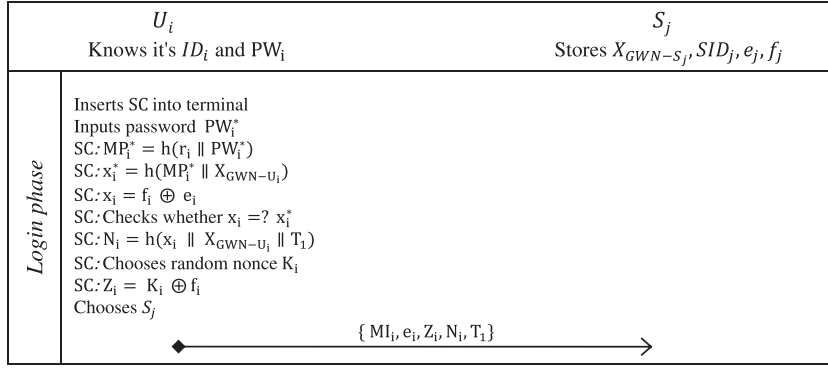


Fig. 3. Login phase of the proposed scheme.



Fig. 4. Authentication phase of the proposed scheme.

remote user can directly connect with a sensor node of the network, thus needing to be authenticated by the sensor node with the help of the GWN. An illustration of the authentication phase is depicted in Fig. 4. The steps for the authentication phase are provided in detail as follows:

- Step 1: Upon receipt of the authentication message  $\{MI_i, e_i, Z_i, N_i, T_1\}$  from the user  $U_i$ , the sensor node  $S_j$  firstly checks whether  $|T_1 - T_c| < \Delta T$ , using its current timestamp  $T_c$  and the received one  $T_1$ . If the verification

does not hold, the  $S_j$  aborts any further action and sends a rejection message to  $U_i$ . If the verification holds, step 2 follows.

- Step 2: Using the stored values  $e_j$  and  $f_j$ ,  $S_j$  computes  $x_j = e_j \oplus f_j$ .
- Step 3: After computing  $x_j$ ,  $S_j$  uses it to compute  $A_j = h(X_{GWN-S_j} \parallel T_1 \parallel T_2) \oplus x_j$ , using the stored shared password-key  $X_{GWN-S_j}$ , the timestamp of user  $T_1$  and its current timestamp  $T_2$ .  $A_j$  is used for further mutual authentication between the  $S_j$  and the GWN.

- Step 4:  $S_j$  connects with the GWN and sends a message including the values  $MI_i, e_i, N_i, T_1, T_2, SID_j, e_j$  and  $A_j$ . Here the GWN will be used to authenticate the user and send the  $U_i$ 's legitimacy status to the  $S_j$ . Therefore  $S_j$  simply forwards the received values  $MI_i, e_i, N_i$  and  $T_1$ , to the GWN, thus allowing GWN to authenticate the  $U_i$ . Additionally, the message contains  $S_j$ 's identity  $SID_j$ , timestamp  $T_2$  and  $e_j$ , which is in contrast used by the GWN for authenticating the  $S_j$ .
- Step 5: After receiving the message from the  $S_j$ , the GWN firstly checks whether  $|T_2 - T_c| < \Delta T$ , using the received timestamp of the  $S_j$  and its current timestamp. If the verification does not hold, the GWN aborts any further action and sends a rejection message to  $S_j$ .
- Step 6: If the above verification holds, the GWN firstly computes its own version of  $f_j^* = h(SID_j || X_{GWN})$ , using its secret password-key  $X_{GWN}$  and  $S_j$ 's identity  $SID_j$ .
- Step 7: Using the computed value  $f_j^*$  and the received value  $e_j$ , the GWN further computes  $x_j^* = e_j \oplus f_j^*$ .
- Step 8: Furthermore, the GWN computes the original  $x_j = A_j \oplus h(X_{GWN-S_j} || T_1 || T_2)$ , using the received values  $A_j, T_1$  and  $T_2$  and the shared password-key  $X_{GWN-S_j}$  according to the identity  $SID_j$  of the  $S_j$ .
- Step 9: After having its own version of  $x_j^*$  and the original one  $x_j$ , the GWN verifies if they are the same, thereby authenticating the sensor node  $S_j$ . If the verification does not hold, the GWN aborts any further action since  $S_j$  proves to be an illegitimate one. Furthermore, the GWN sends a rejection message to the  $S_j$ .
- Step 10: If the verification holds, the GWN has successfully authenticated the  $S_j$  and starts with authenticating the  $U_i$ . Firstly, the GWN computes its own version of  $f_i^* = h(MI_i || X_{GWN})$ , using its own secret password-key  $X_{GWN}$  and user's masked identity  $MI_i$  which is forwarded by the  $S_j$ .
- Step 11: Using the newly computed value  $f_i^*$  and the received value  $e_i$ , the GWN computes its own version of  $x_i^* = e_i \oplus f_i^*$ . The value  $e_i$  is  $U_i$ 's value forwarded by the  $S_j$ .
- Step 12: Before authenticating  $U_i$ , the GWN has to compute  $Q_i = h(x_i^* || X_{GWN-U_i} || T_1)$ , using  $U_i$ 's timestamp  $T_1$ , the newly computed value  $x_i^*$  and the shared password-key  $X_{GWN-U_i}$ , which is chosen according to the received  $MI_i$ . The values  $T_1$  and  $MI_i$  are the forwarded values from the  $S_j$  which were received by the  $U_i$ .
- Step 13: After having computed the value  $Q_i$ , the GWN can use it to check whether it equals with the received  $N_i$ , thus check if  $U_i$ , who initiated the authentication phase, is legitimate. If the verification does not hold, the GWN aborts any further actions and sends a rejection message to the  $S_j$ , stating that the  $U_i$  is not registered and illegitimate. In this case, the  $S_j$  also aborts any further actions and sends a reject message to the  $U_i$ . If the verification holds, the GWN successfully authenticated the  $U_i$  and continues with the authentication phase as follows.
- Step 14: Using the shared password-key  $X_{GWN-S_j}$  and the earlier computed values  $f_i^*$  and  $f_j^*$ , the GWN computes  $F_{ij} = f_i^* \oplus h(f_j^* || X_{GWN-S_j})$ . The value  $F_{ij}$  will further be used by the  $S_j$  to extract  $U_i$ 's nonce  $K_i$  from  $Z_i$ . The value  $Z_i$  was sent by the  $U_i$  as part of the authentication message.
- Step 15: The GWN further computes  $H_j = h(f_j^* || X_{GWN-S_j} || T_1 || T_2 || T_3)$ , using the earlier computed value  $f_j^*$ , the shared password-key  $X_{GWN-S_j}$ ,  $U_i$ 's timestamp  $T_1$ ,  $S_j$ 's timestamp  $T_2$  and its own timestamp  $T_3$ . The value  $H_j$  will be further used by the  $S_j$  to check the legitimacy of the GWN and prevent any impersonation attacks.
- Step 16: In addition, the GWN computes  $S_i = h(Q_i || T_1 || T_2 || T_3)$ , using the aforementioned timestamps and the earlier computed value  $Q_i$ . The value  $S_i$  will be sent to  $U_i$ , whereby  $U_i$  can used to verify GWN's and  $S_j$ 's legitimacy.
- Step 17: After having all the necessary values, the GWN sends a message via an unsecure channel to the  $S_j$  with the following values  $F_{ij}, H_j, S_i, T_1, T_2$  and  $T_3$ . Following this step, the GWN has finished its part within the authentication phase.
- Step 18: Upon receipt of the message, the  $S_j$ , firstly verifies whether  $|T_3 - T_c| < \Delta T$ , using GWN's received timestamp  $T_3$  and its own current timestamp  $T_c$ . If the verification does not hold, the  $S_j$  aborts any further action and sends a rejection message to the GWN and the  $U_i$ , thereby preventing replay attack.
- Step 19: If the verification holds, the  $S_j$  computes  $h(f_j || X_{GWN-S_j} || T_1 || T_2 || T_3)$ , using its own version of  $f_j$ , the shared password-key  $X_{GWN-S_j}$  and all three received timestamps  $T_1, T_2$  and  $T_3$ . It then checks whether the newly computed value equals the received  $H_j$ . If the verification holds, the  $S_j$  has authenticated the GWN and proved its legitimacy. In contrast, the  $S_j$  aborts any further action and sends a rejection message to the GWN and to the  $U_i$ , since the GWN has been proved not to be a legitimate one.
- Step 20: After authenticating the GWN, the  $S_j$  starts to extract and build the shared session key between itself and the  $U_i$ . Firstly, it computes its own version of  $f_i^* = F_{ij} \oplus h(f_j || X_{GWN-S_j})$ , using the received value  $F_{ij}$  and stored values  $f_j$  and  $X_{GWN-S_j}$ .
- Step 21: After having  $f_i^*$ , the  $S_j$  computes  $U_i$ 's secret nonce  $K_i = Z_i \oplus f_i^*$ , using the previously received value  $Z_i$ .
- Step 22: After having computed one part of the secret shared session key, the  $S_j$  chooses a random nonce  $K_j$  as the second part of the session key and computes the final version as follows  $SK = h(K_i \oplus K_j)$ .
- Step 24: Finally, the  $S_j$  computes  $R_{ij} = h(f_i^* || SID_j || T_1 || T_2 || T_3 || T_4) \oplus K_j$ , using its own identity  $SID_j$ , the previously computed version of  $f_i^*$ , the previously generated random nonce  $K_j$ , its own timestamp  $T_4$  and all other previous timestamps  $T_1, T_2$  and  $T_3$ . The value  $R_{ij}$  is used to mask the secret random nonce  $K_j$ , hence the second part of the session key.
- Step 25: At the end, the  $S_j$  sends a message to the  $U_i$  via an unsecure channel with the following values  $S_i, R_{ij}, T_1, T_2, T_3$  and  $T_4$ . Following this step, the  $S_j$  has successfully finished its part in the authentication phase.
- Step 26: After receiving the message from the  $S_j$ , the  $U_i$  firstly checks whether  $|T_4 - T_c| < \Delta T$ , using its own current timestamp  $T_c$  and  $S_j$ 's received timestamp  $T_4$ . If the verification does not hold, the user aborts the authentication phase and sends a rejection message to the  $S_j$ , thereby preventing any replay attack.



