TEM Journal. Volume 6, Issue 2, Pages 192-204, ISSN 2217-8309, DOI: 10.18421/TEM62-02, May 2017.

Protocol Design for Secure Communication in WSN

Oguz Ata [1], Hasan H. Balik [2], Erdem Ucar [3]

[1]*Department of Mechatronics Engineering, Istanbul Aydin University, Istanbul, Turkey*
[2]*Department of Computer Engineering, Yildiz Technical University, Istanbul, Turkey*
[3] *Department of Computer Engineering, Trakya University, Edirne, Turkey*

Abstract **–Wireless sensor networks are widely used in military applications, monitoring of environmental information, health, domestic, and industrial applications. Despite of these wide usages, the main problem of this kind of topologies is the increasing security needs. This paper has been targeted to develop a novel message identity validation protocol in WSN to provide secure data identity in order to decrease some security threads. Introduced protocol also includes key distribution, node identification, sensitivity mechanisms, to strengthen the introduced method. Moreover, there is also repeated symmetrical key update mechanism without the need to synchronization. Finally, the developed protocol has also been tested in terms of various network parameters.**

Keywords **– User authentication; cryptanalysis; Protocol design, message authentication, security, wireless sensor networks**

1. Introduction

The sizes of the computers are decreased whereas process capabilities are increased with developments in technologies. Computers and computer networks have become essential components of social life starting from education to commerce and converted the society from an industrial to an information society.

The sensors are used in various areas from military, health, domestic and industrial applications to the observation of the environmental information [1-3]. Desired information can be sensed by the courtesy of the hardware on the sensors, processed by their processors and the information can be sent to the predetermined centers by the receiver/transmitter units.

Providing the reliability of information is an important issue. It is not possible to provide the security of each sensor. It is essential to be sure receiving the data from the right source at the right time. For example, in data replication attack, some data is stored and then sent to the system at different time to disorder both information reliability, and integrity of the system.

In this paper; novel protocol is designed and developed to provide secure data identity for the wireless sensor networks. In the proposed approach, the HMAC (Hash Message Authentication Code) key that is used for each new packet is generated using both mask and the current time function. Similar to other protocols in the literature (i.e. One Time Password protocol), our protocol is also regularly change the security key that is used amongst nodes. The major featured advantage of our protocol is that the sink can adapt itself to the new key without possessing it. In our protocol there is no need to resent newly generated key to the sink again. This protocol is explained and tested in detail in the following sections.

The paper is organized as follows: Section 2 describes related works. Section 3 describes introduced protocol to provide data identity in detail. In Section 4 presents security architectures in detail. Section 5 presents the simulation results by considering changing node quantity, deployment of node (grid and random) and different transmission platforms (normal and ideal). In Section 6 the results are compared according to the widely used Key Management Schemes. Finally Section 7, concludes this study.

DOI: 10.18421/TEM62-02
https://dx.doi.org/10.18421/TEM62-02

Corresponding author: Oguz Ata,
Department of Mechatronics Engineering, Istanbul Aydin University, Istanbul, Turkey
Email: oguzata@gmail.com

2. Related Works

In this section briefly are discussed related schemes for wireless sensor network. Sensor nodes are built with limited equipment to increase life time. Sensor nodes have limited storage unit, limited power, limited communication ability and limited computing power. Sensor nodes also should provide security of network communication. According to limited equipment and providing security requirements, lightweight security protocols and its designs are become popular in WSN and IOT research area.

Asymmetric cryptology provides more secure communication but it consumes too much computing power, communication requirement and storage for wireless sensor network. In 2004 Watro et. al [4]. proposed a user authentication scheme based on asymmetric cryptology with RSA and diffie helman algorithm which is TinyPK.

Benenson et. al. [5] proposed public-key based user authentication protocol with using elliptic curve cryptography (ECC). This protocol is more feasible than TinyPK.

Yeh et. al. [6], Xu et. al. [7] and Song [8] are also proposed asymmetric user authentication based on Diffie-Helman key agreement protocol. These protocols are having the same problem which is memory requirement therefore are not usable for most WSN applications.

Wonk et. al. [9] proposed password based user authentication with symmetric encryption. This protocol is used only one-way hash function and XOR operation. Therefore it is lightweight protocol. However some vulnerability is discovered for this protocol [10-13]. Das [13] improved Wonk et. al.'s scheme and proposed a two-factor user authentication protocol. Das's scheme resist some attack types such as "replay", "offline-password guessing", "multiple logged in same identifier", "stolen verifier".

In 2010 Khan and Alghathbar [14] proposed improved Das's scheme. Khan and Alghathbar's scheme also supports mutual authentication. And this scheme has solved unsecure password problem.

Additionally, Chen and Shih [15] also pointed out towards parallel session attack vulnerability of Das's scheme and proposed enhanced mutual authentication between all participant but later replay attack and forgery attack weakness are discovered for Chen and Shih's scheme. Vaidya et. al. [16] also improved Das's scheme and proposed the improved 2-factor user authentication scheme.

Xue et. Al. [17] proposed a temporal-credential-based scheme that uses only one way hash function and XOR operation. This scheme provides security without more computational process and more consuming power. Turkanovic et. Al. [18] proposed one way hash function based scheme but Amin et. al. showed that vulnerability such as smart card attack, offline-password guessing etc.

3. Introduced Protocol

To consider all possible security aspects, the system should be considered from the perspective of the attacker. The architectural design of the system has a great importance because the attacker observes the system as a whole and tries to determine the weakest link of the system. The purpose of this paper is to reduce security weakness of WSN by introducing enhanced protocol that guaranties data identity. This section provides detailed information about the introduced protocol. Developed protocol includes;

- Key distributions
- Node identification
- Protocol of authentication of message identity
- Authentication
- Sensitivity

There are five types of network message packet. All nodes behave according the following packet types.

