# Secure Dynamic Access Control Mechanism for Shared Wireless Sensor Networks

D.T.N.I. Perera, Kasun de Zoysa, Jeevani Goonathillake and Asanka Sayakkara
University of Colombo School of Computing,
No 35, Reid Avenue, Colombo 7, Sri Lanka.
email: inoshinidtn@gmail.com, [kasun, jsg, asa]@ucsc.cmb.ac.lk

*Abstract*—**Researchers and organizations from various disciplines are interested in using Wireless Sensor Network (WSN) for their research and applications. However, deploying a sensor network of their own is a difficult task for these communities due to high deployment and maintenance cost. Therefore, the concept of Shared Wireless Sensor Network (SWSN) with multiple base stations is getting popular among these communities. Nevertheless, providing shared access for WSN has given rise to different set of problems such as handling dynamic nature of user privileges, attacks from forged base stations and malicious users. In this paper, we propose a mechanism that can overcome challenges and problems related to access controlling in a SWSN. It uses both symmetric and public key cryptography with an effective key management scheme to ensure the security of the system. Although enforcing this mechanism on a SWSN consumes some energy for communication and processing at the node, it decreases energy for sensing as nodes execute only the queries of authorized users. Since proposed access control scheme is deployed at the node level, each node processes access control individually. Consequently, failure of a node does not cause impact on total access controlling process. We implemented and evaluated this access control mechanism as an enhancement to the TikiriDB data abstraction layer which runs on Contiki development environment.**

***Keywords–Shared wireless sensor networks; user access control; public key cryptography; symmetric cryptography***

## I. Introduction

Wireless Sensor Network (WSN) is a rapidly emerging research area. Most of the researchers and organizations from various disciplines are interested in using WSN for their research and applications to obtain data such as temperature, humidity, pressure and light.

WSNs are composed of large number of tiny sensor devices with wireless communication capabilities in different network connectivity topologies. Sensor devices autonomously form networks through which sensor data is transported. These sensors are highly resource constrained devices with limited processing speed, memory, storage and power. Nowadays, deploying a sensor network individually is not feasible for small business or research groups due to high deployment and maintenance cost of WSNs, authorization issues in deploying own network and accessibility issues with respect to certain sites due to government rules and regulations [1]. As a solution, the concept of Shared Wireless Sensor Network (SWSN) is getting popular among these communities. However, providing shared access for WSN has given rise to a different problem as not all users are allowed to access all the attributes.

Sensor network, base stations and the users are the main components of any SWSN. Base station (BS) is a computer (could also be a smart phone or a PDA) with higher performance which is used to pass user queries to the WSN and to gather results back. There are two categories of users in a SWSN, owners who deploy and maintain the sensor network and the users who requests for sensor data (who are not involved in deployment of the network). Due to security issues, network deployers do not allow users to obtain all types of sensor data attributes from the SWSN. Owners have got privileges to control who will access what data for how long. They may assign privileges on each and every user to obtain data for a particular period of time. Moreover, these privileges for the same user may also vary from time to time depending on user requirements. For example, a forest fire observer who has subscribed/requested temperature may later realize that humidity data should also be required. Hence, it needs to handle this dynamic nature of user privileges carefully to avoid unauthorized access from malicious users, which can be considered as attacks depending on the context.

When providing access control for SWSNs, there are pros and cons according to the topology of the SWSN. For example, access controlling measures of a SWSN with single entry point would be different from the measures considered in a SWSN with multiple entry points. Anyhow, the failure of some of the SWSN nodes should have a limited impact on total access controlling.

In this research paper, we propose a secure dynamic access controlling mechanism for static SWSNs (i.e., a SWSN with non-moving nodes) that can handle, not only the dynamic nature of user privileges of different users, but also issues with forged BSs, access control failure due to single entry point and issues with physical attacks for the sensor motes in the SWSN. This access control mechanism dynamically keeps track of and thus provides security for data in the sensor network from unauthorized/malicious users.

The rest of the paper is organized as follows. In Section II, we discuss the related works. Section III describes the design of the proposed solution. Implementation details are described in Section IV before Section V evaluates various performance aspects. Finally, Section VI provides a summary and conclusions.

## II. Background

Due to the sensor nodes being very resource constrained devices, it is necessary to design operations on those nodes very carefully. A BS cannot be trusted to identify its users correctly to provide network services in WSNs [2]. Moreover, sensors in the WSN could be under the risk of physical attacks which may affect their security. However, this security aspect is beyond the scope of our research.

### A. WSN Scenarios

Four possible scenarios of WSNs are illustrated in Figure 1. These scenarios are formed according to the number of BSs for the WSN and the number of users for a BS.

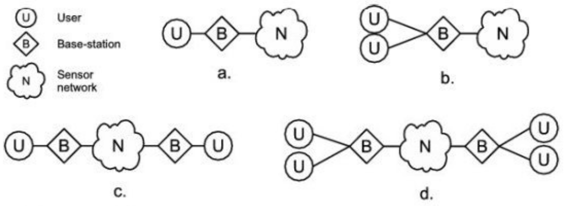


Figure 1. Different scenarios formed by elements of a WSN; (a) Single User with Single Base station (b) Multi Users with Single Base station (c) Multi Base stations and each Base station supporting a Single User (d) Multi Base stations and each Base station supporting Multiple Users.