- Step 27: If the verification holds, the  $U_i$  then computes  $h(h((e_i \oplus f_i) \| T_1) \| T_2 \| T_3)$ , using the received time-stamps  $T_1, T_2$  and  $T_3$ , and values  $e_i$  and  $f_i$ , which are stored in the SC. It then checks whether the newly computed value equals the received  $S_i$ . If the verification holds, then  $U_i$  can be sure of GWN's and the  $S_j$ 's legitimacy and continue with constructing the secretly shared session key. If the verification does not hold, the  $U_i$  aborts the authentication phase and sends a rejection message to the  $S_j$ , since the GWN or the  $S_j$  are illegitimate.
- Step 28: For constructing a session key, the  $U_i$  needs the second part of it, which is the secretly generated random nonce of the  $S_j$ . The computation of it is done as follows:  $K_j = R_{ij} \oplus h(f_i \| SID_j \| T_1 \| T_2 \| T_3 \| T_4)$ .
- Step 29: Finally, after having  $K_j$ , the  $U_i$  can compute the final session key  $SK = h(K_i \oplus K_j)$  and thus successfully end the authentication phase.

#### 4.5. Password-change phase

We now provide the details of the password change-phase. A password-change option is desirable as we are proposing a user-authentication scheme. In our proposed scheme, the user has the ability to freely choose and if desired change his/her password. Our proposed scheme is designed to work with a smart card, whereby a user in order to successfully authenticate, a login to the smart card is required, using a password. Since our proposed login-phase is done locally and offline, meaning the GWN does not need to be involved, the password-change phase is done likewise. An illustration of the password-change phase is depicted in Fig. 5. The following steps are required for a user to change her password:

- Step 1: The user  $U_i$  inserts his/her SC into a terminal and inputs his/her old password  $PW_i^{OLD}$ .
- Step 2: The SC uses the inputted password  $PW_i^{OLD}$  and the stored secret random number  $r_i$  to compute  $MP_i^* = h(r_i \| PW_i^{OLD})$ .
- Step 3: Using the newly computed  $MP_i^*$  and the stored shared password-key  $X_{GWN-U_i}$ , the SC computes its own version of  $x_i^* = h(MP_i^* \| X_{GWN-U_i})$ .

$U_i$ Knows its $ID_i, PW_i$	
Password change phase	Inserts SC into terminal Inputs old/current password $PW_i^{OLD}$ SC: $MP_i^* = h(r_i \  PW_i^{OLD})$ SC: $x_i^* = h(MP_i^* \  X_{GWN-U_i})$ SC: $x_i = f_i \oplus e_i$ SC: Checks whether $x_i = ? x_i^*$ Inputs new password $PW_i^{NEW}$ SC: $MP_i^{**} = h(r_i \  PW_i^{NEW})$ SC: $x_i^{NEW} = h(MP_i^{**} \  X_{GWN-U_i})$ SC: $e_i^{NEW} = f_i \oplus x_i^{NEW}$ SC: Replaces old $e_i$ with the new $e_i^{NEW}$

Fig. 5. Password-change phase of the proposed scheme.

