A Low-Energy Security Algorithm for Exchanging Information in Wireless Sensor Networks

Mohammad AL-Rousan [*], A. Rjoub and Ahmad Baset

Jordan University of Science and technology, School of Computer and Information Technology,
Irbid, Jordan
alrousan@*just.edu*
arjoub@just.edu.jo
abaset@just.edu.jo

Abstract: Because wireless sensor networks usually operate on unattended mode in hostile environments, the sensitive data should be protected using some sort of cryptography. Symmetric key scheme is more appropriate cryptography for wireless sensor networks due to its low energy consumption and simple hardware requirements, but most of them cannot provide sufficient security level (e.g. integrity, confidentiality, and authentication) as public key approach does. In this work, we propose a new security scheme that overcomes the limitations presented in both public- and symmetric-key protocols. The scheme relies on using one-way hash function to implement the data authenticity between nodes in the network, and a mix of symmetric and public key cryptography functions using the pre-distributed keys to implement the data confidentiality service. The symmetric key function is used to guarantee secure communications between in-network nodes while the public key function is used to guarantee a secure data delivery between the source node and the sink node. The proposed scheme is most suitable for wireless sensor networks that incorporate data-centric routing protocols. We have calculated the computational and communication overheads in terms of energy consumption in the new scheme using Directed Diffusion protocol [4]. The results have shown that the proposed scheme is scalable and an strong competitors to pure symmetric key schemes, yet, it maintains all security levels provided by public key schemes.

Keywords: Wireless Sensor Networks, Security, Directed Diffusion, Symmetric Keys, Public Keys.

1. Introduction

Security is a well-established field for general-purpose computing where security mechanisms address computing services (e.g. authentication, intrusion detection, etc.) and provide secure transactions. Cryptographic algorithms are an essential part of the security architecture of Wireless Sensor Networks (WSNs), therefore, selecting low-cost algorithms is an effective means of conserving resources. This paper focuses on the aspects of providing secure communications in WSNs using a new cryptography technique. The objective of this work is to propose a feasible encryption/decryption technique that suites the limited resources of a sensor node while maintaining strong cryptography mechanism.

When sensor networks are deployed in a hostile environment, security becomes extremely important as these networks are prone to different types of malicious attacks [3]-[5]. To provide security, communication transactions should be encrypted and authenticated. The main challenge in sensory networks is how to bootstrap secure communications between sensor nodes, that is, how to set up secret keys between communicating nodes. This problem is known as the key agreement problem which has been handled via two security mechanisms: Public Key Cryptography (PKC) and Symmetric Key Cryptography (SKC) [5]-[6].

Security experts favor the use of PKC whenever it is applicable because it provides all security services for the system under consideration including confidentiality (nondisclosure of secret information), integrity (prevention of data alteration), authentication (proof of identity), and non-repudiation (unique, non- contestable message origin) [7]. In addition, because of its asymmetry property, sensors do not need to carry the pre-distributed keys. Any two sensors can establish a secure channel between themselves, and the capture of some sensors will not affect the security of others [8].

The public key and symmetric key approaches maintain all features of the Directed Diffusion (DD) protocol presented in [4]. However, both of them have a dangerous drawback that affects both the security level accomplished and the efficiency of the protocol in terms of limited resources. For this reason we propose the hybrid approach which is still maintaining in-network processing feature of DD, accomplishing sufficient security level, and providing protocol efficiency in terms of energy and memory requirements. The main drawback of PKC is that it suffers from high computational complexity and communication overhead. Nevertheless, many recent studies have been made to investigate the feasibility of using PKC for sensor networks [9]-[11]. They have concluded that PKC has high feasibility to be implemented in wireless sensor networks and soon will be widely accepted. For example, Gura et al. [10] showed that Elliptic Curve Cryptography (ECC) signature verification takes 1.62s with 160-bit keys on ATmega128 8MHz processor, a processor used for Crossbow motes platform [10].