In this, scenario (a) is a private network and other than a registered user, others cannot access the network to obtain data. In scenario (b), the users can access the network only through this single BS and access controlling for users is required to avoid malicious user access. This type of scenario occurs when a single BS is connected to a pool of user devices such as personal computers [1]. Then, the queries are routed through the common BS, which acts as the gateway to the WSN. According to scenario (c), each user is registered to a specific BS and there is one user per BS. In scenario (d), there are multiple BSs registered/deployed by the owners of the WSN and users should be able to access the WSN via any registered BS available in the network. Due to the sharing nature of SWSNs, users may also access the network directly using their own PCs, laptops or mobile phones which are known as non-registered BSs. This situation may give rise to many security threats such as attacks from forged BSs [3]. The multiple BSs in a SWSN may not be connected to each other [1][4].

### B. Dynamic Nature of User Privileges

Different users of the SWSN request different sensor attributes from the network. The user access privileges for the SWSN given by the owners may vary from one user to another. The privileges for the same user may also vary from time to time. This can be introduced as the dynamic nature of user privileges which is required to be managed for SWSNs.

Consequently, users must be able to obtain required sensor data from SWSN based on the privileges assigned to them. To this end, it is necessary to consider both sensor data types each user can obtain and for how long a user can execute a query to obtain such data as privileges with respect to that user may vary with the time. SWSN does not have the capacity to store the authorization data of every user. As such, these privileges are to be maintained by another trusted party and such data must be released in a secure manner to the SWSN when necessary.

### C. User Access Controlling in WSNs

TinyPK and Kerberos server authentication schemes are well-known protocols which can be used to control the user access in WSNs. TinyPK is a public-key based protocol, which allows authentication and key agreement between a WSN and a third party as well as two WSNs [5]. However, TinyPK is implemented using the TinyOS development environment which does not support SWSNs with multiple BSs. Hence, TinyPK cannot be used to control the user access over SWSNs and also it is very inefficient to use on low-power devices as implementation is based upon public key algorithms, such as RSA [1].

Kerberos is a network authentication protocol which is implemented using symmetric key cryptography [6]. There are two main components for Kerberos server namely Authentication Server(AS) and Ticket Granting Server(TGS). In this protocol, it requires continuous availability of a central server (both AS and TGS). If this server fails, then no one can log in. According to the experts, this problem can be mitigated by using more than one central server and falling backing authentication mechanisms [6]. Kerberos authentication scheme performs BS level user authentication process [2]. As such it is not possible to authenticate users who connect to the SWSN using their own computers or mobile phones(which are non-registered BSs to the network). Provision of access controlling at the node level would avoid forged BS attacks and consequently Kerberos is not suitable to safe guard the network from forged BS attacks by malicious users. Moreover, if the Kerberos Server Authentication Scheme is deployed at the node level, protecting the secrecy of symmetric keys is not possible because WSNs are typically deployed in outdoor environments and the nodes are highly susceptible to physical attacks.

Some of the researchers have proposed solutions to control access over WSNs. Zhang et al.[7] proposed a mechanism to restrict and revoke access privilege of WSN, which has multiple BSs, through establishing a secret key between the BS and the sensor node. Haodong et al. have proposed a scheme based upon public key authentication for access controlling of WSNs using access control list. However, this scheme requires the user to be authenticated twice to get all data even if the two sensors are very close to each other while the design of the more efficient scheme that requires only one authentication [8]. However, none of these access controlling mechanisms proposed for WSN have neither discussed their adaptability for authentication and authorization with respect to the SWSN with multiple base stations nor their capability of handling dynamically evolving user privileges. When considering the access controlling of SWSN, message authentication has been considered as an important security component. Wang et al. [8] have proposed a public key based approach and allow sensors to authenticate the broadcast messages in a distributed way. Using public key approach at node level incurs an overhead cost and in addition to that it lacks the capability of managing dynamic user privileges.

Jef Maerien et al.[9] have proposed a middleware solution, which enables secure multi-party interactions on top of resource constrained sensor nodes. However, this does not handle dynamic nature of user privileges which is our main objective in this research and instead they provide a mechanism to secure SWSNs from malicious applications.

### D. Access controlling Mechanism for SWSN

Based on previous access controlling mechanisms and their limitations we have identified some main features that must be incorporated into access control mechanism of SWSNs. They are : ability to handle dynamic nature of user privileges at the node level to facilitate scalability in order to prevent any kind of node failures of the SWSN from causing negative impact on access controlling, no use of secret keys which are stored inside the sensor motes permanently to ensure secrecy of the secret keys and ability to control the overhead of sensor mote. From overhead point of view public key algorithms consume more energy and comparatively symmetric key cryptographic algorithms and hash functions consume much less computational energy[8]. As such, the consumption of computational power can be reduced by using symmetric key cryptographic algorithms for access controlling mechanisms over WSNs.