- Step 4: The SC uses the stored  $e_i$  and  $f_i$  in order to compute the original  $x_i = f_i \oplus e_i$ .
- Step 5: The SC then checks whether the original  $x_i$  and the computed one  $x_i^*$  are equal. If they are unequal, the user did not input the correct password and the password-change process is aborted.
- Step 6: If the password was correct, the SC invites the  $U_i$  to input his/her new password.
- Step 7: Having the new password  $PW_i^{NEW}$ , the SC then computes the new masked password  $MP_i^{**} = h(r_i \| PW_i^{NEW})$ .
- Step 8: The SC computes the new version of  $x_i^{NEW} = h(MP_i^{**} \| X_{GWN-U_i})$  by using the newly computed  $MP_i^{**}$  and the stored shared password-key  $X_{GWN-U_i}$ .
- Step 9: The SC can compute the new  $e_i^{NEW} = f_i \oplus x_i^{NEW}$  after having the new value  $x_i^{NEW}$ .
- Step 10: Finally, the SC replaces the old  $e_i$  with the newly computed one  $e_i^{NEW}$ , thus successfully ending the password-change phase.

#### 4.6. Dynamic node addition phase

WSNs can be generic, meaning that the number of nodes can vary throughout the lifetime of the network. Since nodes are resource-constrained and low-cost, they can eventually stop working because of energy drainage or because of hardware failure. Moreover, they can be physically destroyed or stolen by an adversary. In such cases, nodes have to be replaced, since black spots emerge within the topology of the network, causing the network to lose its optimality. Moreover, sometimes the network administrators have to add additional ones to the network, regardless of the activeness of all the sensor nodes. Regardless of the cause, node addition is highly desirable and has to be predicted within the security protocols of the WSNs.

Our proposed scheme enables nodes to be simply and dynamically added to the network without causing any changes in the established security states of the current entities within the network. Moreover, node addition is only limited by the memory of the GWN, since the GWN stores password-keys for each node in the network  $X_{GWN-S_k} | \{1 \leq k \leq m\}$ . Here the value  $m$  represents the maximum number of possible shared password-keys.

A registration phase between the node and the GWN is needed for a node to be additionally added to the network. Like all other nodes in the pre-deployment phase, the new node  $S_k$  has to be predefined with an identity  $SID_k$  and a shared password-key  $X_{GWN-S_k}$ . This is done by the same administrator and the setup server as during the pre-deployment phase. The administrator also has to add the new password-key and identity of the new node to the memory of the GWN. After deployment of the new node  $S_k$ , the  $GWN-S_k$  registration phase is initiated (Section 4.2). Since the  $GWN-S_k$  registration phase is done privately between the GWN and the  $S_k$ , the user  $U_i$  does not have to be involved in the process, and hence, is not required for changing anything. After the successfully executed  $GWN-S_k$  registration phase, the  $U_i$  can already initiate the authentication phase with the newly added  $S_k$ .

## 5. Security evaluation of the proposed scheme

In this section we provide the details of the security analysis of the proposed scheme. The security of authentication protocols is highly important but also hard to achieve. We show that our scheme ensures and provides a secure key agreement, mutual authentication, password protection, and is resilient against replay attacks, man-in-the-middle attacks, impersonation attacks, privileged insider attacks, stolen-verifier attacks, stolen smart card attacks, smart card breach attacks, many logged-in users with the same login-id attacks, password change attacks, GWN bypassing attacks, and denial-of-service attacks.

### 5.1. Key agreement

An encryption needs to be set using a secretly shared session key between two parties in order to securely communicate over an unsecure-open network. A session key is negotiated using a key agreement protocol, whereby all parties involved independently and individually form and influence an end key. Usually key agreement protocols are based on asymmetric cryptography or public-key agreement (e.g. Diffie and Hellman [34]), i.e. on the public-private key pairs. Such key agreement protocols are not desired since WSN are resource constrained.

This paper, presents a secure and lightweight key agreement protocol between the sensor node and the user, using only hash and XOR computations, thus ensuring better performance within the resource constrained environment. The key agreement is a result of the authentication phase which is initiated by the user whilst connecting with a desired sensor node. Both parties (i.e. user and sensor node) at the end of the key agreement protocol compute the same session key  $SK = h(K_i \oplus K_j)$  in step 7 of the login phase, whereby individually contributing to it. Hence the user with the nonce  $K_i$  and the sensor node with the nonce  $K_j$ . The session key is negotiated securely via an unsecure channel, therefore the nonces are all always masked. The user  $U_i$  masks his/her nonce  $K_i$  by computing  $Z_i = K_i \oplus f_i$  before sending it to the sensor node  $S_j$  as part of the authentication message. In contrast, the sensor node  $S_j$  masks its nonce  $K_j$  by computing  $R_{ij} = h(f_i^* || SID_j || T_1 || T_2 || T_3 || T_4) \oplus K_j$  in step 24 of the authentication phase, before sending it to the user  $U_i$  as part of a confirmation message. Therefore only the legitimate user  $U_i$  and sensor node  $S_j$  can extract the nonces, proving the key agreement as secure. Moreover, by providing mutual authentication between all parties involved in the key agreement, we have ensured resilience against man-in-the-middle attack, replay attack, and impersonation attack.

### 5.2. Mutual authentication

In a user authentication scheme, mutual authentication is of high importance. When a user tries to sign into a network or a server, he/she needs to be sure of the server's legitimacy, since an adversary could masquerade as a legitimate server by executing a man-in-the-middle attack. In contrast, a server also needs to be sure of the user's legitimacy, since it will allow the user to access

the network and collect or insert data from or into it. Mutual authentication is a process where both parties authenticate each other and are sure of each other's identities. Since in the proposed scheme, the GWN plays a role as a trusted third party and the user directly connects with the sensor node, all three need to mutually authenticate each other.

When the user  $U_i$  starts the authentication phase by connecting with the sensor node  $S_j$ , the  $S_j$  has to delegate the authentication of the user  $U_i$  to the GWN. Whilst the GWN receives the authentication message from the  $S_j$ , it first needs to be sure that the  $S_j$  is legitimate, thereby it verifies its legitimacy by executing step 9 of the authentication phase. Only after verifying  $S_j$ 's legitimacy, does the GWN start authenticating the user  $U_i$ . In order to authenticate  $U_i$ , the GWN verifies  $U_i$ 's legitimacy by executing step 13 of the authentication phase. When both  $S_j$  and  $U_i$  are proven to be legitimate and authenticated, the GWN sends a confirmation message back to  $S_j$ . After receiving the confirmation message from the GWN, the  $S_j$  has to verify whether the GWN was really a legitimate one and thus executes step 19 of the authentication phase, whereby ending the mutual authentication between itself and the GWN. Finally, the  $S_j$  sends a confirmation message to the  $U_i$ . Before computing the session key, the  $U_i$  first has to check whether the confirmation message and thereby  $S_j$  and GWN are legitimate. In order to check their legimitacies, the  $U_i$  executes step 27 of the authentication phase and if successfully proven, the mutual authentication has been executed and done properly.

### 5.3. Password protection

User authentication is usually done via an unsecure open channel. If the passwords were to be sent unmasked and unprotected they could be intercepted and stolen by an adversary. The adversary could use the gained password to impersonate a legitimate user. Moreover, since users sometimes use the same passwords for multiple platforms and accounts, the adversary could login and access a user's private data within other networks. In order to prevent such scenarios, in our proposed schemes we protect all the password occurrences sent via unsecure and open networks, thereby saving the scheme from eavesdropping and passive information gathering. Moreover, the user's password is also not saved by the smart card, thus preventing the stealing of a password using a smart card breach attack (described later on).