- **REPORT_PACKET_NAME: to send sensed data to CN**
- **NETWORK_JOINING_PACKET_NAME: to send join request and to get Pk from CN**
- **PUBLIC_KEY_SENDING_PACKET_NAME: to send public key**
- **NODE_KEYS_PACKET_NAME: to send SN's keys to CN. This data encrypted by pk**
- **NODE_KEYS_OK_PACKET_NAME: "Node keys OK packet successfully received"**

All corresponding codes are written and tested in Castalia 3.2 simulation platform which is one of the commonly used platforms in this resource area.

Notation	Description
SN_{id}	Sensor Node
CN	Central Node or Sink
Pk	Public Key
T_{id}	Current Timestamp of SN_{id} İf id=0 that means CN otherwise SN.
$\oplus$	XOR operation
Rnd()	Random value generator function
$Ckey_{id}$	
$skey_{id}$	
MAC_{key}(key,data)	The HMAC function that generate a message authentication code with using key and data
==	Logical "is equal"
||	Concatenation operation

3.1. Key Distribution Mechanism (KDM)

As seen in Figure 1. the central node (CN) is the node that collects all field information from surrounding nodes. The key to be used by the central node (CN) to get the confidential information of the other nodes securely is forwarded to the node that will perform the communication in the network by the developed Key Distribution Mechanism (KDM). All nodes in the network must have this key in order to identify themselves to the CN before joining WSN. This key can be either predefined by manufacturer or be created by CN. It is assumed in our scenario that the key is deployed into the CN by the manufacturer. In addition CN also called "sink" and distributes the pre-deployed public key. A candidate node that has no public key, delivers "key request" to the network. The key request scenario is shown in Figure 2.

It the Figure 2. step 1, candidate nodes which do not have the CN's public key, send request to the sink. In this stage all neighbors that previously joined the WSN own the key. Then, the nodes can obtain the public key from the available adjacent nodes, which have previously obtained the key, without further communicating to the CN, (second and third steps in Figure 2.). This procedure reduces unnecessary communications. The obtained key is stored by the candidate in a table to be used in the future.

The designed code part that process of KDM is shown in Figure 3. As seen that the code evaluates the network participation packages of the node and gives the decision for routing. In the participation packet, a sequence number is also used for to critical purposes. These are:

- Reduce unnecessary repeated process
- Reduce unnecessary packet propagation

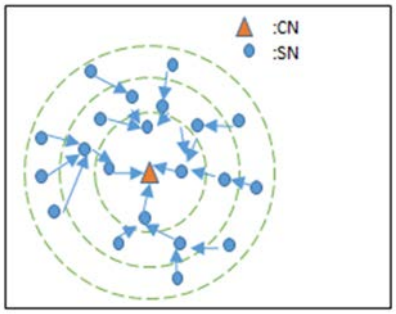

Figure 1. Data flow in WSN

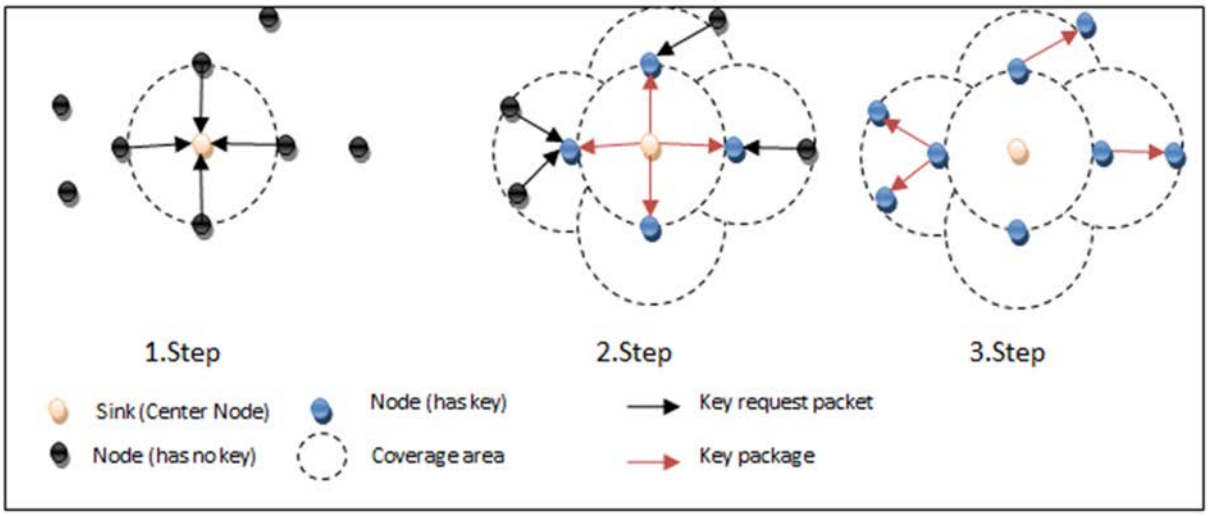

Figure 2. Key Distribution Mechanism

To avoid a repeating process, packet sequence number is also controlled by the code. If the package was seen by the node before, the packet that came from that node is not evaluated. Somehow this package reaches to the CN; CN sends the required key information to the network. In normal cases previously joint nodes provide this information to the network. During these processes, every node in the network stores all the requesting nodes into the relevant tables in order to locate the sink (CN) position.

Phase 1: Getting pk to accomplish secure keys exchange between CN and SN_x

For each SN

- **Check**(pk) if this node hasn't pk, **setTimer**(toBoardcast "NETWORK_JOINING_PACKET_NAME" packet)
- **For every packet which is coming from Network layer Check**(packet type) : if the type is "NETWORK_JOINING_PACKET_NAME " then
- Check(pk) **if** this node has public key then **send**(pk,toSouce)
- **update**(PublicKeyDistributionTable)
- Else **setSourceAdres**(selfAdres) and broadcast this packet

For *CN*

- **For every packet which is coming from Network layer Check**(packet type) : if the type is "NETWORK_JOINING_PACKET_NAME " and check if the packet is new, then create PUBLIC_KEY_SENDING_PACKET_NAME
- Send(pk,toSource)
- **update**(PublicKeyDistributionTable)

Phase 2: Sending cryptic information to CN.