## III. Design

In this research, we consider data-wise access controlling mechanism which makes limitations on accessible data attributes from the WSN for each user taking into consideration the dynamic nature of user privileges.

### A. System Infrastructure

This mechanism requires a combination of public-key infrastructure and symmetric cryptography. Hence, this system can be considered as a hybrid mechanism of both TinyPK protocol (with public-key scheme) and Kerberos Server Authentication mechanism (with symmetric key scheme). As such the newly introduced Hybrid Authority (HA), SWSN and users are the three main components of our proposed access controlling mechanism.

HA is a trusted third party component which is formed by the combination of both Certificate Authority (CA) and Attribute Authority (AA). There is one HA for a SWSN and HA has its own public and private key pair. The CA is an entity which has a private key and a public key that is trusted by external parties to create and sign Public key Certificates. The AA is an entity trusted by one or more external parties to create and sign Attribute Certificates. In this project AA keeps track of which sensor data attributes and when particular users can access these attributes.

The SWSN may consist of small number of tiny sensor devices and each sensor node is to be deployed with the public key of the relevant HA. The network may consist of one or more BSs and the users can access the network using the registered or non-registered BSs.

The user who is trying to access the network, first need to register in the HA of the SWSN and needs to obtain Public key Certificate (PKC), Attribute Certificate (AC) and the corresponding key pair (public and private key). A user cannot obtain sensor data, if he is not registered in the relevant HA of the SWSN. The PKC can be considered to be like a passport: it identifies the holder, tends to last for a long time, and should not be trivial to obtain. The AC is more like an entry visa: it is typically issued by a different authority and does not last for a long time. So the PKC is used for user authentication and AC is used for authorization. Figure 2 illustrates the architecture of the proposed Secure Dynamic Access Control Mechanism.

### B. System Operations to Obtain Sensor Data

There are three steps in this mechanism to obtain data after the user registered with HA.

*1) Step 1-Initializing the Communication:*
First, the user sends a “Hello” message to any sensor node to start communication. Then this recipient node which is known as the root node generates a session key(SK1) and broadcasts it over the SWSN. It then encrypts SK1 using the public key of the HA and sends the encrypted SK1 along with the URL of HA and the expiring time stamp of SK1(SK1 is valid for a certain period which would be decided at the implementation phase) back to the user.

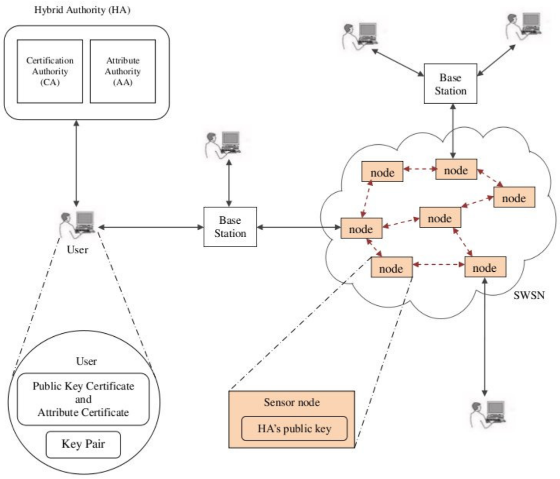


Figure 2. Architecture of Secure Dynamic Access Control Mechanism.

*2) Step 2-User Authentication at the HA:*
After the completion of step 1, user creates a request for a ticket from the HA by including his details and a time stamp. Then he sends both request and the encrypted SK1 to the HA using the URL.

At the HA, it checks for the user in its database and if it is a registered user, HA creates a new session key(SK2) and a ticket by including user details, user address, date, validity, SK2 and sensor data attribute details which the user can access from the SWSN along with the time period for each attribute. Then, HA obtains SK1 using its private key and encrypts the ticket using SK1. After that, it encrypts the SK2 using user’s public key. HA sends both the encrypted ticket and newly created encrypted SK2 to the user.

*3) Step 3-User Authentication and Authorization at SWSN:*
After the completion of step 2, the user has the ticket from the HA. Firstly, he decrypts SK2 using his private key. Then, he creates a query to obtain required details from the network and encrypts it using SK2. The user sends both the encrypted ticket and the encrypted query with a time stamp to the node to obtain data.

At the node, it first decrypts the ticket using SK1. Then it checks the validity of the ticket. If the ticket is not expired, then the node obtains SK2 from the ticket and decrypts the query. After this step the authentication process is completed and the node is able to identify that the ticket is generated by the HA for this user. Then, it starts the authorization process. For this, node checks the requested attributes in the query against the privilege granted attributes in the ticket in order to eliminate the unauthorized attributes if there are any in the query. It then calculates the end of valid duration (i.e., cutoff time) for each authorized attribute requested in the query based upon the ticket generated time. After that, node starts sensing each requested (authorised) attribute till the current time is less than the attribute cutoff time. Then at the completion of the sensing process each node encrypts the results using SK2 (SK2 is in the ticket) and sends them back to the user through the initial routing path. User obtains the encrypted results which he is able to decrypt by using SK2.