### 5.4. User anonymity

Since authentication of a user is done via an unsecure open channel, our scheme ensures user anonymity using the masked identity  $MI_i$  as identification for the user, thus disabling any private information leakage if an adversary would eavesdrop the communication. Since the masked identity  $MI_i = h(r_i || ID_i)$  is computed by hashing the concatenated identity  $ID_i$  and the secret random number  $r_i$ , it is computationally infeasible for an adversary to extract the user's real identity. Moreover, even if a smart card breach attack would occur (Section 5.7) an adversary still would

not be able to infer any user's private information because the identity of a user is not stored in it, but rather the masked identity.

### 5.5. Replay attack

A replay attack is when an adversary intercepts a user's login-authentication message and simply transmits the same to the server, thus trying to impersonate the legitimate user. In order to overcome this threat we have introduced timestamps which are verified at every transmission. The proposed scheme is furthermore resilient against replay attacks, because an adversary has to know  $f_i$  and  $SID_j$  if he/she wants to successfully compute the session key and impersonate a legitimate user. If an adversary intercepts  $U_i$ 's authentication message  $\{MI_i, e_i, Z_i, N_i, T_1\}$ , he/she still needs to execute step 28 of the authentication phase to successfully compute the session key which is impossible, as described earlier.

### 5.6. Privileged insider and stolen-verifier attack

A privileged insider attack is when a malicious user with enough privileges (e.g. system administrator) gathers a private user's data and tries to impersonate that user in other networks and systems where the user is registered. This is possible due the fact that many users use the same identity and password across multiple networks and systems because of the inconvenience of remembering multiple long passwords. Since in our case, the GWN plays the role of a trusted third party, a malicious user would need to have privileged access to the GWN to execute the attack. Moreover, if a simple adversary, who is not a privileged user, were to gain access to the server, he/she could steal the user's verifiers or private data and use them elsewhere to impersonate a legitimate user.

Since our scheme does not use any password tables, there is no way that an adversary or a privileged but malicious user could gather sensitive user information (i.e. verifiers), therefore he/she could not try and impersonate a user on any other network. Furthermore, when the user  $U_i$  initiates the authentication phase, the sensor node  $S_j$  forwards the message to the GWN, whereby a privileged user cannot extract  $U_i$ 's password from  $N_i = h(h(r_i || PW_i) || X_{GWN-U_i}) || X_{GWN-U_i} || T_1$ , because it is computationally infeasible due to the one-way property of the hash function. Consequently our scheme is resilient against both privileged insider and stolen-verifier attacks.

### 5.7. Stolen smart card and smart card breach attack

In view of the convenience of using smart cards, many cryptographic schemes use them for securely storing user's data, and for simple login and authentication. Smart cards are in general tamper-resistant and are safe against simple information stealing. According to Kocher, Jaffe and Jun [43], a highly motivated adversary could use a power analysis attack to breach the smart card and gather secure and sensitive information from it.

Although smart cards in general cannot be breached, such a possibility has to be considered. Let us assume a

smart card has been lost by or stolen from a user, and an adversary has obtained possession of it. The adversary manages to crack the smart card and obtains the information stored in it i.e.,  $r_i, MI_i, e_i, f_i, X_{GWN-U_i}$ . In order to impersonate the legitimate user  $U_i$  when trying to use the stolen smart card, the adversary would still have to initiate the authentication process, hence he/she would need to login with the smart card first. For an adversary to login successfully, she would need to know the legitimate  $U_i$ 's password in order to execute step 1 of the login phase. Even with the obtained information from the smart card, the adversary could not know  $U_i$ 's password, since it is computationally infeasible to extract the password from  $e_i = f_i \oplus h(h(r_i || PW_i) || X_{GWN-U_i})$ , due to the one-way hash function. Therefore, it can be concluded that the presented scheme is resilient against stolen smart card and smart card breach attack.

### 5.8. Impersonation attack

An impersonation attack is successfully executed when an adversary uses some of the private and secure information of a legitimate user and impersonates him/her to the server. In the case of our proposed scheme, the adversary would need to impersonate him/her to the GWN. Our proposed scheme is resilient to such attacks, since an adversary could eavesdrop on a legitimate user's authentication message and retransmit it to the sensor node but he/she still could not compute the session key, as described in the replay attack. Moreover, an adversary could steal a user's smart card and breach it, but without knowing the legitimate user's password, the adversary could not use it to impersonate the legitimate user.

### 5.9. GWN bypassing attack

In WSNs, the GWN sometimes plays the role of a trusted third party and bypassing it, would mean bypassing the authentication. Such an attack is not possible since our proposed scheme uses a rare four-step authentication model (Fig. 1), where a user firstly connects with a sensor node to initiate an authentication phase and not the GWN. In our proposed scheme the sensor node connects with the GWN after the authentication message has been initiated by the user. The sensor node delegates the authentication process to the GWN, whereby the GWN verifies both the sensor node and the GWN.

### 5.10. Many logged-in users with the same login-id attack

Let us assume a legitimate user has successfully signed into a server with his/her credentials. Meanwhile an adversary has managed to steal the user's login-id and password. The adversary impersonates the legitimate user and signs into the server with using same login-id, thereby executing the many logged-in users with the same login-id attack. If the server allows the login of the adversary and is not aware of the fraud, the attack is completed successfully.

This attack is not possible since in our proposed scheme we use a smart card for a user's login and authentication.

An adversary needs a legitimate smart card to login. Furthermore, as described in the smart card breach attack, the adversary would need the user's password to successfully execute the login phase and initiate the authentication phase. Since a smart card is uniquely personalized for successfully executing the registration phase (Step 7 of the  $GWN - U_i$  registration phase), multiple logins are not possible without a second smart card. Consequently our scheme is resilient against many logged-in users with the same login-id attacks.

#### 5.11. Password change attack

For an adversary to change a user's password, he/she would need to be in possession of the user's smart card. Moreover, even if the adversary obtained possession of the smart card by stealing it or finding a lost one, in order to change the password, he/she would need to know the old one first (Section 4.5). Even if the adversary managed to breach the smart card and reveal the information stored in it, it is still computationally infeasible for him/her to extract the password from the known data, as was described in Stolen smart card and smart card breach attack. Therefore our proposed scheme is resilient to password-change attacks.

#### 5.12. Denial-of-service attack

Since WSNs are resource constrained, the denial-of-service (DoS) attacks can be highly dangerous. There are multiple types of DoS attacks, e.g. Jamming, Flooding, Tampering, Misdirection, etc. [44]. DoS attacks can be executed over multiple network layers, i.e. Physical, Link, Routing, and Transport. As defense against such attacks is done mainly at the lower level, we have introduced only the threat of flooding, which can also affect the user and the proposed authentication scheme.