For each SN

- For every packet which is coming from Network **layer Check**(packet type) :
- if the type is " PUBLIC_KEY_SENDING_PACKET_NAME and destination==self
- rkey=**rnd()**;
- $ckey_{id}$= rkey+T_{id}
- $skey_{id}$=**rnd();**
- encrypt(ckey,pk)
- encrypt(skey,pk)
- create NODE_KEYS_PACKET_NAME type packet
- **concat({id,ckey,skey}),**put encrypted ckey and skey in to packet and sent to which node that came PUBLIC_KEY_SENDING_PACKET_NAME packet before. This information stores in PublicKeyDistributionTable
- wait for approval message
- else forward this message to destination

Phase 3: Approval message

For CN

- For every packet which is coming from Network layer Check(packet type) :
- if the type is " NODE_KEYS_PACKET_NAME
- decryptData=**Decrypt**(crypyData)
- **split**(decryptData)
- $node\_ckey_{id}$ =T_0-ckey
- $node\_skey_{id}$=skey
- **update**(Key_Table) with {id,$node\_ckey_{id}$,$node\_skey_{id}$}
- • create NODE_KEYS_OK_PACKET_NAME type packet
- **send**(OkPk,node)

For SN

```
else if
(packetName.compare(NETWORK_JOINING_PACKET_NA
ME) == 0 ) {

  if (updateNetworkJoiningPacketReportTable((int)data,
sequenceNumber))
{ if(isSink){
 sendPublicKey(kaynak.c_str());
 updatePublicKeyDistributionTable((int)data); }
 else if(isSinkResponsed == 0 ){
 rcvPacket -> setKaynak(selfs.c_str());
toNetworkLayer(rcvPacket->dup(),
BROADCAST_NETWORK_ADDRESS);}
else if(isSinkResponsed >= 1 ){
```

Figure 3. Key Distribution Mechanism code part

3.2. Node Identification Mechanism (NIM)

Node identification process is the stage where the CN has information about the participating node. In this stage, the participating node gets the cryptic information with which message authentication code (MAC) is produced. It is assumed that the key distribution process is completed before this stage. The node having a communication key sends its own identification information to the central node (CN) and waits for approval. After the arrival of authentication message, the network participation process of the node is completed and it starts to work in its normal form. The stage of node identification depending on time is shown in Figure 4.

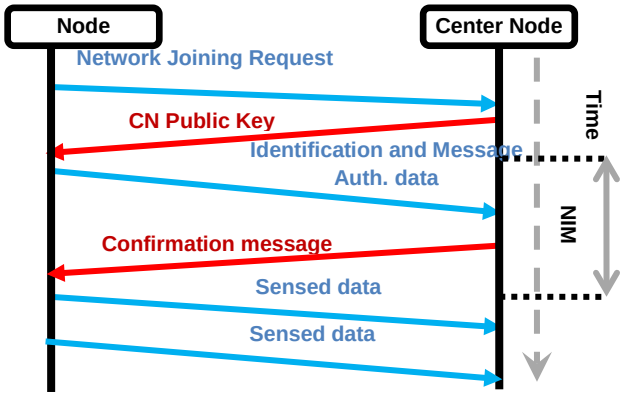

Figure 4. Node Identification Mechanism depending on time

Similar communication method suggested in KDM, and it is used to send the identification information to the central node (CN). The central node decodes both the identification and MAC then stores them in the table to be used in the communication with the relevant node. After that, it sends the authentication package to the node from which the information is received, to be directed to the next node in the network. This process is repeated on every node till the package arrives to the requesting node. In other words, the confirmation message is sent by using the same way that authentication data is received. The communication mechanism is shown in Figure 5.

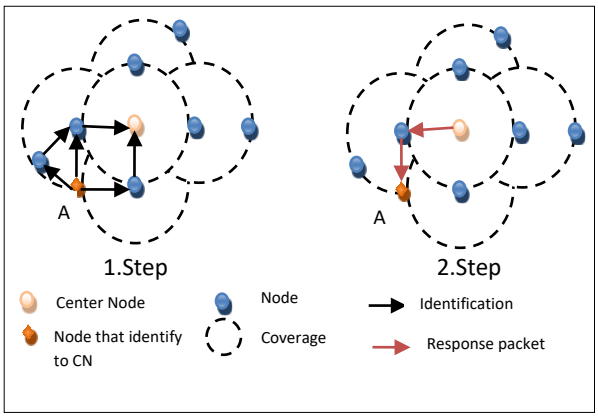

Figure 5. Node Identification Mechanism

```
else

if (packetName.compare(NODE_KEYS_PACKET_NAME) == 0) {
if(isSink){

            if(updateNodeKeysReportTable((int)data,
sequenceNumber, stringData1 ,  atoi(Kaynak.c_str())) == 1)
             {

            decodeNodeInfo((int)data, stringData2,
stringData1);
```

Figure 6. Node Identification Mechanism Code

As seen in Figure 6., if CN receives the package, the returned value is controlled by calling the NodeKeysReportTable updating function. The updating function records node information and returns the "true" value. If node information are founded in CN's table, it returns the "false" value. If the returned value is "true"; then all the information regarding this node are decoded with the decodeNodeInfo function and the encrypted information to be used in the communication with the node is recorded on the relevant table. Then the authentication message is sent by the

sendNodeKeysOk function in order to have the node participate in the network by CN.

The code given in Figure 7. shows how a package will be evaluated when a package is authenticating the participation of a node in the network which is forwarded by other nodes or came from CN.

```
else if(packetName.compare(NODE_KEYS_OK_PACKET_NAME)
== 0 && !isSink)

{           if(self == (int) data && isSinkResponsed == 1)
            {isSinkResponsed =2;  }

            else if(self == (int) data && isSinkResponsed == 2){
```

Figure 7. Code of packetName.compare

As clearly seen in Figure 7., if the target of the receiving package is itself, the node will start to work as a participated normal node by changing the status of isSinkResponsed flag to 2. If the target is not itself, then it will forward the package to the neighboring nodes by calling the sendNodeKeysOkForward function.

