Hardware Implementation of Reliable Network Recovery from Base Station Failure of Surveillance WSN

M. H. Megahed
Communications Department, Military Technical college, Cairo, Egypt

Khaled H. Moustafa
Communications Department, Military Technical college, Cairo, Egypt

Abstract
In this paper, the hardware implementation of the first component of SurvSec security architecture, which is the reliable network recovery from base station failure, is presented. The implementation of reliable network recovery from base station failure was implemented on Arduino Uno microcontroller boards, the used transceivers are X-Bee modules 1 mw series 1 and the used motion detection sensor is X-Band Doppler radar motion detection sensor. The X-Bee transceiver cannot be connected directly to the Arduino Uno microcontroller board; therefore an X-Bee shield card is used to connect between the X-Bee transceiver and the Arduino Uno board. The motion detection sensor is connected to the Arduino Uno board. AES encryption algorithm is implemented on Arduino Uno microcontroller board to encrypt the security reports that is sent from the sensor nodes to the security manager and from the security managers to the new base station. The reliable network recovery from base station failure is hardware implemented to show its validity in real time implementation. Power consumption of the sent and received security report was measured to show that the reliable network recovery from base station failure has low power consumption.

1. Introduction

To the best of our knowledge, there is no contribution in the open literature addressing the situation a user has to deal with from the time the Base Station (BS) fails to the time the BS is operational again. Also, we haven't found any research explaining how the new BS can verify the trustworthiness of the existing sensor nodes except our work at [1]. By lacking the ability to verify the trustworthiness of the existing sensor nodes, a user has no choice but to "scrape" the existing deployment and proceed with a new one, despite the deficiencies associated with this choice (e.g. high cost and long duration of having unreliable coverage of the deployed WSN).

Our work is addressing this important issue to provide practical answers to this challenging problem. Based on our work, we propose a new security architecture called SurvSec. SurvSec is capable of maintaining security information even during the BS failure periods. This is accomplished in two steps. The first step, storing the security related data until the recovery of the BS or the deployment of new BS. The second step, sending the stored data to the recovered BS or the new BS after it is authenticated. The **objectives of the hardware implementation** are first, the low power consumption of transmitted and received security report which is measured and with calculation the battery can work for one year and second, the validity of transmitter and receiver code for reliable network recovery from base station failure. Furthermore, BS failure shows the importance of continuous storage of the security reports of the monitored security threats towards the WSN through securely storing the security related data of sensor nodes.

This sub-section describes the steps for the hardware implementation:

1- Hardware design of reliable network recovery from base BS is introduced.
2- Programming of the X-Bee Transceivers [6] is done using the X-Bee programmer board [7].
3- Writing down the code of the transmitter and the code of the receiver on the Arduino Uno microcontroller boards is done.
4- Writing down the code of the AES encryption algorithm on Arduino Uno microcontroller boards is done. The AES is used to encrypt and decrypt the sent security reports from the sensor nodes to the security managers (SMs) and to encrypt and decrypt the sent security reports from the security managers to the new BS.
5- Writing down the code of the motion detection sensor is done.
6- Hardware implementation for reliable network recovery from BS failure is done using Arduino Uno microcontroller boards [8], X-Bee shields [9], X-Bee transceivers, and X-Band motion detection sensors [10].
7- Debugging the errors on the code is done to achieve correct code for hardware implementation of reliable network recovery from base station failure.
8- Hardware simulation of the proposed system on Arduino Uno simulator is done.
9- Measurements of the passing current on Arduino Uno board and the power consumption of the sent and received security report is done.
10- Serial monitor software is used to monitor transmitted and received data.

In this section, an introduction to hardware implementation for reliable network recovery from BS failure is introduced. Section 2 presents the related work, and the proposed system components. Section 3 presents the design and implementation of the proposed system. Section 4 presents the results and the evaluation metrics. Section 5 presents the comparison between our work and previous works. Finally, section 6 presents the summary. Figure 1 shows the block diagram of the proposed system. The motion detection sensor is connected to the Arduino Uno microcontroller board. The X-Bee transceiver is connected to the shield card. Shield card is connected to the microcontroller board.