A DoS attack is not possible since our proposed scheme works on the principle of request-response communication; hence the user always receives a confirmation or rejection message from the sensor node, thus knowing that the received response is an authentic one and not a DoS attack. Furthermore, we have introduced timestamps into the scheme, which mitigate any consequential request. Consequently our scheme is resilient against DoS attacks.

### 6. Performance evaluation of the proposed scheme

This section introduces the performance evaluation like the comparison of computational costs, security features, communication costs and storage consumption, between our proposed scheme and other related user-authentication schemes for WSNs. The evaluations give an insight into the effectiveness of the proposed scheme and the possible network lifetime while using the scheme. Not all schemes have been developed with the same objective as our proposed scheme, hence the more similar one is the scheme proposed by Xue et al. [27]. Other schemes which do not focus on the IOT notion use other authentication

models which do not use smart cards, etc. Moreover, since our scheme is designed to enable a remote user to directly connect to a desired sensor node of the scheme, the sensor node has to run more computations than the sensor nodes of other related schemes, but nevertheless with the help of the evaluations we show that the proposed scheme is still the most effective one when taking into account all the performance aspects.

#### 6.1. Computational performance and security features analysis

The computational comparison of our scheme and others-related is summarized in Table 2, and the security features comparison in Table 3. In Table 2, where the computational comparison is made, we have summarized and presented only the login and authentication phase. In Table 3 we have only summarized those security features which have been presented by the authors of the related schemes.

The summary of computational comparison shows that our scheme uses only a lightweight hash operation, whereby some scheme also use symmetric or ECC encryption-decryption operations [23,26,27,40], which are more computationally costly. In comparison to the more similar scheme [27], which has almost the same security features as our scheme, our scheme is less computationally costly and has more security features, hence we have used seven hash operations less. Furthermore, in comparison to all other related schemes, our scheme supports much more security features and has thus proved to be more secure. The additional hash computations done by our scheme can be tracked back to the fact that our scheme uses the sensor node as the main actor within the network. The small computational overhead of our proposed scheme in comparison to others is worth the additionally gained security features and the novel approach of the first-step authentication based on IOT notion.

For a comparison, according to the authors [45], a typical hash operation run by a sensor node (e.g. Cross-Bow's MICA2) on a 24 byte messages, using the SHA-1 function, consumes about 0.075 J (Ws) of CPU energy when the CPU is in Active mode, while the encryption operation of the same 24 bytes message using the AES, consumes about 0.241 J (Ws) of CPU energy while in Active state, i.e. about 3 times more energy than a hash computation. A MICA2 sensor node, typically runs on two AA batteries (e.g. Alkaline), which combined have about 18500 J (i.e. 4400 mAh, 3 V), while the same node using our proposed scheme would consume 0.375 J (i.e. 5 times 0.075 J for each hash operation) for one user authentication cycle [46].

#### 6.2. Communication performance analysis

Transmission typically consumes more energy than computation; hence 1bit transmitted equals an execution of about 900 CPU instructions [47]. Since sensor nodes are resource constrained (e.g. MICA2 mote has 7.3 MHz CPU, 4 KB RAM, 128 KB memory and 38.4Kbps

**Table 2**

Comparison of computational cost between the proposed scheme and other related schemes.

Authentication scheme	User	Sensor node	GWN/cluster head	Base station
Proposed scheme	$7T_h$	$5T_h$	$7T_h$	–
Xue et al. [27]	$7T_h$	$6T_h$	$13T_h$	–
Das et al. [26]	$5T_h + 1T_{E D}$	–	$2T_h + 1T_{D E}$	$3T_h + 3T_{D E}$
Turkanović and Hölbl [40]	$4T_h + 1T_{E D}$	–	$1T_h + 1T_{D E}$	$2T_h + 3T_{D E}$
Yeh et al. [23]	$1T_h + 2T_{ECC}$	$3T_h + 2T_{ECC}$	$4T_h + 4T_{ECC}$	–
Chen and Shih [22]	$4T_h$	$1T_h$	$5T_h$	–
Khan and Alghathbar [21]	$4T_h$	$2T_h$	$6T_h$	–
Fan et al. [20]	$7T_h$	$2T_h$	$8T_h$	$2T_h$
Vaidya et al. [19]	$6T_h$	$2T_h$	$5T_h$	–
He et al. [18]	$5T_h$	$1T_h$	$5T_h$	–
Huang et al. [17]	$4T_h$	$1T_h$	$6T_h$	–

$T_h$  – time for a hash operation;  $T_{D|E}$  – time for symmetric-key decryption/encryption;  $T_{ECC}$  – the time for the encryption/decryption operation in ECC-160 algorithm.

**Table 3**

Comparison of security features between the proposed scheme and other related schemes.

Security Feature	Proposed scheme	Xue et al. [27]	Das et al. [26]	Turkanović and Hölbl [40]	Yeh et al. [23]	Chen and Shih [22]	Khan and Alghathbar [21]	Fan et al. [20]	Vaidya et al. [19]	He et al. [18]	Huang et al. [17]
Mutual authentication	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No
Key agreement	Yes	Yes	Yes	Yes	Yes	No	No	Yes	No	No	No
Password protection	Yes	Yes	Yes	Yes	Yes	No	Yes	–	–	Yes	–
Password-change	Yes	Yes	Yes	Yes	No	No	Yes	No	Yes	Yes	Yes
Dynamic node addition	Yes	–	Yes	Yes	–	No	–	No	No	No	No
User anonymity	Yes	Yes	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes
Replay attack resilience	Yes	Yes	Yes	Yes	No	Yes	Yes	–	–	Yes	–
Privileged-insider attack resilience	Yes	Yes	–	Yes	Yes	No	Yes	–	–	No	–
Stolen verifier attack resilience	Yes	–	–	Yes	–	–	–	–	–	–	–
Stolen smart card and smart card breach attack resilience	Yes	Yes	–	Yes	No	No	No	–	–	No	–
Impersonation attack resilience	Yes	Yes	–	Yes	Yes	No	Yes	–	–	No	–
GWN bypassing attack resilience	Yes	Yes	–	Yes	Yes	No	No	–	–	No	–
Many logged-in users with the same login-id attack resilience	Yes	–	–	Yes	–	–	–	–	–	–	–
Password change attack resilience	Yes	–	–	Yes	–	–	–	–	–	–	–
DoS attack resilience	Yes	–	Yes	Yes	–	No	–	Yes	Yes	No	No

Bandwidth [46,47]) it is even more important to lower the energy consumption, thus reducing the transmissions and the energy consumption of the Radio unit. The energy of sensor nodes inside the WSN is mainly consumed due to their constant sensing and coordination, whereby they collect and route data either to other sensor nodes in the WSN or to the GWN and base station. Such process requires the Radio to receive and transmit messages constantly. A typical message is about 40 bytes (i.e. 320 bits or the equivalent of 288,000 CPU instructions) [47].