In Figure 8. the forwarding process of the sendNodeKeysOkForward function is described. The target part of the package is searched in the "node_keys_report_table" table. By finding the node required to be forwarded to send the package to the target; it is continued with the next step. Then the required changes are made in the package and the package is forwarded to its next target. This process continues till the package reaches its target by the forwarding nodes.

```
int OgobridgeTest::sendNodeKeysOkForward(int dst,
ApplicationPacket *rcv){

        for(int i = 0  ; i<node_keys_report_table.size(); i++) {

 if(node_keys_report_table[i].source == dst){

  for(int k = 0 ; k<node_keys_report_table[i].parts.size(); k++)

        {par = k;

hopBacWSddress =  node_keys_report_table[i].lastHopAddress[par];

        pos = i;
```

Figure 8. Code of sendNodeKeysOkForward function

3.3. Protocol of authenticating the message identity

After a node completes its network participation processes, it starts to work in its normal form. In case the sent value is over a specific sensitivity level, this information means that the information should be send to CN. While this information is being sent to CN, it can pass through various nodes. At these nodes, the information can be changed/degraded or the attacker can send wrong information to CN in the name of another node. The usage of the message authentication code (MAC) is preferred in this paper study to prevent such cases. There are many MAC algorithms in the literature, including CBC-MAC, HMAC, XMAC and CMAC [19].

The selection of the MAC algorithm became it should be performed carefully affect the energy use in WSN. So according to [19], HMAC is chosen in this study. The selection of key is very important in terms of secure communication in the network to authenticate data identity. Therefore "Individual Key" method is chosen in the designed protocol.

In the presented design, the identity authentication is performed for the data incoming from the nodes for the communication with CN by the key information is notified to CN in the node identification protocol. This mechanism is shown in Figure 9.

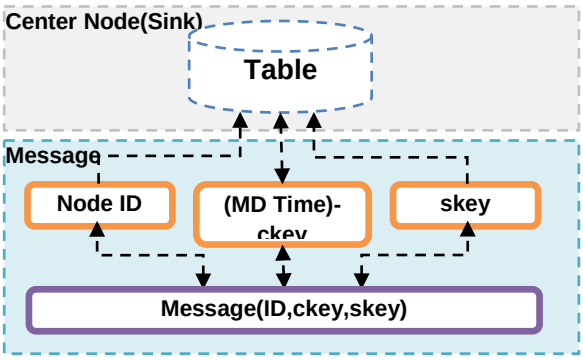

Figure 9. Node information recording process of CN

To increase security, an ever-changing key mechanism is developed instead of a fixed key method. The first step of the introduced mechanism is submitting the key information to CN which will be used by the node in communication, as explained in the node identification mechanism. In this stage, submission of the countermeasure (ckey = rkey+tkey), which is the total of the random numerical values (rkey) formed by the node and the time value of the node (tkey) to CN and the other random value (skey) submission formed by the node, are completed (Figure 9.). The submitted ckey and skey are stored by the CN on their own tables. skey is stored as the same but ckey is stored on the table as the difference of the current time value of CN and the ckey. The current time value of CN is subtracted from the difference (MD Time-ckey) on the table when required and the current ckey value of the node is obtained. Thus a time-dependent, ever-changing ckey and a fixed skey value is kept by the CN.

3.4. Authentication Mechanism

Message authentication mechanism controls the identity of the sender node by using the message authentication code of the message incoming to CN. If the identity of sender is authenticated, the message will be accepted; otherwise it will be rejected by CN.

HMAC is used for the authentication process, and it creates the authentication code according to a specific password. Thus CN which will perform the controlling process should use the same password with the sender. CN needs some information to find out the password when the source node is used while sending the message. In Figure 10., the part specifying the "source node address" of the message with the shown PDU structure is taken and searched

in the password tables where the CN keeps the password information. If no information is found, the message is rejected. if stored node information is found in the password table of CN, controlling process starts.. CN has to know HMAC key of source node to check the message authentication code. CN subtracts ckey from current time stamp for calculating ckey. Final stage of calculation is masking/demasking the HMAC key which is calculated value XOR with skey. The result of last calculation which is final value for the password is to be used by CN for the authentication of the message.

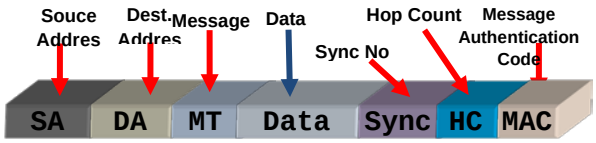

Figure 10. PDU structure of the message

The main advantages of this process that XOR process provides the masking of the data to be used as a password. If the key is known by the attacker anyway, the attacker will not be able to estimate the next key because of this masking process.

3.5. Sensitivity Mechanism

The time function used for the password change also alters the message authentication code. The password authentication mechanism also controls with the previous password value of the node for a specific period of time to have the packages send during the change accepted by CN. The sensitivity mechanism suggested for the current study is shown in Figure 11. As seen in Figure 11; if the password changing period is "T" and the sensitivity is "d"; CN, till the T+d time, controls the package incoming from the source node with the previous password value in case it is not authenticated with the current password. Thus the packages sent recently to the password changing time are protected against package losses.

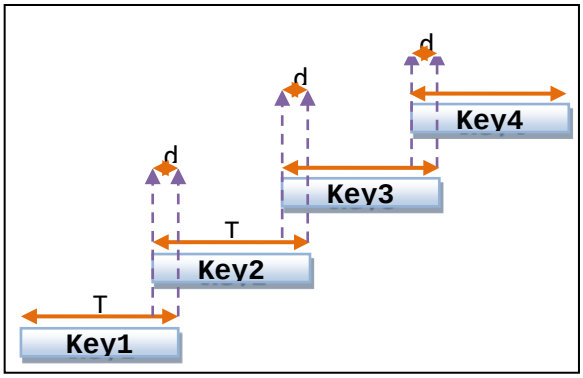

Figure 11. Sensitivity mechanism

4. Security Analysis

In this study, we have used nonmonotonic cryptographic protocol(NCP) to accomplish security analysis. This protocol is also known as Rubin Logic. [20]

Symbol	Description
SN	Sensor Node
CN	Center Node (sink)
T_{SN}	Current time value of sn or cn
$ckey_{SN}$	$r + T_{SN}$
$skey_{SN}$	Secret key for masking
$pkey^{+-}$	Private and Public key
r	Random value
$\rightarrow$	Assignment operator
Hash()	Hash function
XOR()	Xor function

4.1. Protocol Specification

- Principal Set: P={SN,CN} SN is the initiator of protocol.
- Rule Set: Inference rules [20] [21]
- Secret Set: $S=\{pkey^{+-},ckey_{SN},skey_{SN},T_{CN}\}$
- Observers Set:
- Observers($pkey^{+}$):{CN}
- Observers($ckey_{SN}$, $skey_{SN}$):{SN}
- Observers(T_{CN},$pkey^{-}$)={CN}
- Observers(T_{SN})={SN}

Local sets: Local sets of SN and CN shown below.