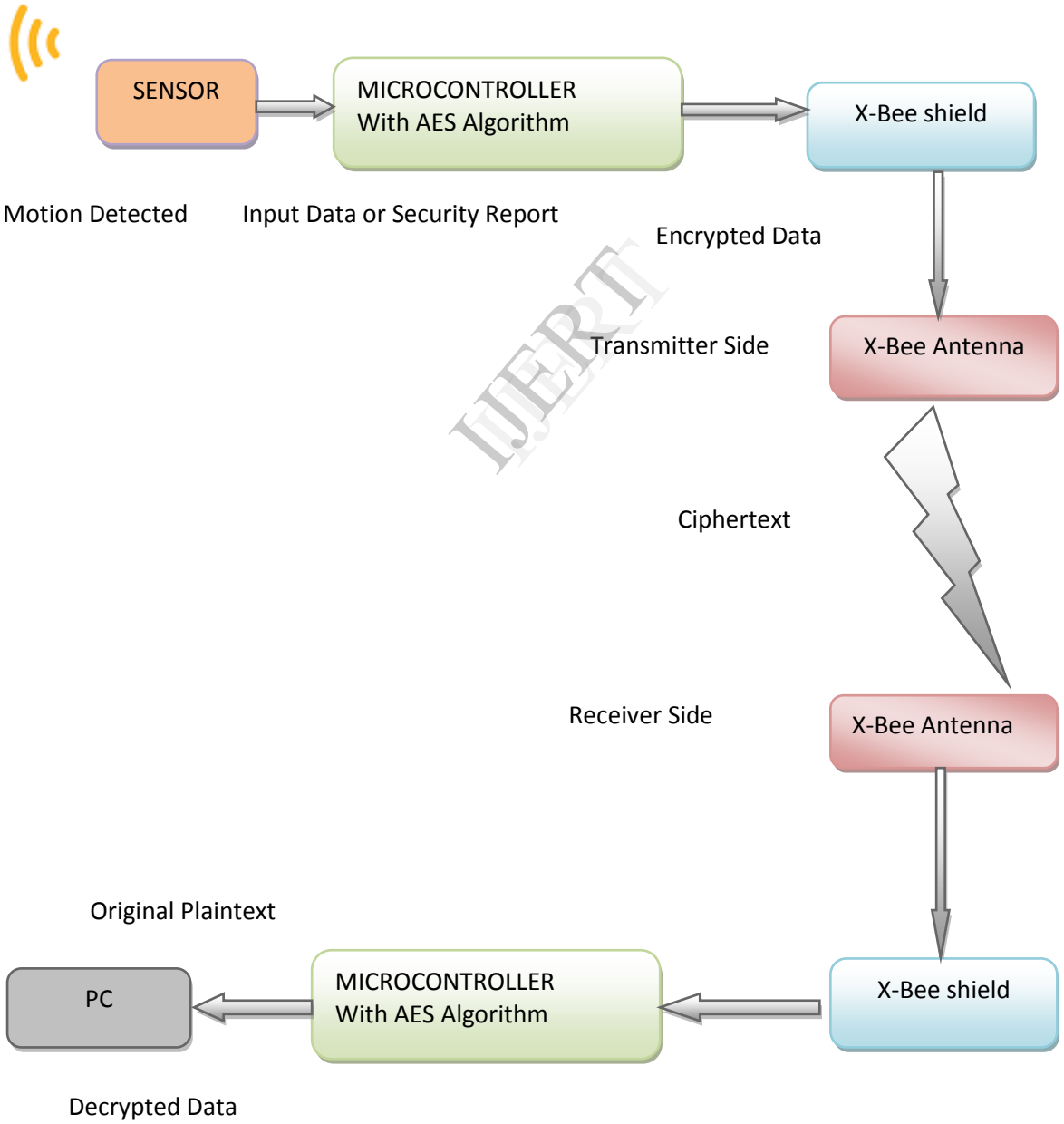

Figure 1, The Proposed System Block Diagram

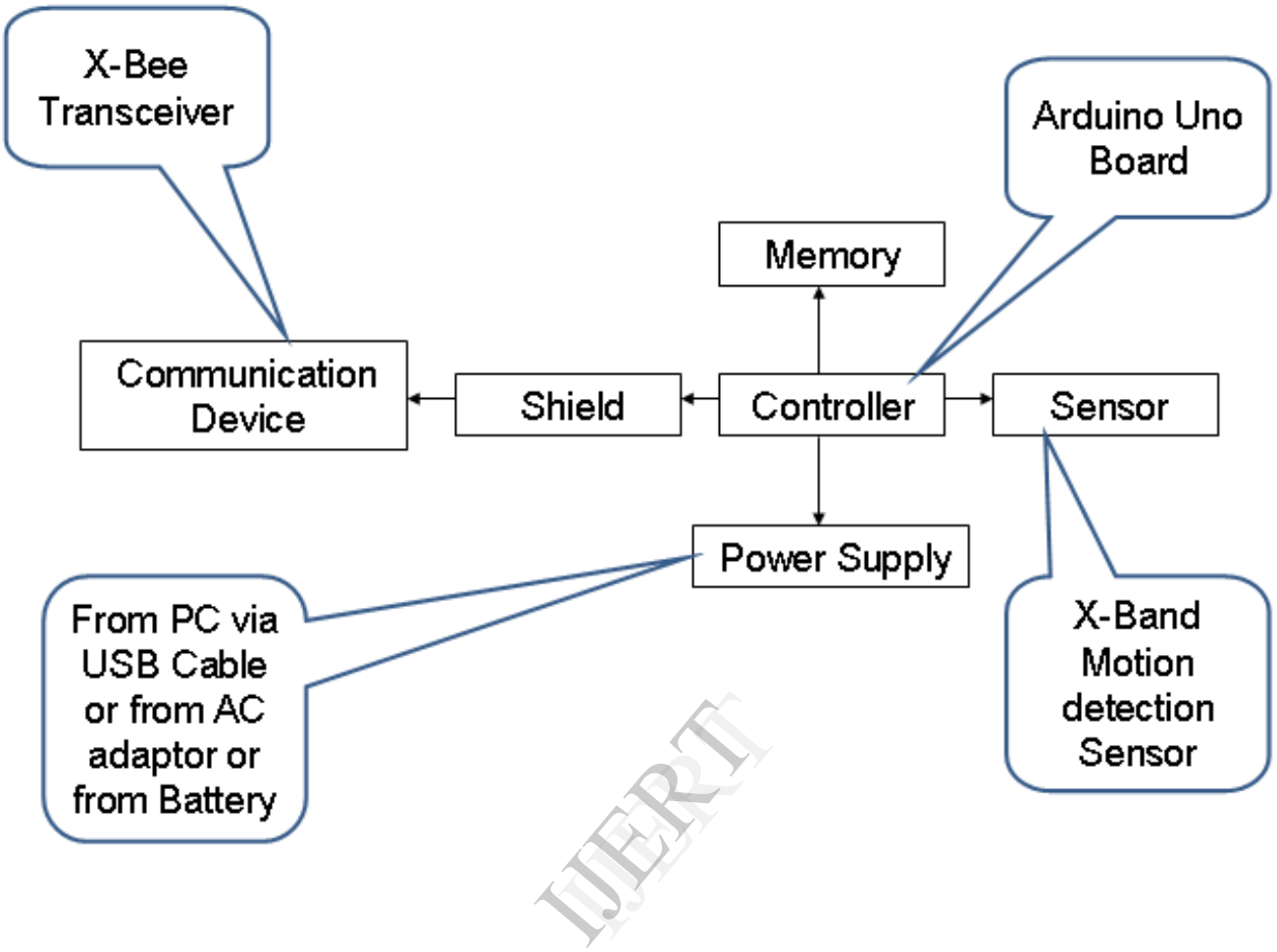

Figure 2, Arduino Uno Board Interconnections

Figure 2 shows the transmitter side or the receiver side. The Arduino Uno microcontroller board is connected to the shield card. The shield card is connected to the X-Bee Transceiver module. The X-Bee transceiver module is 1 mw Series 1. The Arduino Uno board is powered by PC via USB cable or from AC adaptor or from 9 V Battery. The Arduino Uno board is connected to X-Band Doppler radar motion detection sensor.