By making CA and AA as one physical unit called HA, it is possible to reduce the number of keys stored in sensor nodes to one. Moreover, it will reduce the overhead at the node since it uses symmetric encryption algorithm at the node side. In this design we assume that there are encrypted channels between user and HA, user and SWSN.

## IV. Implementation

We use TikiriDB data abstraction layer to implement this access controlling mechanism. It runs on ContikiOS development environment. At the execution of TikiriDB, when queries reach at a BS, they are parsed and check for errors. If there are no errors in the query then it is preprocessed to convert into a mote readable format. After that, they are sent to the network for execution. The BSs in TikiriDB are sensor motes with different behaviours than other motes in the network. This is due to the fact that motes that work as BSs are not going to do sensing but they just forward queries to the other motes and gather results back as BSs.

In this project, when the user tries to obtain data from the SWSN it needs to travel through the TikiriDB data abstraction layer twice. The first one is transmitting the “Hello” message to the network and the second one is transmitting user query and ticket from the HA to the network. The components of TikiriDB and the functionality related to this mechanism are described from Section IV-A to IV-E.

### A. Lexical Analyzer

Lexical analyzer is the first component in the process of executing the queries in TikiriDB. It identifies defined tokens in the acquisitional query and sends them to the parser. TikiriDB uses “flex” the lex tool to tokenize input stream.

### B. Parser

Parser receives the stream of tokens generated by lexical analyzer and it evaluates the semantic meaning of those tokens.

### C. Query Processor

The data fetched from parser is processed in this module and the output is a data packet. Generally, the packet is a byte array which contains the query data. The default packet size is 128 bits. In this project, query processor generates two types of data packets. They are “Hello” packet and packet with query and ticket. The maximum size of encrypted ticket is 80 bits. So, the size of this packet is 208 bits (128 bits + 80 bits). The structure of this data packet is illustrated in Figure 3. According to current implementations, TikiriDB allows to transmit maximum 200 bytes of size data packet through the network. Therefore, it can transmit this 26(208/8) bytes packet easily over the WSN.

### D. Serial Forwarder

This module works as the root node of the SWSN and the existing serial forwarder of TikiriDB is enhanced as described in this Section. When the serial forwarder receives the “Hello” data packet from query processor, it generates a session key(SK1) which is a 128 bits AES key. Then, it broadcasts SK1 through the network using its routing mechanism. The purpose of this SK1 broadcasting is to keep the generated session key in the network till the end of its time stamp even if the root node fails. Then, SK1 is encrypted using HA’s public key (1024 bits size) by using RSA public key encryption algorithm [10]. After that, this module sends encrypted SK1 with URL of the HA and time stamp of the SK1 to client.

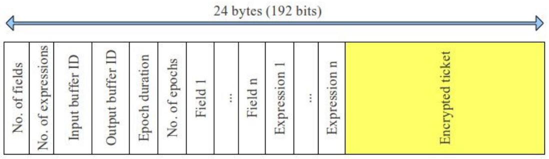


Figure 3. Data packet structure with both query and encrypted ticket.

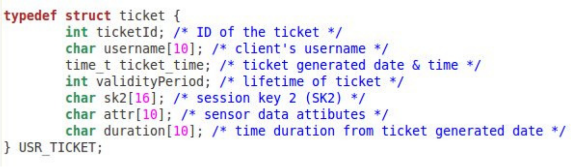
When the serial forwarder receives the data packet with query and ticket, it broadcasts that to the network. After completion of the sensing process, this module obtains the encrypted results from the other nodes and it sends them back to the client.

### E. Execution at node

When a node receives a query as a bit stream of data it is stored in a data structure. The node validates received query to discard invalid or corrupted queries. The parsed query is set into a function named “ctimer set” with a time which uses the epoch duration that the user has provided with. Then, that function call executes the “execute select query()” function periodically.

When a node receives the bit stream of SK1, it stores that until its valid period. The node will discard each session key after the end of the corresponding time stamp. On receipt of the bit stream of encrypted ticket and query, node works as described in Section III. The whole process inside the sensor node is depicted in the following pseudo code.

```
parse query(){
    extract data from query
    extract ticket from query
    if( ticket is valid)
        execute query() once epoch duration
    else
        send ticket invalid response to user
}

execute query(){
    if( current epoch < for−period){
        while( all the attributes are sensed){
            if( current time < cutoff time){
                sense current attribute
            } else{
                add N/A as attribute result
            }
        }
    } else{
        create query result ()
        send query result ()
        free memory
    }
    current epoch++
}
```



```
typedef struct ticket {
        int ticketId; /* ID of the ticket */
        char username[10]; /* client's username */
        time_t ticket_time; /* ticket generated date & time */
        int validityPeriod; /* lifetime of ticket */
        char sk2[16]; /* session key 2 (SK2) */
        char attr[10]; /* sensor data attibutes */
        char duration[10]; /* time duration from ticket generated date */
} USR_TICKET;
```

Figure 4. Ticket Structure.

### F. Execution at User

The functionality of this is implemented inside TikiriDB/gateway/tikirisql folder as described in Section III. It uses RSA decryption algorithm and AES encryption algorithm in openssl library at the user side.