Entity SN

POSS(SN)={TSN}

BEL(SN)={ #TSN}

BL(SN) ={

Phase 1:

- [sn1]**Send**(boardcast,$pkey^{+}$req)
- [sn2]**Update(**$pkey^{+}$req**)**
- [sn3]**Receive**(CN, $pkey^{+}$)

Phase 2:

- [sn4]Generate-secret(r)
- [sn5]Generate-secret($skey_{SN}$)
- [sn6]$r+T_{SN}\rightarrow ckey_{SN}$
- [sn7]**Concat**({$skey_{SN}$, $ckey_{SN}$})$\rightarrow key_{SN}$
- [sn8]**Apply-asymkey**(key_{SN}, $pkey^{+}$)$\rightarrow \lambda$

- [sn9]**send**(CN, λ)
- [sn10]Update(λ)
- [sn11]**Receive**(CN, {OkMessage, ψ})
- [sn12]$ckey_{SN}^{*} \leftarrow r+T_{SN}$
- [sn13]**XOR**($ckey_{SN}^{*}$, $skey_{SN}$) →ν
- [sn14]**Hash**(hmac(ν),OkMessage) → ψ^{*}
- [sn15]**Check**(ψ,ψ^{*})

Phase 3:

- [sn16]$ckey_{SN}^{*} \leftarrow r+T_{SN}$
- [sn17]**XOR**($ckey_{SN}^{*}$, $skey_{SN}$) →ν
- [sn18]Hash(hmac(ν),data) →ψ
- [sn19]**Send**(CN, { data , ψ})
- [sn20]Update({ data , ψ})

}

Entity CN

POSS(CN)={TCN, pkey+, pkey- }

BEL(CN)={ #TCN,# pkey+,# pkey- }

BL(CN) ={

Phase 1:

- [cn1]**Receive**(SN, $pkey^{+}$req)
- [cn2]**Send**(SN, $pkey^{+}$)
- [cn3]**Update(**$pkey^{+}$**)**

Phase 2:

- [cn4]**Receive**(SN, λ)
- [cn5]**Apply-asymkey** (λ,$pkey^{-}$)
- [cn6]**Split**(λ)
- [cn7]$ckeydif_{SN} \leftarrow T_{CN}$- $ckey_{SN}$
- [cn8]$ckey_{SN}^{*} \leftarrow T_{CN-}$ $ckeydif_{SN}$
- [cn9]**XOR**($ckey_{SN}^{*}$, $skey_{SN}$) →ν
- [cn10]forget(λ,$ckey_{SN,}$ $ckey_{SN}^{*}$)
- [cn11]**Hash**(hmac(ν),OkPk) →ψ
- [cn12]**Send**(SN, { OkPk , ψ})
- [cn13]**Update(**{ OkPk , ψ})

Phase 3:

- [cn14]**Receive**(SN,{data, ψ})
- [cn15]$ckey_{SN}^{*} \leftarrow T_{CN-}$ $ckeydif_{SN}$
- [cn16]**XOR**($ckey_{SN}^{*}$, $skey_{SN}$) →ν
- [cn17]**Hash**(hmac(ν),data) →ψ^{*}
- [cn18]**Check(**ψ^{*},ψ**)**

4.2. The Analysis

In this section, we discuss the analysis of proposed protocol. In order to analyze the protocol, Rubin logic will be used.

The protocol has three phases which are phase I as Registration phase, phase II as login phase and phase III as key agreement phase.

BL(SN) is the initiator of the protocol, so the protocol starts to execute at [sn1]. First two actions [sn1] and [sn2] are executed and then update function is performed. Next action has to be executed in the behavior list of CN (BL(CN)). When CN receive key request packet is received, it sends the public key [cn2] to SN [sn3]. After these actions, the global and local sets are shown below.

POSS(SN):{ TSN ,pkey+}
BEL(SN):{ TSN, #pkey+}
Observers(pkey+):{CN,SN}

Then the [sn4]-[sn8] actions of BL(SN) are executed. At [sn4] and [sn5] some security secrets are generated then possession and observers sets are also updated. At [sn6], the current time value of SN which is T_{SN}, and "r" value generate $ckey_{SN}$. At [sn7], as a result of concat action, key_{SN} value is created.At [sn8] key_{SN} is crypted(λ) then at [sn9] λ is send to CN. At this point possession, observer and belief sets are updated.

POSS(SN):{ TSN ,pkey+,r, TSN, λ}
BEL(SN):{ TSN, #pkey+,#r,# TSN ,# λ}.
Observers(r, TSN):{SN}
Observers(λ):{SN,CN}

After completion of[sn10], [cn4][c13] actions of BL(CN) are executed. When CN receives crypted "λ", it is decrypted and split λ then { skeySN, ckeySN } are added POSS(CN), at this point possible origin rule and submessage origin rule(for public key) are satisfied. Possible origin rule and submessage origin rule is successfully applied.

At [cn7] CN store current time value minus ckey in ckeydifSN. Then to mask the ckeySN*, XORing at [cn9]. And hash code is generated 4 using hmac function. Then the "ok" packet is sent to SN. And possession set is shown below.