Although user authentication is not a constant process like the aforementioned routing processes, it is still important to ensure low energy consumption while executing it. While developing our proposed scheme we also focused on these facts and tried to maintain the best ratio between security and performance, thus we tried to lower the communication cost of the sensor nodes.

**Table 4**

Comparison of communication cost between the proposed scheme and other related schemes.

Authentication scheme	Communication cost
Proposed scheme	4 Messages
Xue et al. [27]	6 Messages
Das et al. [26]	4 Messages
Turkanović and Hölbl [40]	4 Messages
Yeh et al. [23]	3 Messages
Chen and Shih [22]	4 Messages
Khan and Alghathbar [21]	4 Messages
Fan et al. [20]	3 Messages
Vaidya et al. [19]	5 Messages
He et al. [18]	3 Messages
Huang et al. [17]	3 Messages

In Table 4 we have summarized the communication comparison of our proposed scheme and other related scheme. We used the number of exchanged messages for



a successful user authentication as the key of the communication cost comparison. It can be noted from Table 4 that our scheme requires four messages for a successful user authentication, while the most similar scheme to ours (i.e. Xue et al. [27]) requires six messages and thus consumes more energy than our proposed scheme. Other related scheme have the same or less communication cost, but this is due to the fact of lower security as already mentioned and presented in the previous subsection.

### 6.3. Storage consumption comparison

Our proposed scheme is a computationally highly lightweight scheme designed especially for the user authentication of the WSN and, based on the IOT environment. The scheme ensures high security, while using only simple hash and XOR computations, thus consumes more storage than other related schemes. Since the novelty of the proposed scheme is the “sensor-node-first-contact-approach”, combined with its high security, the sensor nodes and the GWN have to carry some extra storage burden. In this subsection we present the storage consumption of our proposed scheme and compare it with other related scheme. We also show that although our scheme requires more

storage space than others related ones, it is still way below the maximum storage space a typical sensor node has.

A depiction of the storage consumption comparison is presented in Figs. 6 and 7. We made separate storage consumption analysis for the GWN and for the sensor nodes. Fig. 6 shows the comparison of the GWN storage consumption between our proposed scheme and other related schemes, while Fig. 7 focuses on the sensor node.

The analysis was done by inspecting each scheme in detail and calculating the sum of each variable length the GWN or the sensor node have to save into its memory, while successfully executing one Registration and Authentication phase. Before the inspection, for the purpose of an objective analysis, we firstly determined the length of each type of a variable which could be used in the schemes, regardless on the lengths some authors proposed. For example, an identity or password string were determined to be 8 bytes, a nonce 16 bytes, a timestamp 19 bytes, a hash value 20 bytes, and a symmetric key 16 bytes. Since the compared schemes use different protocols, we determined that regardless of the scheme architecture and their author's proposals, the WSN consists of one GWN or base station, ten sensor nodes and one user who is registering

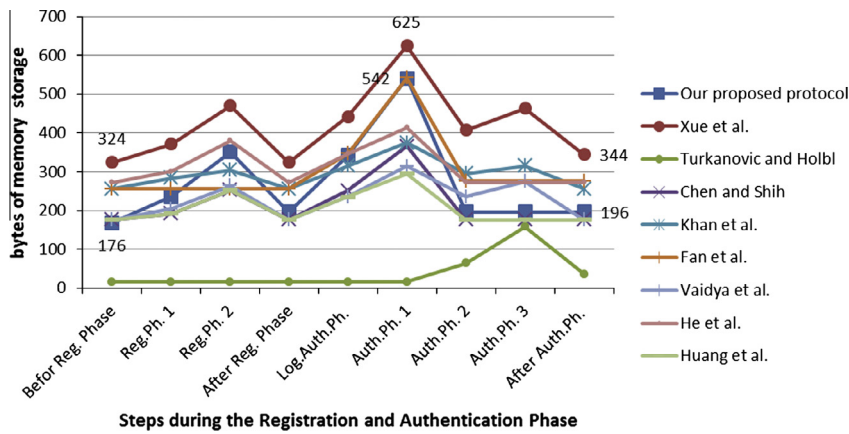


Fig. 6. The comparison of the GWN storage consumption between our proposed scheme and other related schemes.

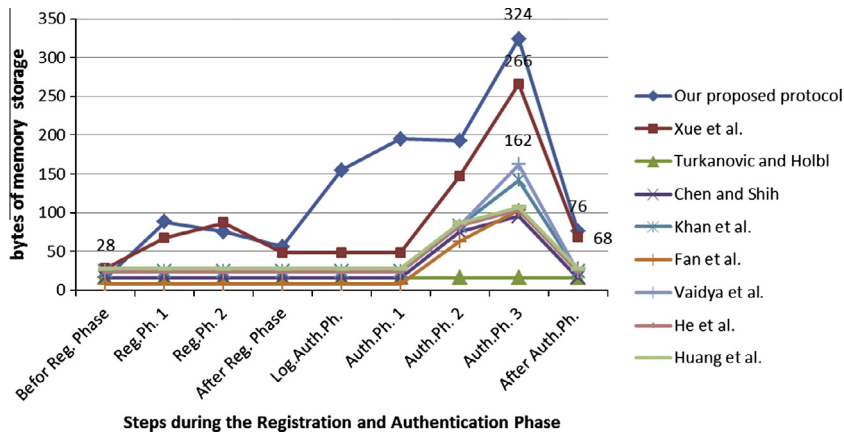


Fig. 7. The comparison of the sensor node storage consumption between our proposed scheme and other related schemes.



during the calculation. This is done due to the fact that almost every schemes generate secret keys for each sensor node and save all those keys, including their identities, into the GWN or base station before the actual deployment of the WSN happens. Because some schemes use this approach, we also included the registration phase into the analysis. Since some schemes (i.e. our proposed scheme and [27]) have multiple registration phases, one for a sensor node and one for a user, we had to choose only one for the calculation of the memory storage consumption, thus we chose the greedier part in order to insure objectivity of the comparison.

All parties of the authentication scheme (i.e. user, sensor node, GWN, base station) carry multiple steps before a successful handshake. During these steps a party receives some values, computes new ones with the help of the received and sends other values back or forward. While receiving the values, the parties have to save those into its memory. When new ones are computed, before they can be sent further, they also have to be saved into the memory alongside those already in the memory. After sending the values, the parties can then delete some, which are of no use for the further process. These steps repeat until the authentication is successful. Because of this procedure, we prepared the analysis in a way that we focused on the “peak of a moment” and the memory at this point, during the possible steps of the registration and authentication process. This is due to the fact that the parties, at a specific step (i.e. peak of a moment) of the registration or authentication procedure, save different amount of values.