### G. Execution at HA

The HA is a separate module from TikiriDB. It connects with a conventional database to maintain user details. In this research, HA runs on IP 127.0.0.1 and port 8080. HA also uses RSA and ASE algorithms for its operations. The structure of the ticket which is generated inside HA is illustrated in Figure 4.

## V. Evaluation

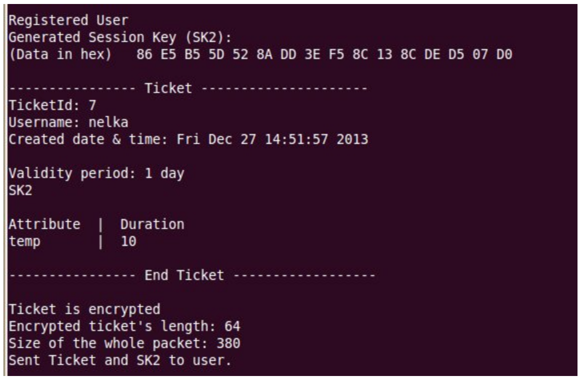
In this project, we used COOJA simulator as the simulating platform. COOJA is a cross level simulator for Contiki OS which can simultaneously simulate networking layer, operating system layer and the instruction layer.

### A. Results of Query execution

According to the objectives of this research, it is necessary to check both authentication and authorization of each and every user at the time they send queries to the SWSN to obtain data. The authentication means establishing a relation between the user and some identity. In this research, the ticket issued by the HA is the identity for users to obtain data from the SWSN. The authorization means establishing a relation between a user and a set of privileges such as, what types of sensor data attributes that a particular user can obtain and for how long the user is able to execute the query to obtain those data.

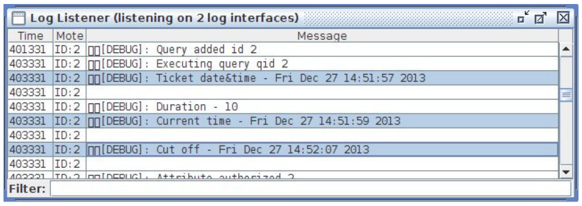
In this Section, we observe the results of implemented mechanism for SWSN under following scenarios.

- Scenario 1: A user who doesn't register in the HA sends a query to access data.
- Scenario 2: A registered user who can access temperature for 10s sends a query to obtain it for 10s.
- Scenario 3: A registered user who can access temperature for 10s sends a query to obtain it for 8s.
- Scenario 4: A registered user who can access temperature for 5s sends a query to obtain it for 10s.
- Scenario 5: A registered user who can access only temperature for 10s sends a query to obtain both temperature and humidity for 10s.
- Scenario 6: A registered user who can access both temperature and humidity for different durations sends query to access both attributes.



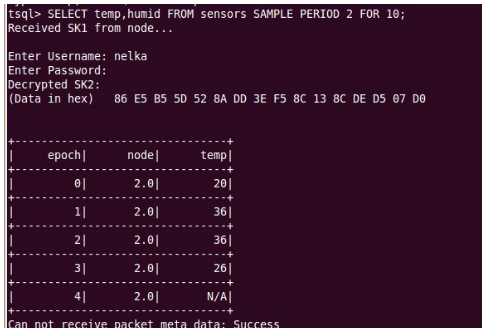
```
Registered User
Generated Session Key (SK2):
(Data in hex)   86 E5 B5 5D 52 8A DD 3E F5 8C 13 8C DE D5 07 D0

--------------- Ticket --------------------
TicketId: 7
Username: nelka
Created date & time: Fri Dec 27 14:51:57 2013

Validity period: 1 day
SK2

Attribute  |  Duration
temp       |  10

--------------- End Ticket -----------------

Ticket is encrypted
Encrypted ticket's length: 64
Size of the whole packet: 380
Sent Ticket and SK2 to user.
```

Figure 5. Generated ticket at the HA for Scenario 5.



```
Time   Mote  Message
401331 ID:2  [DEBUG]: Query added id 2
403331 ID:2  [DEBUG]: Executing query qid 2
403331 ID:2  [DEBUG]: Ticket date&time - Fri Dec 27 14:51:57 2013
403331 ID:2
403331 ID:2  [DEBUG]: Duration - 10
403331 ID:2  [DEBUG]: Current time - Fri Dec 27 14:51:59 2013
403331 ID:2
403331 ID:2  [DEBUG]: Cut off - Fri Dec 27 14:52:07 2013
403331 ID:2
403331 ID:2  [DEBUG]: Attribute authorized 2
```

Figure 6. Time details for temperature attribute in Log Listener of the COOJA simulator for Scenario 5.



```
tsql> SELECT temp,humid FROM sensors SAMPLE PERIOD 2 FOR 10;
Received SK1 from node...

Enter Username: nelka
Enter Password:
Decrypted SK2:
(Data in hex)   86 E5 B5 5D 52 8A DD 3E F5 8C 13 8C DE D5 07 D0


+--------------------------------+
|     epoch|      node|      temp|
+--------------------------------+
|         0|       2.0|        20|
+--------------------------------+
|         1|       2.0|        36|
+--------------------------------+
|         2|       2.0|        36|
+--------------------------------+
|         3|       2.0|        26|
+--------------------------------+
|         4|       2.0|       N/A|
+--------------------------------+
Can not receive packet meta data: Success
```

Figure 7. Response to the user for the query in Scenario 5.