POSS(CN)={TCN, pkey+, pkey- , skeySN, ckeySN, ν,ψ, ckeydifSN ,{ OkPk , ψ}}
Observers(skeySN):{CN,SN}
Observers(ckeySN):{CN,SN}

When the "ok" packet is received by SN at [sn11], the packet's MAC code is checked. Current ckeySN

secret is generated at [sn12] then it is masked with $skey_{SN}$ at [sn13]. And at [sn15] ψ and ψ^* are compared to each other. If both codes are the same, this means CN's and SN's secret codes are synchronized and fresh. After this acknowledgement, the protocol goes to normal communication mode.

At [sn16][sn20], MAC code generating with using current secret keys and data, then generated MAC code and data are sent to CN at [sn19].

Some related rules are described below. Detail of rubin logic is disclosed [20] and [21]

- Possible Origin Rule:

$$\frac{X \in POSS(P), X \text{ contains } x_1, R \in Obs(x_1), R \neq P}{x_2 \text{ from } R \in POSS(P)}$$

- Submessage Origin Rule:

$$\frac{\{X\}_{k^+} \in POSS(P), X \text{ contains } x_1 \text{ from } Q, \; R \in Observers(k^-), X \text{ contains } x_2 \; R \neq P}{x_2 \text{ from } Q \in POSS(P), x_2 \text{ from } R \in POSS(P)}$$

- Submessage Origin Rule(for public key):

$$\frac{\{X\}_{k_p+} \in POSS(P), X \text{ contains } x_1 \text{ from } Q, \; X \text{ contains } x_2}{x_2 \text{ from } Q \in POSS(P)}$$

- Submessage Origin Rule(for private key):

$$\frac{\{X\}_{k_p-} \in POSS(P), X \text{ contains } x_2}{x_2 \text{ from } Q \in POSS(P)}.$$

5. Result

In this section, the effects of the mechanisms and protocols are suggested and the design processes are demonstrated. All the results are yielded by using the Castalia simulation programme. Every restoration is considered separately and the developed mechanism results are given under separate titles.

5.1. Effect of sensitivity

Both the node quantity and the node sequences are emphasized to determine the effect of sensitivity on the results correctly. In Figure 12. the nodes are distributed randomly, according to the node quantity, and the changes of the rejection of the key by sensitivity are given.

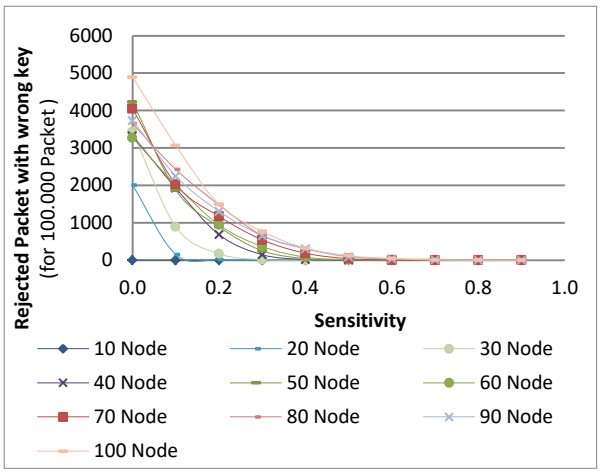

Figure 12. Effect of sensitivity on the nodes which is positioned randomly

Figure 12. clearly indicates that increasing the node quantity causes propagation delays and the selection of sensitivity value becomes important in terms of working of the created system. As the node quantity increases, the rejected package quantity also increases because of the key. For example when sensitivity 0.1 is sufficient for 20 nodes, the sensitivity may be extracted to 0.6 in a system with 100 nodes in order not to reject the keys.

A similar situation is also valid for the case when the nodes do not position randomly but to be at the corners of a grid having a regular geometry (Figure 13.).

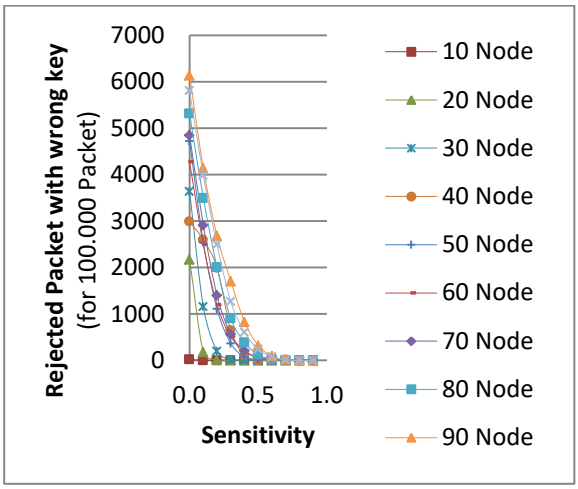

Figure 13. Effect of sensitivity on the nodes which are positioned in a grid shape

When Figure 12. and Figure 13. are evaluated together, it should be noted that the rejected key quantity is bigger when the nodes are positioned on the corners of a regular grid. The reason is that the distances between the target and the source node in the second situation are more than the first one. However, in both cases, the rejected key quantity according to the valence decreases exponentially.

As a result of our experiments under the light of these results given above, the sensitivity is chosen 0.6 for the following experiments.

5.2. Behaviours of the scheme in different communication environment

Castalia simulation programme already has built-in environment parameters. When the communication environment is ideal, all the environment losses are discarded. In this case, the effect of sensitivity according to the number of nodes is shown in Figure 14. for positioning the nodes randomly and in Figure 15. for positioning them on the corners of the grid.

The reason of the decrease in the maximum rejected package quantity in Figure 14. and Figure 15. is that there are no delays of communication environment present in the ideal environment. However, this exponential decrease is so similar to the normal transmission environment shown in Figure 12. where the effect of sensitivity on the nodes is positioned randomly and in Figure 13. where effects of sensitivity on the nodes are positioned in the grid shape.

6. Comparison

In the literature, key management schemes are compared according to several network based criteria. These criteria are shown in Table 1. The introduced scheme and other widely used key management schemes are compared in Table 2.