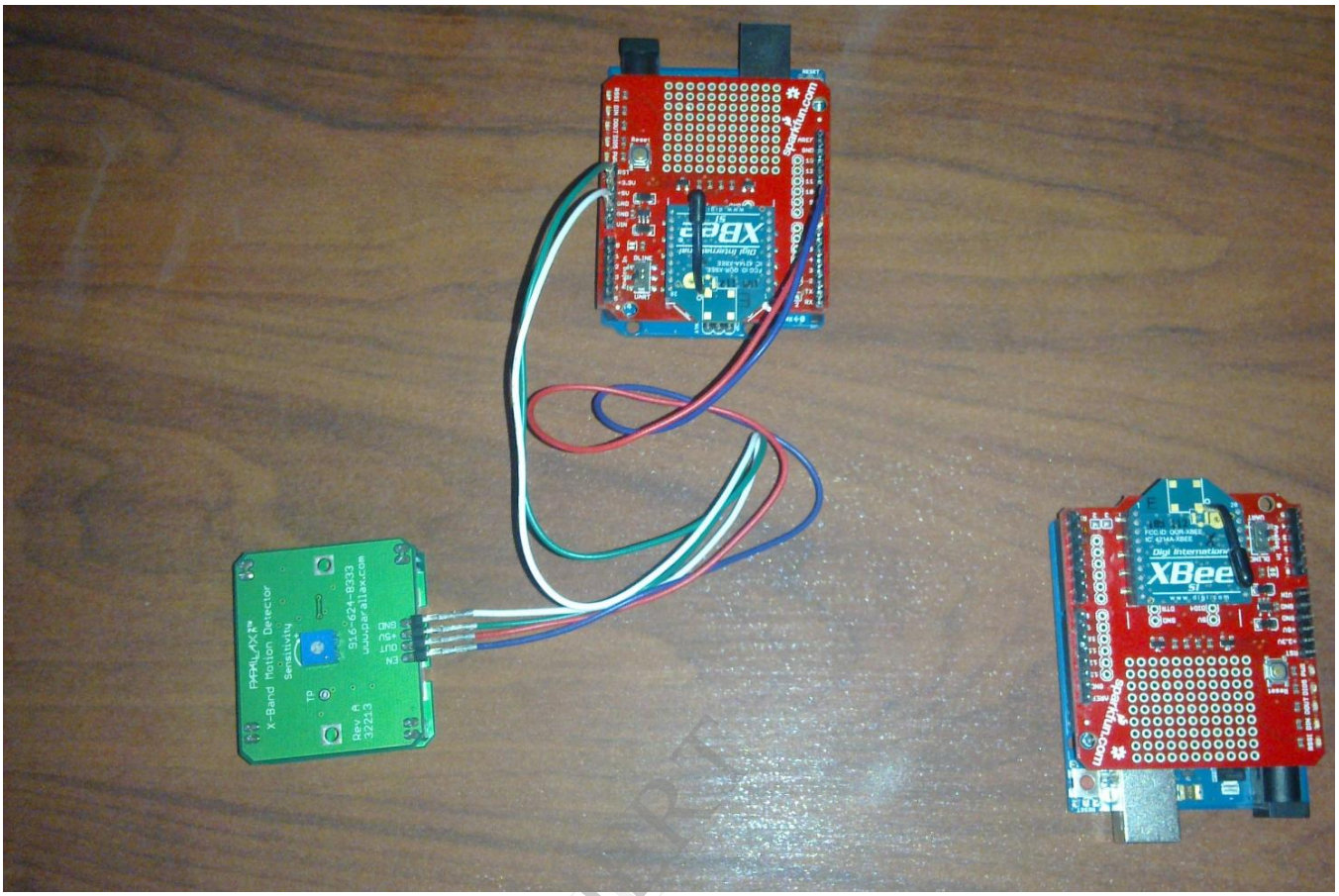

Figure 3, Proposed System Transmitter and Receiver

Figure 3 shows the transmitter and receiver components. The transmitter is composed of Arduino Uno microcontroller board, Shield card, X-Bee transceiver 1mw series 1 module, X-Band Doppler Radar motion detection sensor and AC adaptor power supply. The receiver is composed of Arduino Uno microcontroller board, Shield card, X-Bee transceiver 1mw series 1 module, AC adaptor power supply.

2 Related Work, Proposed System Components and their Specifications

This section introduces the related work, and the proposed system components and its specifications.

2.1 Related Work

Surveillance WSN can be hardware implemented using motes or using Arduino Uno microcontroller boards. A sensor node, also known as a mote, is a node in a wireless sensor network that is capable of performing some processing, gathering

sensory information and communicating with other connected nodes in the network. A mote is a node but a node is not always a mote. We choose to hardware implement our surveillance WSN on Arduino Uno microcontroller boards to fully control all the hardware of the node. Surveillance WSN for battlefield or boarders is previously hardware implemented as works discussed in [2, 3, 4, 5].

There is no previous hardware implementation works for reliable network recovery from BS failure. Our work is the first work that addresses the reliable network recovery from BS failure.

2.2 Proposed System Components and their Specifications

This sub-section describes the proposed system components which are seven components and their specifications and features to be used for the hardware implementation of reliable network recovery from base station failure.

Table 1, The Proposed System Components

No.	Item	Quantity
1	X-Bee 1mw Series 1 Module	2
2	Arduino X-Bee Shield	2
3	X-Bee USB Programmer	1
4	X-band Motion Detection Sensor	1
5	Arduino UNO Microcontroller Board	2
6	USB cable	2
7	Mini USB cable	1

2.2.1 X- Band Doppler Radar Motion Detection Sensor

The X-Band Motion Detection Sensor that is shown in Figure 4 is a common ingredient in security systems, automatic lighting, and automatic door openers. It can detect movements in a room, yard, or even on the other side of a wall. It is a Doppler radar sensor that operates in the X-band frequency at 10.525 GHz. It indicates movements with oscillations using its high/low output. Sensitivity is manually adjusted with a potentiometer

on the back of the device, offering direct line of sight detection from roughly 8 ft to slightly over 30 ft (2.4 m to 9 m).

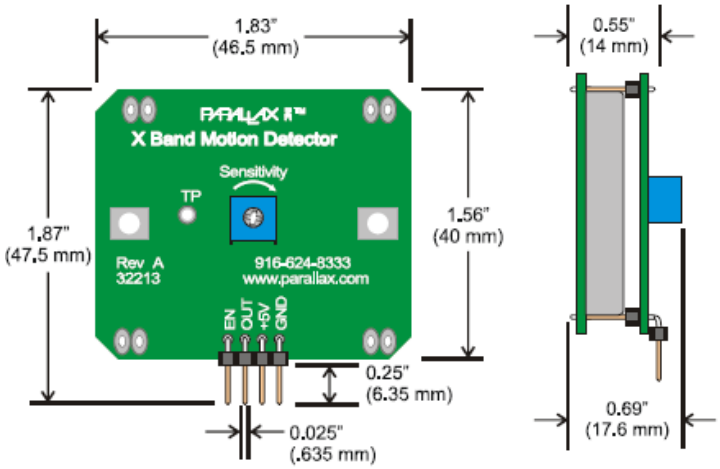

Figure 4, The X-Band Motion Detection Sensor Dimensions

The X-Band Motion Detection Sensor module is constructed of two boards connected together. The two boards are a control board, and the antenna PCB with the Doppler sensor.