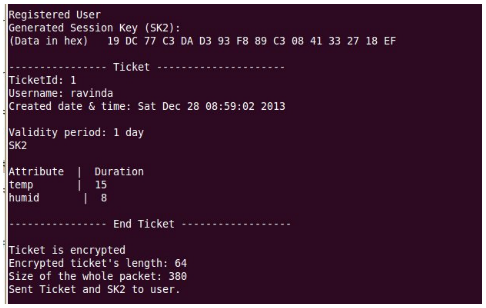
- Scenario 7: A registered user sends a query to obtain data with invalid(expired) ticket from HA.

For the evaluation purpose, we use time duration for each attributes in seconds. For this paper, we have selected 2 scenarios (5 & 6) to illustrate the results and those are depicted from Figures 5- 10. In Scenario 1, HA doesn't send a ticket to the user but it sends the message “You aren't a registered user”. In Scenario 7, user doesn't receive any sensor data but he receives the message “The ticket is expired”.

### B. Performance Analysis

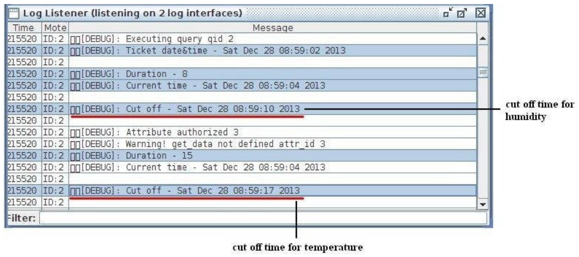
In this Section, we measure the execution times for main operations of the proposed mechanism using time.h library. To calculate these execution times, we executed this system on Intel(R) Core(TM)2 Duo dual core computer with 2GB RAM.

According to the original version of TikiriDB (the version before adding this dynamic access control mechanism), there is an initial delay of 19ms for retrieving results after entering the query to the tikirisql query interface.



```
Registered User
Generated Session Key (SK2):
(Data in hex)   19 DC 77 C3 DA D3 93 F8 89 C3 08 41 33 27 18 EF

---------------- Ticket ---------------------
TicketId: 1
Username: ravinda
Created date & time: Sat Dec 28 08:59:02 2013

Validity period: 1 day
SK2

Attribute  |  Duration
temp       |  15
humid       |  8

---------------- End Ticket ------------------

Ticket is encrypted
Encrypted ticket's length: 64
Size of the whole packet: 380
Sent Ticket and SK2 to user.
```

Figure 8. Generated ticket at the HA for Scenario 6.

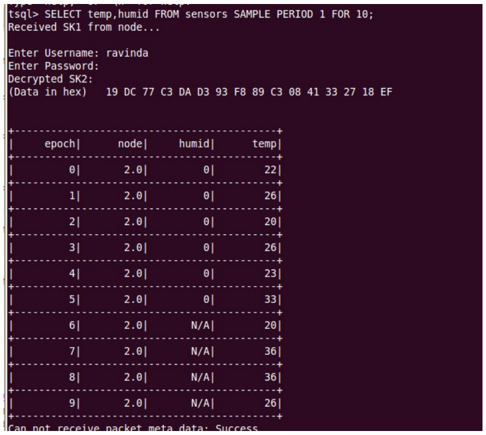


```
Time   Mote  Message
215520 ID:2 [DEBUG]: Executing query qid 2
215520 ID:2 [DEBUG]: Ticket date&time - Sat Dec 28 08:59:02 2013
215520 ID:2
215520 ID:2 [DEBUG]: Duration - 8
215520 ID:2 [DEBUG]: Current time - Sat Dec 28 08:59:04 2013
215520 ID:2
215520 ID:2 [DEBUG]: Cut off - Sat Dec 28 08:59:10 2013
215520 ID:2
215520 ID:2 [DEBUG]: Attribute authorized 3
215520 ID:2 [DEBUG]: Warning! get_data not defined attr_id 3
215520 ID:2 [DEBUG]: Duration - 15
215520 ID:2 [DEBUG]: Current time - Sat Dec 28 08:59:04 2013
215520 ID:2
215520 ID:2 [DEBUG]: Cut off - Sat Dec 28 08:59:17 2013
215520 ID:2
```