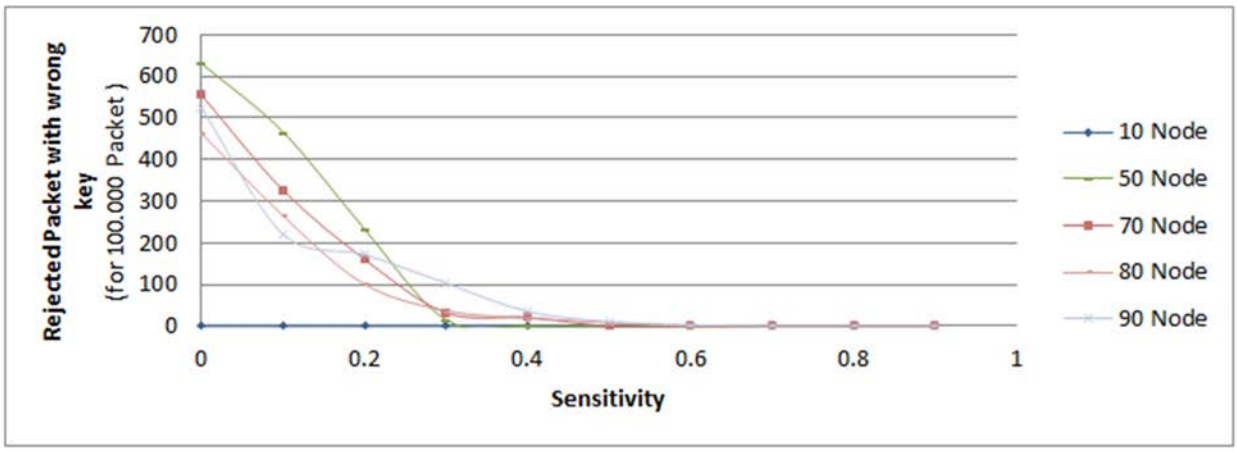

Figure 14. Effect of sensitivity on the nodes positioned randomly in an ideal environment

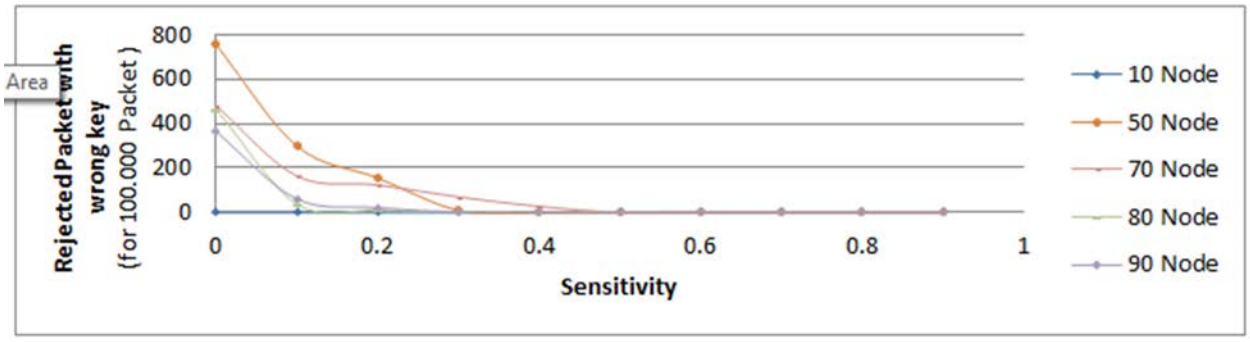

Figure 15. Effect of sensitivity on the nodes positioned on the corners on a grid in an ideal environment

Table 1. Comperation criterias

Memory footprint:	**Mf**	ROM and RAM used for the protocol
Communication overhead:	**Co**	Number of messages exchanged between peers
Processing speed:	**Sp**	Computational cost of the protocol
Network bootstrapping:	**Nb**	Confidentiality of the bootstrap process
Network resilience:	**Nr**	Resistance against stolen credentials
Connectivity (Global conn.):	**G C**	Existence of a key path between any node
Connectivity (Local conn.):	**L C**	Existence of a shared secret between neighbor nodes
Connectivity (Node conn.):	**N C**	Existence of a shared secret between any nodes
Scalability:	**Sc**	Support for big networks
Extensibility:	**Ex**	Capability of adding new nodes
Energy:	**En**	Optimization of the energy usage

Table 2. Comparison of the introduced scheme with widely used key management schemes

Protocol	Co	GC	LC	NC	En	Ex	Mf	Nr	Sc	Nb	Sp
Proposed Scheme	**+1**	**+1**	**+0**	**+0**	**0**	**+1**		**+1**	**+1**		**+1**
OSCAR	+0	+1	+1	+1		+0	+0	+1	+1	+0	
EDDK	+0	+1	+1	+0	-1	+1	-0	+1	+1	+0	-1
CARPY+	+1	+1	+1	+1		+1	-0	+0	+1	+1	
Robust continuity Blom	+0	+1	+1	+1		+1	0	0	+0	+0	-0
SAKE		+1	+1	-1		+1		+1	+1		+1
Multiple Space Blom	+1					-1	0	0	0		
Generalized quadrangle	-1	+1		-1		-1	+1				+1
Symmetric design	+1	+1	+1	+1		-1	+1	-1			+1
Hybrid designs e generalized quadrangle		0	0	0		-0	+1		0		+1
RPB scheme	+1	+1	+1	+1		+1	-0	+0	+1	+1	
Panja		+1	+1	-1		-1	+1	-1	+1		-1
Key infection	-0	+1	+1	-1	+0	-1	+1	+1	+1	-1	+0

Another comparison about computation cost according to the usage of security function. Computational costs are shown below in Table 3.