2.2.2 X-Bee 1 mw Series 1 Transceiver

The Digi X-Bee 1 mw Series 1 transceiver 802.15.4 modules are the easiest-to-use, most reliable and cost-effective RF devices. The X-Bee transceiver 802.15.4 modules provide two friendly modes of communication; a simple serial method of transmit/receive or a framed mode providing advanced features. X-Bees Transceiver is ready to use out of the package, or they can be configured through the X-CTU program utility. The transceiver is controlled by the Arduino Uno microcontroller. These modules can communicate point to point as in Series 1, from one point to multipoints as in Series 2, or in a mesh network as in Series 2.

2.2.3 X-Bee Programmer

This is a simple to use USB to serial base unit for the X-Bee transceiver. This unit works with all X-Bee modules including the Series 1 and Series 2, standard and Pro version. Plug the X-Bee module into the X-Bee programmer, attach a mini USB cable, and you will have direct access to the serial and programming pins on the X-Bee unit. The X-Bee programmer is used with the X-CTU software to program the X-Bee modules as coordinator and end device.

2.2.4 X-CTU Program

The X-CTU program is used to program the X-Bee modules after connecting the programmer board with the PC through the Mini USB cable. First, we select the Com

Port at which the X-Bee module is located. Second, we press Test/Query bottom to assure the right assignment to the module.

The X-CTU program has Modem Configuration bottom to program the X-Bee modules. The library of the X-CTU software should be updated before the programming takes place. There are two methodologies to program the X-Bee modules, first methodology is through AT commands and the second methodology is through the API commands. We choose to program the X-Bee modules through the AT commands.

2.2.5 Arduino Uno Microcontroller Board

The Arduino Uno board is a microcontroller board based on the Atmel microcontroller ATmega328. It has 14 digital input/output pins where 6 pins can be used as Pulse Width Modulation (PWM) outputs, 6 pins can be used as analog inputs, one pin can be used as a 16 MHz crystal oscillator, one pin can be used as a reset button, a USB connection, and a power jack. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

2.2.6 Arduino Uno Software

The Arduino Uno software is a software program that runs C programming language that is used to write down the code on the Arduino Uno microcontroller board. The code refers to the code of the motion detection sensor, the code of the transmitter, the code of the receiver and the code of the AES encryption algorithm. First, we write down the C code then we verify the code and debug the code. Second, we upload the code on the microcontroller to test it for validation.

2.2.7 X-Bee Shield Card

The serial pins (DIN and DOUT) of the X-Bee Transceiver are connected to the Arduino Uno microcontroller board through the shield card which allows the programmer to select a connection to either the UART pins (D0, D1) or any digital pins on the Arduino Uno microcontroller board (D2 and D3 default). Power is taken from the 5V pin of the Arduino Uno board and regulated on-board to 3.3VDC before being supplied to the X-Bee Transceiver. The shield also takes care of level shifting on the DIN and DOUT pins of the X-Bee Transceiver. In the latest revision the diode level shifter is replaced with a more robust MOSFET level shifter.

The board also includes LEDs to indicate power and activity on DIN, DOUT, RSSI, and DIO5 pins of the X-Bee Transceiver. The Arduino Uno board reset button is brought out

on the shield, and a 9x11 grid of 0.1" holes are available for prototyping. The shield does not come with headers installed.

3 Design and Implementation of the Proposed System

In this section, the design and implementation of the proposed system is introduced. This section is concerned with the programming of the microcontroller board, the motion detection sensor program, programming the microcontroller at the transmitter side with the transmitter code, programming the microcontroller at the receiver side with the receiver code, programming the microcontroller at the transmitter side and the receiver side with the AES encryption algorithm, configuring the X-Bee transmitter using X-CTU software, configuring the X-Bee receiver using X-CTU software and finally providing the interconnections between the serial monitor cable and the Arduino Uno board at the receiver.

3.1 Security Report Content

We assume that the security report content is the following fields: count which is the number of threats at the sensor node and it is 8 bits, the time which is 24 bits, attacked node ID which is 16 bits, node reputation which is 8 bits, replica number which is 8 bits, attack ID which is 8 bits and data integrity which is 32 bits. The size of the security report for one attack is 104 bits.