The SKC, on the other hand, is very attractive for sensor networks due to their energy and memory efficiency. There are extensive studies on using symmetric-key cryptography to achieve various aspects of security in sensor networks [12]-[17]. The main idea In SKC techniques is that the secret keys are pre-distributed among sensors before their deployment [14]. Doing so, a small amount of memory is used resulting in achieving the highest level of connectivity where high percentage of the neighboring sensors will be able to establish secure

Received November 20, 2008

communications between them [17]. Moreover, the capture of some sensors by an adversary should not jeopardize the security of other sensors, i.e. high degree of resilience can be achieved. However, due to memory limitation of sensor nodes, perfect security satisfaction has not been achieved yet [18], [19]. Furthermore, the only security service guaranteed when using SKC is the data confidentiality service, this means that the other services should be guaranteed by using another security system, consequently lower the efficiency of SKC schemes [7].

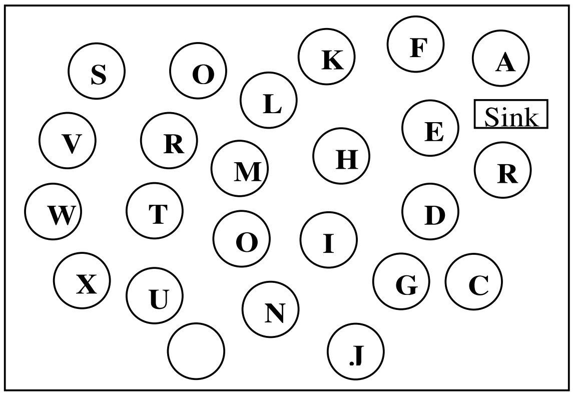

Figure 1: Wireless sensor network with 24 sensors randomly distributed.

From the above discussion, it is obvious that PKC schemes must be improved to alleviate their high complexity and high memory overheads, and SKC schemes, on the other hand, must be utilized more efficiently in order to provide more security satisfaction. This work proposes a new hybrid approach that combines the use of both PKC and SKC schemes in wireless sensor networks. The new approach still treats PKC as expensive operations, and uses it more selective and efficiently in order to maximize the lifetime of sensor networks. At the same time it maximizes the use of SKC in more efficient way. The validity of the new security method is implemented using Directed Diffusion routing protocols proposed in [4]. Directed Diffusion routing protocol has been developed and has become a breakthrough in data-centric routing [20]-[22]. In data-centric routing, the sink sends queries to certain regions and waits for data from the sensors located in the selected regions. Since data is being requested through queries, attribute based naming is necessary to specify the properties of data [23].

This paper is organized as follows. Section 2 gives a brief description about DD and how the proposed scheme is implemented in the protocol. In Section 3, we discuss the basis we follow in selecting public key and symmetric key schemes, then we derive the energy cost for each security schemes including the proposed one. In Section 5 we discuss the results obtained from the energy calculations made in Section 4. Finally, we conclude in Section 6.

2. Directed Diffusion Routing Protocol

Before we describe the proposed security scheme in details, we give a brief background about DD protocol because we will use it to explain how the system can be implemented in a data-centric wireless networks. The key idea in DD is to get rid of unnecessary operations of network layer routing in order to save energy. The DD protocol bases its operations and communications on named data. The sink (see Figure 1) requests data by sending an interest for named data which is broadcast through its neighbors. An example of an interest is shown in Table 1 for animal detection application in forests described in [4], [20]. Each sensor node receiving the interest can do in-network data aggregation and caching the interest for later use. The interest entry also contains several gradient fields where a gradient can be used to determine a reply link to a neighbor from which the interest was received. Using interest and gradients, paths can be established between sink and sources. Several paths can be established so that one of them is selected by the reinforcement process [4]. When a sensor node in the specified region receives an interest, it activates its sensors to begin collecting information about animal specified in the interest. When the sensors report the presence of animals (see Table 1), this information returns along the reverse path of interest propagation. Intermediate nodes might aggregate the data by combining reports from several sensors. We would to emphasis here that not all fields in the packets (interests and replies) are needed for aggregation at intermediate nodes, while the sink and the source must see the whole packet. We utilize this point in the implementation of the new security scheme.

Table. Attribute-value pairs in an interest and a reply.

An interest	A reply from a sensor
Type = two-legged animal	**Type = two-legged animal**
Interval = 30 ms	**Instance = bird**
Duration = 20 seconds	**Location = [125, 220]**
Rect. = [-100, 100, 200, 400]	**Intensity = 0.67**
	Confidence = 0.80
	Timestamp = 01:20:40

The DD protocol has several advantages; first, there is no need for a node to have a global or a local address since all communications occurs between neighboring nodes. Second, it is highly energy efficient since it is on demand and node do not have to maintain global information about network topology. Finally, node can do aggregation and caching, in addition to sensing.