cut off time for humidity

cut off time for temperature

Figure 9. Time details for temperature and humidity attributes in Log Listener of the COOJA simulator for Scenario 6.



```
tsql> SELECT temp,humid FROM sensors SAMPLE PERIOD 1 FOR 10;
Received SK1 from node...

Enter Username: ravinda
Enter Password:
Decrypted SK2:
(Data in hex)   19 DC 77 C3 DA D3 93 F8 89 C3 08 41 33 27 18 EF


+----------------------------------------+
|     epoch|      node|     humid|      temp|
+----------------------------------------+
|         0|       2.0|         0|        22|
+----------------------------------------+
|         1|       2.0|         0|        26|
+----------------------------------------+
|         2|       2.0|         0|        20|
+----------------------------------------+
|         3|       2.0|         0|        26|
+----------------------------------------+
|         4|       2.0|         0|        23|
+----------------------------------------+
|         5|       2.0|         0|        33|
+----------------------------------------+
|         6|       2.0|       N/A|        20|
+----------------------------------------+
|         7|       2.0|       N/A|        36|
+----------------------------------------+
|         8|       2.0|       N/A|        36|
+----------------------------------------+
|         9|       2.0|       N/A|        26|
+----------------------------------------+
Can not receive packet meta data: Success
```

Figure 10. Response to the user for the query in Scenario 6.

According to our observations, with the enhancement of this dynamic access controlling mechanism, the delay of retrieving results of a query is about 6171.98 ms That means, it takes about 6171.98 ms for user authentication and authorization process with an additional delay of 6152.98 (6171.98 - 19)ms due to this access controlling scheme. Moreover, this delay is not a fixed delay value and it depends on the execution times of each operation at the HA side, user side, network side and communication delays between user, HA and network. According to this access controlling mechanism, the main operations at the HA side are SK2(128 bits) generation, RSA encryption of SK2 with 1024 bits key, ticket generation and AES encrypt of ticket with 128 bits symmetric key. At the user side RSA decryption of SK2, query generation and AES encryption of query using SK2 are the main operations. At the network side, it performs SK1(128 bits) generation, RSA encryption with 1024 bits key, 2 AES decryptions using 128 bits symmetric key and simple table generation inside the node before starting sensing process.

Although various encryptions and decryptions occur in HA side and user side, the operations within the network (means inside the nodes) directly and significantly affects the performance of the system, because the sensor network is the most resource constrained environment.

The nodes in the network has to generate 128 bits key (SK1) and encrypt it using RSA encryption algorithm with 1024 bits key for all of the scenarios that is mentioned in Section V-A. Moreover, when the network receives the encrypted ticket and query, it is necessary to perform AES decryption twice on those. According to this mechanism, every RSA encryption is done for fixed sized data(128 bits). As well as, although the size of the ticket and the query is varying according to the user and scenario, the size of the encrypted ticket and the size of the encrypted query is fixed. As a result, the total execution times for each of these decryptions may be the same for each scenario. However, we have obtained the execution times for main operations with respect to all entities. Table I contains the mean execution time according to the scenarios in Section V-A.

TABLE I. MEAN EXECUTION TIMES FOR MAIN OPERATIONS IN PROPOSED DYNAMIC ACCESS CONTROL MECHANISM.

| *Operation* | *Mean execution time (ms)* |
|---|---|
| 128bits key generation | 497.57 |
| RSA encryption with 1024bits key | 849.28 |
| RSA decryption with 1024bits key | 1587.65 |
| AES encryption with 128bits symmetric key | 51.33 |
| AES decryption with 128bits symmetric key | 87.16 |

According to these mean values, RSA decryption takes more time to execute (which is occurring at HA side and user side). The execution times of RSA and AES encryptions and decryptions are little bit high because of the openssl libraries. Generally, openssl libraries are heavy to use on sensor motes in a WSN but it supports high levels of security [11]. However, when we compare these RSA execution times with TinyPK protocol, it requires average time of 14.5s for RSA operations with 1024bit key size in TinyPK [5]. Hence, the delay for its authentication process also become very high than our solution. Since we have tried to implement a prototype of this access controlling mechanism in this research, a simulator is used to run this system. To this end, we opted to use openssl libraries for the implementation. An encryption using the OpenSSL implementation of RSA algorithm is performed only once at a node while all the other encryption and decryption tasks are performed using lightweight AES algorithm. Therefore, the overhead of heavy RSA algorithm has a very lower effect on the performance of the network.

### C. Energy consumption

According to this mechanism, in addition to communication and sensing some energy consumption in node is also used for authentication process. It generates additional data packets before the query result data for all registered users' requests. These data packets are namely "Hello" data packet, "Hello" reply data packet, SK1 broadcasting packet, Ticket request, data packet with encrypted ticket and query request. Moreover, the number of SK1 broadcasting packets and data packets with encrypted ticket and query depend on the number of nodes in the SWSN since these packets are to be transmitted to the whole network.

Conceptually, it is understood that the suggested access controlling mechanism increases the number of data packet transmissions and hence the energy required for communication in SWSN with this mechanism would be higher than a SWSN without this mechanism.

On the other hand, the enforcement of the proposed access controlling mechanism would prevent some users from obtaining results thus reducing sensing cost and results transmission cost. As a result, energy consumption for sensing and thus result transmission would be comparatively lower in a SWSN with this mechanism.

### *D. Node failure recovery*

Generally, sensor nodes in the SWSN have high frequency to fail than BSs. According to the design of this mechanism, the user authentication and authorization have been distributed to be handled individually by the nodes themselves. Initially, the root node generates SK1 and then it broadcasts that key to the SWSN. As a result, SK1 is in the network although the root node fails and the new node which works as the root node now also has the key.