It can be noted from Fig. 6 that a GWN processing our scheme, has to store the maximum 542 bytes of information into its memory, at one peak of a moment. This memory is afterwards cleaned to the amount of 196 bytes at the end of a successful authentication. While in comparison, the GWN, while processing the most similar scheme to ours (i.e. Xue et al.’s scheme), has to save the maximum 625 bytes in a peak of a moment, thus 83 bytes more than ours. And at the end of a successful authentication the amount of storage consumed by Xue et al.’s scheme, is also 148 bytes more (i.e. 344 bytes) than our proposed scheme (i.e. 196 bytes).

In Fig. 7, where the storage consumption of a sensor node was analyzed, it can be noted that a sensor node using our scheme has to save the maximum of 324 bytes of information in a peak of a moment, while if using Xue et al.’s scheme it would have to save the maximum of 266 bytes, thus 58 bytes less than ours. If the authentication ends successfully, a sensor node using our proposed scheme needs to save 76 bytes of information, while the sensor node using Xue et al.’s scheme 8 bytes less.

Even though our proposed scheme requires more storage space from a GWN or a sensor node in comparison to other related schemes, it is still an insignificant amount. This is due to the fact that a typical sensor node, like the Crossbow’s MICA2, has 128 K bytes (128,000 bytes) of available memory [46]. As already mentioned our proposed scheme requires 76 bytes of continuous memory and 324 bytes of memory in a peak of a moment.

## 7. Conclusion

This paper proposed a user and mutual authentication key agreement scheme for heterogeneous ad hoc WSNs using smart cards. This scheme focuses on the IOT notion, thereby enabling the remote user to directly contact a specific sensor node. Using a novel key agreement protocol, a user can negotiate a session key with a desired sensor node without personally connecting with the GWN, and use it to establish a secure communication. Our scheme uses a rare four-step authentication model which is, according to our research, the most suitable for such requirements and environment. To the best of our knowledge, this is a first of its kind scheme, designed to optimally work within the IOT environment.

The proposed scheme provides and enables mutual authentication between all parties, password protection, free password choice, password changing, and dynamic node addition.

Since WSNs are resource constrained, the scheme is designed to be lightweight, since it uses only hash and XOR computations, which are in comparison to others (i.e. asymmetric cryptography) less computationally intensive.

Moreover, although the scheme is highly lightweight it provides high security since it is resilient to replay attacks, privileged-insider attacks, stolen verifier attacks, stolen smart card and smart card breach attacks, impersonation attacks, many logged-in users with the same login-id attacks, GWN bypassing attacks, password-change attacks, and DoS attacks.

## Acknowledgments

The authors sincerely thank the anonymous reviewers and the Area Editor of the Journal for their helpful concerns, comments and suggestions, which helped with the improvement of the original content.

## References

- [1] L. Atzori, A. Iera, G. Morabito, The Internet of things: a survey, *Comput. Netw.* 54 (2010) 2787–2805.
- [2] K. Romer, F. Mattern, The design space of wireless sensor networks, *Wireless Commun., IEEE* 11 (2004) 54–61.
- [3] M. Bottero, B. Dalla Chiara, F.P. Deflorio, Wireless sensor networks for traffic monitoring in a logistic centre, *Transp. Res. Part C: Emerg. Technol.* 26 (2013) 99–124.
- [4] G. Owajaiye, Y. Sun, Focal design issues affecting the deployment of wireless sensor networks for pipeline monitoring, *Ad Hoc Netw.* 11 (2013) 1237–1253.
- [5] M.V. Ramesh, Design, development, and deployment of a wireless sensor network for detection of landslides, *Ad Hoc Netw.* 13 (2014) 2–18.
- [6] A. Somov, A. Baranov, D. Spirjakin, A. Spirjakin, V. Sleptsov, R. Passerone, Deployment and evaluation of a wireless sensor network for methane leak detection, *Sens. Actuat. A: Phys.* 202 (2013) 217–225.
- [7] Z. Sun, P. Wang, M.C. Vuran, M.A. Al-Rodhaan, A.M. Al-Dhelaan, I.F. Akyildiz, BorderSense: border patrol through advanced wireless sensor networks, *Ad Hoc Netw.* 9 (2011) 468–477.
- [8] X. Dong, M.C. Vuran, S. Irmak, Autonomous precision agriculture through integration of wireless underground sensor networks with center pivot irrigation systems, *Ad Hoc Netw.* 11 (2013) 1975–1987.