The important issue is how the DD protocol would implement the hybrid scheme. The key idea behind the hybrid scheme is that it uses PKC and SKC in the encryption/decryption process. The possibility of applying the hybrid scheme to any routing protocol depends on the answer of the following question, "is it possible to divide the message into two parts so that the first part contains the data the sink is interested in, and the remaining part contains the data that is used by the intermediate nodes (between the source and the sink) to perform data aggregation?"

Looking at the interest shown in Table 1, it is obvious from the interest format that the "Rect.", "Interval", and "Duration" fields are the only fields that are used by the intermediate nodes to deliver the interest from the sink to the source and to perform the aggregation operation. Therefore, the aggregation data portion of the proposed hybrid scheme contains these fields, and the remaining field(s) constitute the required data in the hybrid scheme (see Table 2). Similarly, the reply shown in Table 1 can be divided into aggregation data (Location, Intensity, and Confidence) and required data (Type, instance, and timestamp). The proposed hybrid security scheme suggests that a symmetric key algorithm should be used by the intermediate nodes to encrypt/decrypt the aggregation data portion, while the required data portion is encrypted/decrypted using a public key algorithm. We present more details in Section 4 about this implementation.

Table. Two main portions in a DD message.

Aggregation data	Required data

The following steps should be followed before network deployment to implement a secure Directed Diffusion using the hybrid security scheme:

1) Store Public key, Symmetric key, and hash function codes in each node
2) For each node, select and save a randomly private key and keep the associated public key at the sink.
3) Save a public key of the sink at each node.
4) Save the same common symmetric key in all sensor nodes.

3. Security Requirements in WSNs

Recall that the most important network services that should be implemented by any security mechanism are: data confidentiality, data Integrity, and data Authenticity. Since our interest lies in implementing a security scheme for wireless sensor networks, we here must choose the schemes that meet the limited resources of sensor nodes. In this section we explain how and why we choose some of these schemes in our study. The process by which public key and symmetric key cryptography schemes should be selected is based on the following criteria:

- Energy: how much energy is required to execute the encrypt/decryption functions
- Program memory: the memory required to store the encryption/decryption program
- Temporary memory: the required RAM size or number of registers required temporarily when the encryption/decryption code is being executed
- Execution time: the time required to execute the encryption/decryption code.
- Program parameters memory: the required memory size to save the required number of keys used by the encryption/decryption function.

Since our proposed method suggests using a combination of two algorithms; a public key based algorithm and a symmetric key based algorithm, we here show how we use the above criteria in selecting these algorithms.

3.1 Selecting the Public Key Algorithm

Rivest-Shamir-Adelman (RSA) algorithm [24] and Elliptic Curve Cryptography (ECC) [10] are amongst the most well-known public key algorithms used in security systems. Many papers and articles [8], [10], [25] discussed the efficiency of each of these protocols, and showed that ECC is more efficient than RSA in terms of memory requirements because it requires much lower key size than RSA to achieve the same security level. ECC with 160-bit keys provides the currently accepted security level, and is equivalent in strength to RSA with 1024 (RSA-1024). However, it has been shown in [10] that RSA outperforms the ECC in terms of execution time because RSA relies on the modular exponentiations of large integers to execute encryption and decryption whereas ECC uses curve point multiplication to implement theses security functions. Recently, Gura et al. in [10] developed a mathematical improvement to speed up the point multiplication process by converting it into point additions and point doublings, this paper proved that ECC will outperform RSA when the microprocessor word size decreases and the encryption decryption key size increases. As a result, we choose the modified ECC proposed in [10] as major public key candidate to be used in WSN since it outperforms its counterpart (RSA) in terms of both memory requirements and execution time. In the following we analyze the ECC parameters such energy consumptions and memory requirements.