Table 2, Security Report Content

Field	First Field Count	Second Field Time	Third Field Attack ID	Fourth Field Attacked Node ID	Fifth Field Reputation	Sixth Field Replica Number	Seventh Field Data Integrity
Data Size	8 bits	24 bits	8 bits	16 bits	8 bits	8 bits	32 bits

3.2 Programming the Arduino Uno Microcontroller

The Arduino Uno board is a microcontroller board based on the ATmega328 microcontroller. Two Arduino Uno boards are used; one of them is used at the transmitter side and the other is used at the receiver side. The board can be powered by connecting it to a PC through USB

cable or by connecting it to AC adaptor or by connecting it to a 9 V battery. The Arduino Uno board at the transmitter side is connected to X-Band motion detection sensor, shield card and the X-Bee 1 mw Series 1 Transceiver is connected to the shield card. The Arduino Uno board at the transmitter has three codes; the motion detection sensor code, the transmitter code and the AES encryption algorithm code. The Arduino Uno board at the transmitter has two inputs; the input from the motion detection sensor and the input from the security report which is 80 bits for each threat. The Arduino Uno board at the receiver side is connected to shield card and the X-Bee 1 mw Series 1 Transceiver is connected to the shield card. The receiver is connected to a PC through serial monitor cable. The Arduino Uno board at the receiver has two codes; the receiver code and the AES encryption algorithm code. The Arduino Uno board at the receiver has two outputs; the output represents that a motion is detected and the output from the security report which is 80 bits for each threat.

3.3 Programming the Microcontroller with the Motion Detection Sensor Code

The X-band motion detection sensor has four legs which are connected to the Arduino Uno microcontroller board as shown in Figure5.

- Pin GND at Sensor ⟹ Pin GND at Arduino board
- Pin ENABLE at Sensor ⟹ Pin # 8 at Arduino board
- Pin OUT at Sensor ⟹ Pin # 7 at Arduino board
- Pin 5 V at Sensor ⟹ Pin 5 V at Arduino board

Figure 5, Interconnections between Arduino Uno Board and Motion Sensor

The sensor is controlled by the Arduino Uno microcontroller board. The X-band motion detection sensor uses the Doppler Effect which means that it sends a band of frequencies towards the watched area and receives these frequencies again to calculate the shift between the transmitted and received frequencies due to a motion in the surveillance area. After collecting the motion data, the sensor now will pass these data to the microcontroller which in turn will encrypt the messages using the AES algorithm.

3.4 Programming the Microcontroller with the Transmitter program

The transmitter collects the sensed data from the motion detection sensor and the security report that contains the threats where the fields of the security report are the followings: count which is 8 bits, time which is 24 bits, attacked node ID which is 16 bits, node reputation which is 8 bits, replica number which is 8 bits, data integrity which is 32 bits and attack ID which is 8 bits. The total security report for one threat is 104 bits

We faced two problems at the transmitter. The first problem is that the transmitter sends the data in ASCI format. The second problem is that the transmitter does not have a start frame. We solved the two problems. The first problem is solved by allowing the transmitter to send in ASCI format then the receiver changes the input data from ASCI format to binary format to return the original plaintext. The second problem is solved by inserting a start frame before the transmission. This start frame is A frame. At the receiver side the receiver starts to receive with A frame then the sent data. The transmitter encrypts the sensed data or the security report with AES encryption algorithm to send the data encrypted. The receiver receives the data in ASCI format then converts it to binary format then decrypts the data using AES encryption algorithm.

3.5 Programming the Microcontroller with the Receiver Program

The receiver which is a security manager received the sent data but at first the receiver received the “A” frame which is the start frame from the transmitter.

The receiver converts the received data ASCI format to binary format then the receiver decrypts the received data using the AES encryption algorithm.

3.6 Programming the Microcontroller with AES Encryption Algorithm

The microcontroller is programmed with the AES encryption algorithm at the transmitter and receiver.

3.7 Programming the X-Bee Transceiver with the Programmer Board and X-CTU Program

Both the X-Bee transmitter and the X-Bee receiver are programmed using the programmer board which is connected to the PC through the Mini USB cable and the X-CTU program.

3.7.1 Programming the X-Bee Transmitter

The transmitter X-Bee will be programmed to be a coordinator X-Bee with Personal Area Network of 1111 and the destination addresses high and low will be programmed from the back of the receiver X-Bee module using X-CTU program.

3.7.2 Programming the X-Bee Receiver

The receiver X-Bee will be programmed to be an End device X-Bee with Personal Area Network of 1111 and the destination addresses high and low from will be programmed from the back of the transmitter X-Bee module using X-CTU program.

4 Results and Evaluation Metrics

This section shows the evaluation metrics for the proposed system and the results from the proposed system.

4.1 Evaluation Metrics

1- Security Report Size
The security report size is 104 bits for each threat.
2- Passing Current at the Transmitter and Receiver from the Security Report
The current is measured from the USB port at the transmitter and receiver side. The USB port connects between the Arduino Uno board and the PC.
3- Power Consumption at the Transmitter and Receiver from the Security Report
The power consumption at the receiver side from the security report is the multiplication of the passing current and the input voltage.

4.2 Results

The results of the hardware implementation for reliable network recovery from base station failure is shown in two folds; the measurements of the passing current at the Arduino Uno microcontroller board and the measurements of the power consumption at the transmitter and receiver side for the security report.