- [9] A. Hadjidj, M. Souil, A. Bouabdallah, Y. Challal, H. Owen, Wireless sensor networks for rehabilitation applications: challenges and opportunities, *J. Netw. Comput. Appl.* 36 (2013) 1–15.
- [10] M. Jou, J. Wang, Ubiquitous tutoring in laboratories based on wireless sensor networks, *Comput. Hum. Behav.* 29 (2013) 439–444.
- [11] A. Pietrabissa, C. Poli, D.G. Ferriero, M. Grigioni, Optimal planning of sensor networks for asset tracking in hospital environments, *Decis. Support Syst.* 55 (2013) 304–313.
- [12] V. Sivaraman, A. Dhamdhere, H. Chen, A. Kurusingal, S. Grover, An experimental study of wireless connectivity and routing in ad hoc sensor networks for real-time soccer player monitoring, *Ad Hoc Netw.* 11 (2013) 798–817.
- [13] A.A. Rezaee, M.H. Yaghmaee, A.M. Rahmani, A.H. Mohajerzadeh, HOCA: healthcare aware optimized congestion avoidance and control protocol for wireless sensor networks, *J. Netw. Comput. Appl.* 37 (2014) 216–228.
- [14] N. Barroca, L.M. Borges, F.J. Velez, F. Monteiro, M. Górski, J. Castro-Gomes, Wireless sensor networks for temperature and humidity monitoring within concrete structures, *Constr. Build. Mater.* 40 (2013) 1156–1166.
- [15] M.C. Bell, F. Galatioto, Novel wireless pervasive sensor network to improve the understanding of noise in street canyons, *Appl. Acoust.* 74 (2013) 169–180.
- [16] Y.-C. Chang, C.-C. Lin, P.-H. Lin, C.-C. Chen, R.-G. Lee, J.-S. Huang, T.-H. Tsai, eFurniture for home-based frailty detection using artificial neural networks and wireless sensors, *Med. Eng. Phys.* 35 (2013) 263–268.
- [17] H.-F. Huang, Y.-F. Chang, C.-H. Liu, Enhancement of two-factor user authentication in wireless sensor networks, in: *Proceedings of the 2010 Sixth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*: IEEE Computer Society, 2010, pp. 27–30.
- [18] D. He, Y. Gao, S. Chan, C. Chen, J. Bu, An enhanced two-factor user authentication scheme in wireless sensor networks, *Ad Hoc Sens. Wirel. Netw.* 10 (2010) 361–371.
- [19] B. Vaidya, D. Makrakis, H.T. Mouftah, Improved two-factor user authentication in wireless sensor networks, in: *Wireless and Mobile Computing, Networking and Communications (WiMob)*, 2010 IEEE 6th International Conference on, 2010, pp. 600–606.
- [20] R. Fan, L.-D. Ping, J.-Q. Fu, X.-Z. Pan, A secure and efficient user authentication protocol for two-tiered wireless sensor networks, in: *Circuits, Communications and System (PACCS)*, 2010 Second Pacific-Asia Conference on, 2010, pp. 425–428.
- [21] M.K. Khan, K. Alghathbar, Cryptanalysis and security improvements of 'two-factor user authentication in wireless sensor networks', *Sensors* 10 (2010) 2450–2459.
- [22] T.H. Chen, W.K. Shih, A robust mutual authentication protocol for wireless sensor networks, *Etri J.* 32 (2010) 704–712.
- [23] H.-L. Yeh, T.-H. Chen, P.-C. Liu, T.-H. Kim, H.-W. Wei, A secured authentication protocol for wireless sensor networks using elliptic curves cryptography, *Sensors* 11 (2011) 4767–4779.
- [24] M.L. Das, Two-factor user authentication in wireless sensor networks, *Wirel. Commun. IEEE Trans.* 8 (2009) 1086–1090.
- [25] A.K. Das, An unconditionally secure key management scheme for large-scale heterogeneous wireless sensor networks, in: *Proceedings of the First International Conference on COMMunication Systems and NETWORKS*, IEEE Press, Bangalore, India, 2009, pp. 653–662.
- [26] A. Das, Kumar, P. Sharma, S. Chatterjee, S.J. Sing, Kanta, A dynamic password-based user authentication scheme for hierarchical wireless sensor networks, *J. Netw. Comput. Appl.* 35 (2012) 1646–1656.
- [27] K. Xue, C. Ma, P. Hong, R. Ding, A temporal-credential-based mutual authentication and key agreement scheme for wireless sensor networks, *J. Netw. Comput. Appl.* 36 (2012) 316–323.
- [28] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, Wireless sensor networks: a survey, *Comput. Netw.* 38 (2002) 393–422.
- [29] S. Ozdemir, Y. Xiao, Secure data aggregation in wireless sensor networks: a comprehensive overview, *Comput. Netw.* 53 (2009) 2022–2037.
- [30] C. Schurgers, G. Kulkarni, M.B. Srivastava, Distributed assignment of encoded MAC addresses in sensor networks, in: *Proceedings of the 2nd ACM International Symposium on Mobile Ad Hoc Networking & Computing*, ACM, Long Beach, CA, USA, 2001, pp. 295–298.
- [31] K. Sohrabi, J. Gao, V. Ailawadhi, G.J. Pottie, Protocols for self-organization of a wireless sensor network, *Pers. Commun. IEEE* 7 (2000) 16–27.
- [32] G. Sharma, S. Bala, A.K. Verma, Security frameworks for wireless sensor networks-review, in: *2nd International Conference on Communication, Computing & Security [ICCCS-2012]*, vol. 1, 2012, pp. 978–987.
- [33] R. Watro, D. Kong, S.-F. Cuti, C. Gardiner, C. Lynn, P. Kruus, TinyPK: securing sensor networks with public key technology, in: *Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks*, ACM, Washington DC, USA, 2004, pp. 59–64.
- [34] W. Diffie, M.E. Hellman, New directions in cryptography (invited paper), *IEEE Trans. Inf. Theory* IT-22 (1976) 10.
- [35] J. Xu, W.-T. Zhu, D.-G. Feng, An improved smart card based password authentication scheme with provable security, *Comput. Stand. Interf.* 31 (2009) 723–728.
- [36] R. Song, Advanced smart card based password authentication protocol, *Comput. Stand. Interf.* 32 (2010) 321–325.
- [37] K.H.M. Wong, Y. Zheng, J. Cao, S. Wang, A dynamic user authentication scheme for wireless sensor networks, in: *Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing*, vol. 1, (SUTC'06) – vol. 01, IEEE Computer Society, 2006, pp. 244–251.
- [38] D. Nyang, M.-K. Lee, Improvement of Das's two-factor authentication protocol in wireless sensor networks, in: *CORD Conference Proceedings*, 2009 (2009).
- [39] S. Xu, X. Wang, A new user authentication scheme for hierarchical wireless sensor networks, *Int. Rev. Comput. Softw.* 8 (2013) 197–203.
- [40] M. Turkanović, M. Hölbl, An improved dynamic password-based user authentication scheme for hierarchical wireless sensor networks, *Electron. Electric. Eng.* 19 (2013) 109–116.
- [41] C.-T. Li, C.-Y. Weng, C.-C. Lee, An advanced temporal credential-based security scheme with mutual authentication and key agreement for wireless sensor networks, *Sensors* 13 (2013) 9589–9603.
- [42] M. Turkanović, M. Hölbl, Notes on "a temporal-credential-based mutual authentication and key agreement scheme for wireless sensor networks", *Wireless Pers. Commun.* (2013) 1–16.
- [43] P.C. Kocher, J. Jaffe, B. Jun, Differential power analysis, in: *Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology*, Springer-Verlag, 1999, pp. 388–397.
- [44] A. Wood, J.A. Stankovic, Denial of service in sensor networks, *Computer* 35 (2002) 54–62.
- [45] C. Chih-Chun, D.J. Nagel, S. Muftic, Assessment of energy consumption in wireless sensor networks: a case study for security algorithms, in: *Mobile Adhoc and Sensor Systems*, 2007 MASS 2007 IEEE International Conference on, 2007, pp. 1–6.
- [46] I. Crossbow technology, MICA 2, Wireless Measurement System, in, EOL: Crossbow.
- [47] M.A. Simplicio Jr., B.T. de Oliveira, C.B. Margi, P.S.L.M. Barreto, T.C.M.B. Carvalho, M. Näslund, Survey and comparison of message authentication solutions on wireless sensor networks, *Ad Hoc Netw.* 11 (2013) 1221–1236.



**Muhamed Turkanović** received his B.Sc. degree in Computer Science and Informatics from the University of Maribor in 2011. He is currently pursuing his Ph.D. degree in Computer Science and Informatics from the University of Maribor. His current research interests include cryptography, information security, network security and wireless sensor networks.



**Boštjan Brumen** received his B.Sc. in Electrical Engineering from the University of Maribor in 1996 and his Ph.D. degree in Computer Science and Informatics from the University of Maribor in 2004. He is currently an assistant professor (part time) at Faculty of Electrical Engineering and Computer Science, University of Maribor. His research interests include cryptography, network security, data security, data analyses, and data reusability. As a member of Database Technologies Laboratory he actively participates in several interna-

tional and national projects, related to data issues. He cooperates with researchers at Tampere University of Technology since 1999. The results were published at several international conferences and in journals.



**Marko Hölbl** received his Ph.D. degree in Computer Science and Informatics from the University of Maribor in 2009. His first degree is Bachelor of Science (Bs.C.) in Computer Science and Informatics, received from University of Maribor in 2004. He is currently an assistant professor for Information Technology at the Faculty of Electrical Engineering and Computer Science, University of Maribor. His research interests include cryptography, network security, web security, smart cards and mobile communications security.