The experiment result of executing the modified ECC with 160-bit key size and 1024-bit message size in [10] shows that the execution time of the improved ECC on 8-bit ATMEL microprocessor with 8 MHz clock rate is 0.81s. the execution time is defined as:

$$Execution\ \ Time = IC * CPI * CCT \qquad (1)$$

where, IC is the Instruction Count, and the CPI is the Clock Rate Per Instruction. Equation 1yields:

$$IC = \frac{ExecutionT\ \ ime}{CPI\ \ * CCT} \qquad (2)$$

Since *CCT=1/processor frequency* (for 8 MHz, CCT= 125 ns) and the average *CPI*=1.3, *IC* can be calculated as:

$$IC = \frac{0.81}{1.3 * 125 * 10^{-9}} = 4984615\ = 4.9 \text{ MIPS} \qquad (3)$$

Given that each instruction represents one unit of energy consumption then ECC Computation Energy Consumption, EC(ECC)

$$E_c(ECC) = 4.984615 \quad energy \quad units \qquad (4)$$

The studies presented in [25]-[26], and [10] showed that the Atmega128 processor with 16 MHz can execute each instruction in one clock cycle, therefore, we can compute the number of executed instructions of ECC using the above equations as follows.

The *ECC execution time on 16 MHz/ ECC Execution time on 8 MHz={(CPI*CCT) on 16 MHz}/{(CPI*CCT) on 8 MHz}= (1/16*10^6)/(1.3/8*10^6)=0.38*.

Consequently, the ECC Execution time on 16 MHz=0.38*ECC Execution time on 8 MHz= 0.38*0.81=0.31 seconds. This means that the number of clock cycles (which is equal to the number of executed instructions) equals to

$$\begin{aligned} number\ of\ clock\ Cycle &= 0.31s * 16MHz \\ &= 4984615 \quad cycles \end{aligned} \tag{5}$$

Through out this work we will assume that each clock cycle is equivalent to one unit of energy for computation. In addition and according to [10], the improved ECC requires a code memory of 3682 bytes (3.6Kbytes), a temporary memory of 282 bytes (0.2 Kbytes), and a protocol parameters memory (public-key size (160) + private-key size (160)) of 40 bytes.

3.2 Selecting the Symmetric Key Algorithm

The research in [26] proposed a performance analysis for symmetric key algorithms and other security functions against many microprocessor architectures with different word size and operating frequency. The result of work concluded that RC5 with 64-bit key and 64-bit block size is the best algorithm in terms of execution time on all the microprocessor architectures that were tested. Based on results in [26], the symmetric key encryption/decryption algorithm (RC5) with a block size of 64 bits and a key size of 64 bits accomplishes the same security level as the ECC discussed in the previous section.

As done for ECC, we can compute the energy consumed when executing RC5 by extracting the number of instructions executed in the algorithm using Equation 2. Since the operations of RC5 consist of only XOR, Add, and rotation operations [27], [9], we can conclude that the CPI of RC5 is 1 cycle on Atmega128 [27]. Thus the total amount of RC5 Computation Energy Consumption, RC5_CEC, is given by

$$\begin{aligned} \text{RC5\_CEC} &= numbber\ \ of\ \ clock\ \ Cycle \\ &= \frac{Executiont\ \ ime}{CCT} \\ &= Execution\ \ \ time \times clock\ \ \ frequency \end{aligned} \tag{6}$$

The work by Ganesan et al. in [26] showed that the execution time of RC5 on Atmega 128 8-bit 16 MHz microcontroller is 0.002823s, thus, Equation 6 gives:

$$\begin{aligned} \text{RC5\_CEC} &= 0.002823 \quad * 16 \ * 106 \\ &= 45168 \quad \text{cycles} \quad = 45168 \quad \text{energy} \quad \text{units} \end{aligned} \tag{7}$$

As far as the hashing function is concerned, we choose to use Secure Hashing Algorithm (SHA-1) [26], [28]. SHA-1 is also a one-way hash function that produces a 160-bit output when any message of any length less than 264 bits is input. The operations constitute XOR, AND, OR, NOT and rotation, thus the CPI is 1 cycle when running on Atmega128 [27]. It has been found in [26] that the total execution time of SHA-1 using a 512-bit message is 0.007777 seconds. Thus, using the same logic in Equations 6 and 7, SHA-1 Computation Energy Consumption, SHA-1_CEC can be calculated as:

$$\begin{aligned} SHA\ \_1\_\ CEC &= 0.007777 \quad \times 16 \times 16 \\ &= 124432 \quad cycles \quad = 124432 \quad energy \quad units \end{aligned} \tag{8}$$

The work in [32] discusses the implementation of SHA-1 and showed that an m-bit message is processed by SHA$\left\lceil \frac{m\ +\ 65}{512} \right\rceil$ times.