4.2.1 Measurements of Passing Current at the Receiver from the Security Report

The passing current is measured at the receiver from the USB port. The passing current is 100 mA.

4.2.2 Measurements of Power Consumption at the Receiver from the Security Report

The power consumption is the multiplication of the passing current and the input voltage. The measured power consumption is at receiver which is the security manager. The sent data for one threat is 104 bits. The power consumption is 5 V multiplied by 100 mA which is equal to 500 mw. For 1 second, the energy consumption is 500 mille Joule. We test the proposed model with 1A/Hour battery. The battery can remain for 1A/100mA which is equal to 10 Hours. For large network, we assume that the total threats reported at a security manager are 25 threats in a day therefore; we can send the threats in 25 Seconds for one day. The transmitted data current is variable according to the distance between the sensor node which is transmitter and the security manager which is the receiver. The transmitted data current is from 100 mA to 300 mA. For the received data at the security manager, the battery can remain for 10H/25 Seconds which is equal to 360X4 days. If the transmitted data power consumption is three times the power consumption of the received data, therefore; the battery can remain for 360 days which is low power consumption. The transmitter can send five threats in one second which is the maximum transmitted data from the transmitter.

4.2.3 Plaintext Input Data to Transmitter

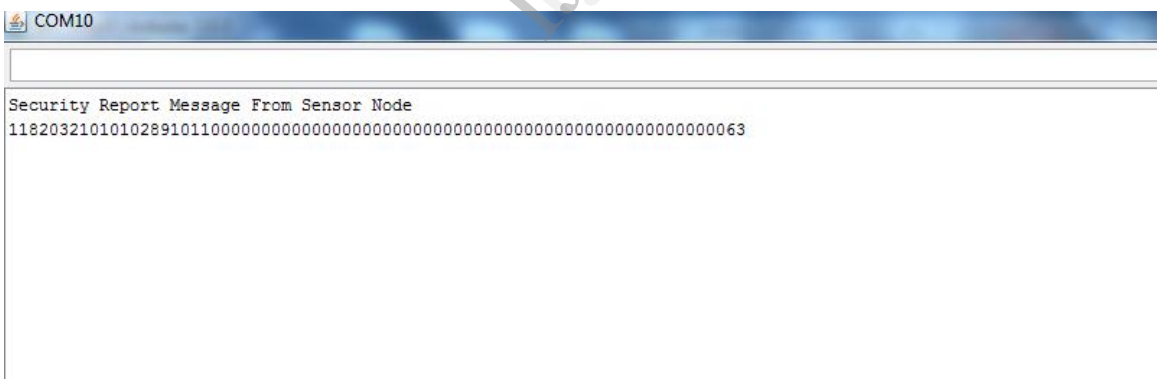

Figure 6, Security Report Content Data Input to Transmitter

Figure 6 shows the security report content which is 1 then 18 then 20 then 32 then 10 then 10 then 10 then 2 then 8 then 9 then 10 then 11 where 1 is the count, 18 is the hours and 20 is the minutes and 32 is the seconds, 10 is the attack ID which is Sybil attack, 10 is the attacked node ID, 10 is the replica number, 2 is the reputation of the node and 8 then 9 then 10 then 11 are the data integrity of the security report.

4.2.4 Ciphertext Output Data from Transmitter

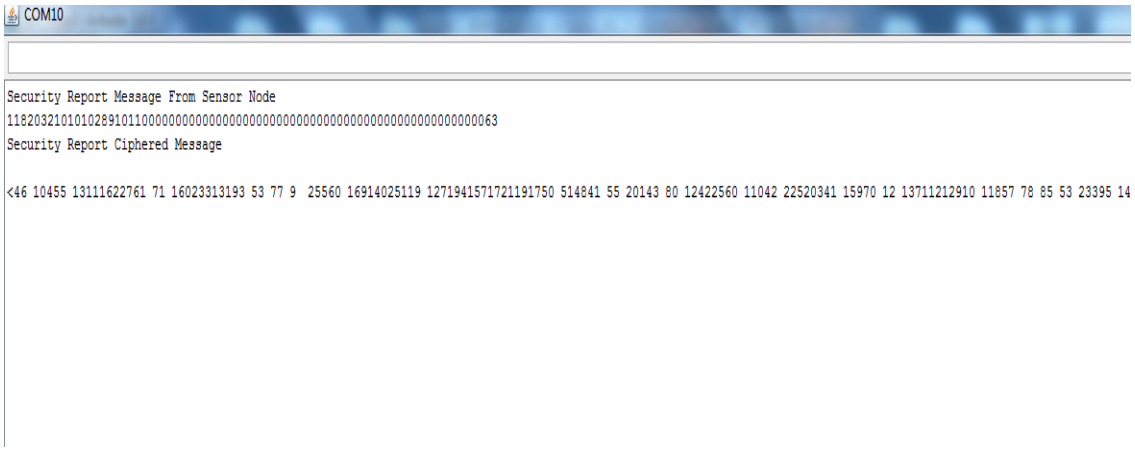

Figure 7, Ciphertext Data Output from Transmitter

Figure 7 shows the security report ciphered message which is the ciphertext output from the transmitter.