Table 3.Comparation of related schemes

Schemes	User	Gateway	Sensor node	Total
Das	4 Hash + 1 XOR	7 Hash+1 XOR	1 Hash	11 Hash + 2 XOR
Khan-AlghaHashbar	4 Hash +1 XOR	7 Hash + 1 XOR	2 Hash	13 Hash + 2 XOR
Chen-shih	4 Hash +1 XOR	8 Hash+ 1 XOR	1 Hash	13 Hash + 2 XOR
Vaidya et.al	9 Hash + 7 XOR	10 Hash +9 XOR	3 Hash + 2 XOR	22 Hash + 18 XOR
Xue et.al.	10 Hash	14 Hash	6 Hash	30 Hash
Wong et.al.	-	4 Hash	3 Hash	7 Hash
Watro et.al.	1puKey + 2prKey +1 Hash	1 prKey	2puKey + 1 Hash	6 AsymKey +2 Hash
Proposed Scheme	-	1 prKey+2 Hash +2 XOR	1 puKey+2 Hash+2 XOR	2 AsymKey +4 Hash +4 XOR
	puKey,prKey=asymmetric encryption/decryption			

7. Conclusion

In this proposed scheme, the security problem of wireless sensor networks has been targeted. This paper has introduced Key distribution and Node identification mechanisms, Protocol of authenticating the message identity, Authentication and Sensitivity mechanisms, as part of the novel scheme for wireless sensor networks' security. Results are tested, and discussed. Introduced schemes were compared with similar key management schemes. The scheme provides dynamical node join on network, continuously changing message verifier password.

It was found that these methods which are introduced in this study are superior to other published techniques.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam and E. Cayirci, "Wireless sensor networks: a survey.," *Computer Networks,* vol. 38, no. 4, pp. 393-422, March 2002.

[2] T. He, S. Krishnamurthy, L. Luo, T. Yan, L. Gu, G. Zhou, Q. Cao, P. Vicaire, J. A. Stankovic, T. F. Abdelzaher, J. Hui and B. Krogh, "VigilNet: an integrated sensor network system for energy-efficient surveillance," *ACM Transactions on Sensor Networks,* vol. 2, no. 1, pp. 1-38, February 2006.

[3] Á. Lédeczi, A. Nádas, P. Völgyesi, G. Balogh, B. Kusy, J. Sallai, G. Pap, S. Dóra, K. Molnár, M. Maróti and G. Simon, "Countersniper system for urban warfare," *ACM Transactions on Sensor Networks,* vol. 1, no. 2, p. 153–177, November 2005.

[4] R. Watro, D. Kong, S.-f. Cuti, C. Gardiner, C. Lynn and P. Kruus, "TinyPK: securing sensor networks with public key technology," in *SASN '04 Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks*, New York, 2004.

[5] Z. Benenson, N. Gedicke and O. Raivio, "Realizing Robust User Authentication in Sensor Networks (2005)," in *In Real-World Wireless Sensor Networks REALWSN*, 2005.

[6] H.-L. Yeh, T.-H. Chen, P.-C. Liu, T.-H. Kim and H.-W. Wei, "A Secured Authentication Protocol for Wireless Sensor Networks Using Elliptic Curves Cryptography," in *Sensors 2011*, Basel, Switzerland, 2011.

[7] J. Xu, W.-T. Zhu and D.-G. Feng, "An improved smart card based password authentication scheme with provable security," *Computer Standards & Interfaces,* vol. 31, no. 4, p. 723–728, 2009.

[8] R. Song, "Advanced smart card based password authentication protocol," *Computer Standards & Interfaces,* vol. 32, no. 5–6, p. 321–325, October 2010.

[9] K. H. M. Wong, Y. Zheng, J. Cao and S. Wang, "A dynamic user authentication scheme for wireless sensor networks," in *IEEE International Conference*, Taichung, 2006.

[10] H.-R. Tseng, R.-H. Jan and W. Yang, "An Improved Dynamic User Authentication Scheme for Wireless Sensor Networks," in *IEEE GLOBECOM 2007 - IEEE Global Telecommunications Conference*, Washington, DC, 2007.

[11] T.-H. Lee, "Simple Dynamic User Authentication Protocols for Wireless Sensor Networks," in *Sensor Technologies and Applications, 2008. SENSORCOMM '08. Second International Conference*, Cap Esterel, 2008.

[12] B. Vaidya, J. J. Rodrigues and J. H. Park, "User authentication schemes with pseudonymity for ubiquitous sensor network in NGN," *Int. J. Commun. Syst.,* vol. 23, no. 9-10, pp. 1201-1222, 2010.

[13] M. L. Das, "Two-factor user authentication in wireless sensor networks," *IEEE Transactions on Wireless Communications,* vol. 8, no. 3, pp. 1086 - 1090, 2009.

[14] M. K. Khan and K. Alghathbar, "Cryptanalysis and Security Improvements of 'Two-Factor User Authentication in Wireless Sensor Networks'," *Sensors 2010,* vol. 3, pp. 2450-2459, 2010.

[15] T.-H. Chen and W.-K. Shih, "A Robust Mutual Authentication Protocol for Wireless Sensor Networks," *ETRI Journal,* vol. 32, no. 5, pp. 704-712, 2010.

[16] B. Vaidya, D. Makrakis and H. T. Mouftah, "Improved Two-factor User Authentication in Wireless Sensor Networks," in *2010 IEEE 6th International Conference on Wireless and Mobile Computing, Networking and Communications*, 2010.

[17] K. Xue, C. Ma, P. Hong and R. Ding, "A temporal-credential-based mutual authentication and key agreement scheme for wireless sensor networks," *Journal of Network and Computer Applications,* vol. 36, p. 316–323, 2013.

[18] M. Turkanovic, . B. Brumen and M. Hölbl, "A novel user authentication and key agreement scheme for heterogeneous ad hoc wireless sensor networks, based on the Internet of Things notion," *Ad Hoc Networks,* vol. 20, pp. 96-112, 2014.

[19] C.Alcaraz, J.Lopez, R. Roman and H.-H. C. Chen, "Selecting key Management schemes for WSN," *Computer & Security,* pp. 956-966, 2012 July.

[20] A. D. Rubin and P. Honeyman, "Nonmonotonic Cryptographic Protocols," *Computer Security Foundations Workshop,* 1994.

[21] A. D. Rubin, "Extending NCP for protocols using public keys," *Journal of Mobile Networks and Applications Special issue: protocols for mobile environments,* vol. 2, no. 3, pp. 227-241, 1997.