4. Implementing Secured Directed Diffusion

In this section, we explain how different security schemes can be implemented in DD protocol. We first describe how DD operates and we highlight the key ideas behind the protocol. We discuss the implementation of DD using ECC public key, RC5 symmetric key, and the proposed hybrid key schemes. For each implementation, we derive the energy consumption.

In DD protocol an interest travels between three different types of nodes; the sink node, the intermediate node(s), and the source node. Therefore, we show how each of these node implements the security schemes under consideration and how much energy is consumed to run such implementation within the node. We assume that a node uses the first radio model for sending ad receiving data [30], [31]. According to that model, the total amount of energy to transmit and receive a message a message containing K bits is given in Equations 9 and 10 respectively.

$$E_{Tx}(k,d) = E_{elec} * K + \varepsilon_{amp} * K * d^2 \tag{9}$$

$$E_{Tx}(k,d) = E_{elec} * K \tag{10}$$

where, E_{elec} is the energy consumed in transmitting or receiving one bit, and is the energy consumed in amplification process.

According to [30] it is well known that the energy required to send 1 bit is equivalent to the energy to perform 1000 computations, with each computation equal to one instruction. Therefore, the energy required to execute one instruction, $\boldsymbol{E_C}$, is

$$E_c = \frac{E_{Tx}(1,d)}{1000} \tag{11}$$

We start with the public key approach followed by the symmetric key approach, and finally the proposed hybrid approach.

4.1 Implementing Public Key Algorithm

Figure 2 shows the implementation of the public key algorithm in the source node. A message in DD is issued by the source node and it encrypts the data packet using the ECC public key algorithm. Note that the packet is encrypted twice to achieve three security services: confidentiality, Integrity, and Authenticity [32].

The total amount of energy consumption at the source node, Esn,, to encrypt an m-bit packet using ECC, and then send it, is given by Equation 12:

$$E_{sn}(ECC) = E_{Tx}(m,d) + 2 \times E_{enc-dec}(ECC) \tag{12}$$

The encryption/decryption energy equals the number of instructions in ECC (Equation 4) multiplied by the energy consumed by one instruction (Equation11), Thus using Equations 4, 9 and 11 in 12 gives Equation 13:

Received November 20, 2008

$$E_{sn}(ECC) = E_{elec} * m + \varepsilon_{amp} * m * d^2 + [2 * 4984615 * (\frac{E_{elec} + \varepsilon_{amp} * d^2}{1000})] \quad (13)$$

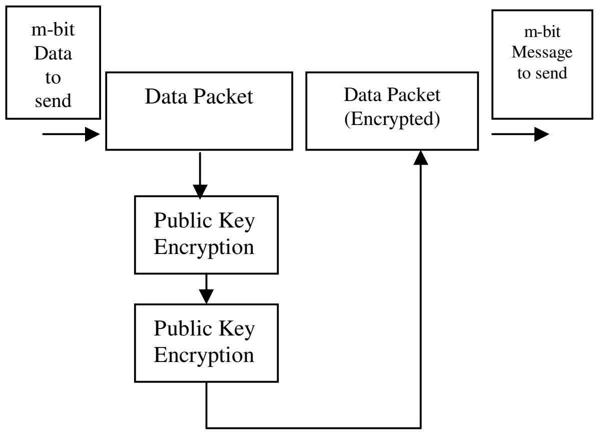

Figure 2: Public scheme-steps executed by source node before it sends the encrypted message

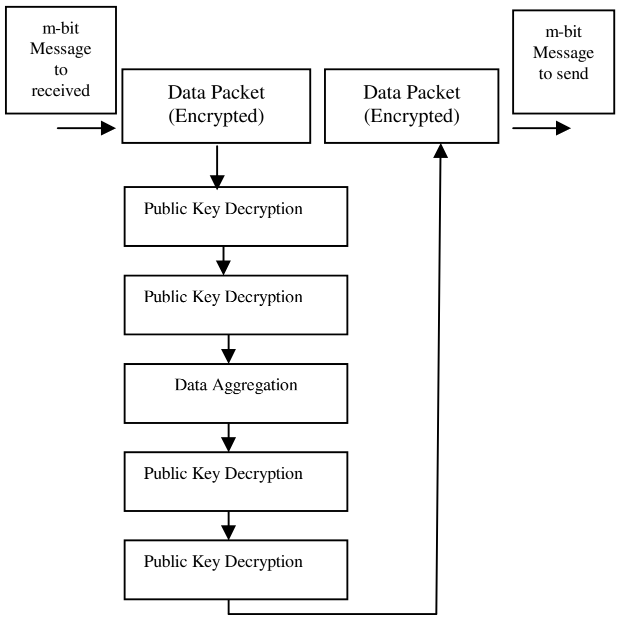

Figure 3: Public scheme-steps executed by each Intermediate node after receiving the encrypted message and before resending it again