4.2.5 Data at Receiver Output

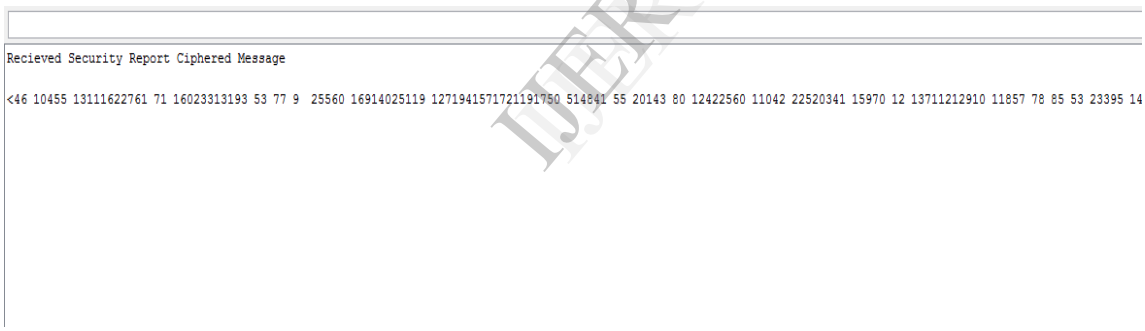

Figure 8, Ciphered Security Report Input to Receiver

Figure 8 shows the input ciphered security report at the receiver which is equal to the ciphered security report at the output of the transmitter.

Figure 9 shows the decrypted security report at the receiver output which is equal to the input security report at the transmitter input.

Figure 9, Security Report Output at Receiver

5 Comparison between our Work and Previous Works

To the best of our knowledge, there is not any scheme in the open literature addresses the base station failure. The current security schemes proposed for wireless sensor networks lack the ability of providing reliable network recovery in the case of base station failure.

The power consumption of the received security report is 500 mw which is low power consumption. This enables the security manager to receive and send security reports without affecting the life time of the security manager.

6 Conclusion

The design and hardware implementation of reliable network recovery from base station failure was implemented on Arduino Uno microcontroller boards. The used Transceiver is X-Bee 1 mw series 1 module. The motion detection sensor is X-Band Doppler Radar motion detection sensor. The X-Bee transceivers are programmed using programmer board and X-CTU program. The output data was monitored using serial

monitor cable and HyperTerminal Program. The code of the transmitter, the code of the receiver and the AES encryption algorithm code were done on Arduino Software as shown in Appendix A. The power measurements of the received security report at the security manager show that the reliable network recovery from base station failure has low power consumption.

References

[1] Mohamed Megahed, Dimitrios Makrakis, and Bidi Yang "SurvSec: A New Security Architecture for Reliable Network Recovery from Base Station Failure of Surveillance WSN", the 2nd International Conference on Ambient Systems Networks and Technologies, ANT 2011, September 17-19, 2011, Niagara Falls, Canada.

[2] Mahmood Ali, Annette Böhm, and Magnus Jonsson, "Wireless Sensor Networks for Surveillance Applications – A Comparative Survey of MAC Protocols", The Fourth International Conference on Wireless and Mobile Communications, IEEE 2008.

[3] Tatiana Bokareva, Wen Hu, Salil Kanhere, Branko Ristic, Neil Gordon, Travis Bessell, Mark Rutten and Sanjay Jha, "Wireless Sensor Networks for Battlefield Surveillance", Proceedings of The Land Warfare Conference (LWC), October 2006.

[4] Mario Lopez-Ramos, Jérémie Leguay, and Vania Conan, "Designing a Novel SOA Architecture for Security and Surveillance WSNs with COTS", International Conference on Mobile Ad-hoc and Sensor Systems 2007, IEEE 2007.

[5] Tian He, Sudha Krishnamurthy, John A. Stankovic, Tarek Abdelzaher, Liqian Luo, Radu Stoleru, Ting Yan, Lin Gu, Jonathan Hui, and Bruce Krogh, "Energy-Efficient Surveillance System Using Wireless Sensor Networks", MobiSYS'04, June 6–9, 2004, Boston, Massachusetts, USA, ACM 2004.

[6] http://www.digi.com/xbee/

[7] https://www.sparkfun.com/products/8687

[8] http://arduino.cc/en/Main/arduinoBoardUno

[9] http://arduino.cc/en/Main/ArduinoShields

[10]http://www.parallax.com/Store/Sensors/ObjectDetection/tabid/176/CategoryID/51/List/0/SortField/0/Level/a/catpageindex/2/Default.aspx