Moreover, since the ticket is obtained by all the nodes in the SWSN even if the root node fails at the completion of sensing process, SK2 is there inside all nodes. This enables each node to encrypt results individually using SK2 before sending them to the user. Since each node has the ability to process individually, failure of other nodes in the network does not affect the overall access controlling mechanism thus causing node failure recovery.

## VI. Conclusion

Nowadays, access controlling of SWSNs is a big challenge particularly to manage dynamic access privileges of users. Existing access control mechanisms for sensor networks are not suitable to be used on SWSNs for this purpose as explained in the above Sections. Therefore, in this paper, we have proposed a secure access controlling mechanism which is a flexible solution to overcome user access controlling issues in static SWSNs through handling dynamic nature of user privileges at the node level.

The system architecture of the proposed mechanism consists with a HA, user and SWSN. It uses both symmetric encryption and public key encryption to guarantee the security. We have implemented this system, as a module to the TikiriDB data abstraction layer which runs on ContikiOS and for the simulation purpose we used COOJA simulator.

We tried to make a comparison in a SWSN with and without this authentication scheme under several points such as time to retrieve results and energy consumption. We got the execution time for each operation inside HA, user and network. Based on these results, it spends more time for RSA decryption. At the node, more time is spent for RSA encryption which happens only once for each user request. With this access controlling scheme, SWSN needs to consume energy for communication, sensing and authentication process at the node. Moreover, since this mechanism handles access controlling at the node level, it provides node failure recovery as an additional advantage. As a result, the failure of sensor nodes in the SWSN does not have an impact on total access controlling scheme.

The presence of a malicious node within the network can reveal the session key to an attacker since the session key is distributed by broadcasting. Therefore, it is highly necessary to protect the network from malicious nodes. Various previous research has been conducted to mitigate the threat of malicious nodes which should be incorporated with our solution in the future [12].

We handle access control on SWSN not only dynamically but also flexibly as different users can be limited to access different sensor data for different time durations at the discretion of the network owner. Access control is tightened further by enforcing an authentication mechanism to prevent any authorized or unauthorized user from illegitimately obtaining sensor data of another user when transmitted through the network. The unavailability of such an access control mechanism for SWSN in the literature makes our contribution significant. While any authenticated packet transmission on SWSN would increase the execution time, there was no significant increase in our authentication mechanism as compared to TinyPK [5].

## References

[1] N. M. Laxaman, M. D. J. S. Goonatillake, and K. D. Zoysa, “Tikiridb: Shared wireless sensor network database for multi-user data access,” *The Computer Society of Sri Lanka (CSSL)*, pp. 26–32, August 2010.

[2] Q. Siddique, “Kerberos authentication in wireless sensor networks,” *Annual University Tibiscus Computer Science Series*, vol. 8, pp. 67–80, 2010.

[3] Y. Faye, I. Niang, and T. Noel, “A survey of access control schemes in wireless sensor networks,” *Proc. World Acad. Sci. Eng. Tech*, vol. 59, pp. 814–823, 2011.

[4] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, “Tinydb: an acquisitional query processing system for sensor networks,” *ACM Transactions on database systems (TODS)*, vol. 30, no. 1, pp. 122–173, 2005.

[5] R. Watro, D. Kong, S.-f. Cuti, C. Gardiner, C. Lynn, and 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, 2004, pp. 59–64.

[6] “How Kerberos Works,” 2010, URL: http://mccltd.net/blog/?p=1053 [accessed: 14-October-2015].

[7] W. Zhang, H. Song, S. Zhu, and G. Cao, “Least privilege and privilege deprivation: towards tolerating mobile sink compromises in wireless sensor networks,” in *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*. ACM, 2005, pp. 378–389.

[8] H. Wang and Q. Li, “Achieving distributed user access control in sensor networks,” *Ad Hoc Networks*, vol. 10, no. 3, pp. 272–283, 2012.

[9] J. Maerien, S. Michiels, D. Hughes, C. Huygens, and W. Joosen, “Seclooci: A comprehensive security middleware architecture for shared wireless sensor networks ad hoc networks,” vol. 25, Part A, February 2015, pp. 141–169.

[10] G. Singh and Supriya, “A study of encryption algorithms (rsa, des, 3des and aes) for information security,” vol. 67, no. 19, April 2013, pp. 33–38.

[11] G. Hatzivasilis, A. Theodoridis, E. Gasparis, and C. Manifavas, “Ulcl: An ultra-lightweight cryptographic library for embedded systems,” in *Proceedings of the 4th International Conference on Pervasive and Embedded Computing and Communication Systems*, January 2014, pp. 247–254, ISBN: 978-989-758-000-0.

[12] W. R. Pires, T. H. de Paula Figueiredo, H. C. Wong, and A. Loureiro, “Malicious node detection in wireless sensor networks,” in *Parallel and Distributed Processing Symposium, 2004. Proceedings. 18th International*. IEEE, 2004, p. 24.