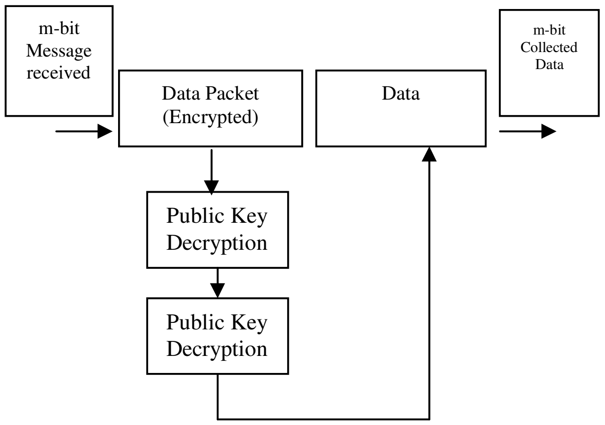

Figure 3: Public scheme-steps executed by sink node after receiving the encrypted message

When a packet reaches an intermediate node, the nodes will consume a total amount of energy, *Ein(ECC)* for receiving, decrypting, encrypting, and then sending the packet, is as follows:

$$E_{in}(ECC) = E_{Rx}(m) + 4 * E_{enc-dec}(ECC) + E_{Tx}(m,d) \quad (14)$$

Note the node here needs to decrypt/encrypt the packet twice to provide confidentiality, Integrity, and Authenticity, as seen in Figure 3. Substituting Equations 4, 10 and 11 into 14 give:

$$E_{in}(ECC) = 2(E_{elec} * m) + \varepsilon_{amp} * m * d^2 + [4 * 4984615 * (\frac{E_{elec} + \varepsilon_{amp} * d^2}{1000})] \quad (15)$$

Figure 4 explains what the sink node does when it receives a packet. The sink node consumes a total amount of energy, $E_{snk}(ECC)$, on only receiving and decrypting the packet, as follows

$$E_{snk}(ECC) = E_{Rx}(m) + 2 * E_{enc-dec}(ECC) \quad (16)$$

$$E_{snk}(ECC) = E_{elec} * m + [2 * 4984615 * (\frac{E_{elec} + \varepsilon_{amp} d^2}{1000})] \quad (17)$$

Therefore, the overall energy consumption of sending a packet (data or interest) from the sink node to the source node or vice versa, *Es-snk(ECC))*, assuming that there are I (I <=m) in-between intermediate nodes, can be given by:

$$E_{s-snk}(ECC) = E_{sn}(ECC) + I \times E_{in}(ECC) + E_{snk}(ECC) \quad (18)$$

$$E_{s-snk}(ECC) = \left[E_{elec} * m + [2 * 4984615 * (\frac{E_{elec} + \varepsilon_{amp} d^2}{1000})]\right] + I * \left[\begin{array}{l} 2(E_{elec} * m) + \varepsilon_{amp} * m * d^2 + \\ {[4 * 4984615 * (\frac{E_{elec} + \varepsilon_{amp} * d^2}{1000}}\end{array}\right] + \left[E_{elec} * m + [2 * 4984615 * (\frac{E_{elec} + \varepsilon_{amp} d^2}{1000})]\right] \quad (19)$$

4.2 Implementing Symmetric Key Algorithm

We now move to analyze the energy consumption when the DD protocol implements the symmetric key for encryption/decryption process. As we did for the public key algorithm, we start with the source node, as explained in Figure 5. Note that the symmetric key encryption does not implement the authenticity nor the integrity, for this reason, we have to use the hash function to implement them, but this will incur additional (h) bits to be sent along with the original data packet.

The total amount of energy consumption at the source node, *Esn,(RC5)* to encrypt and send an m-bit packet using RC5 and SHA-1 is given by Equation 20.

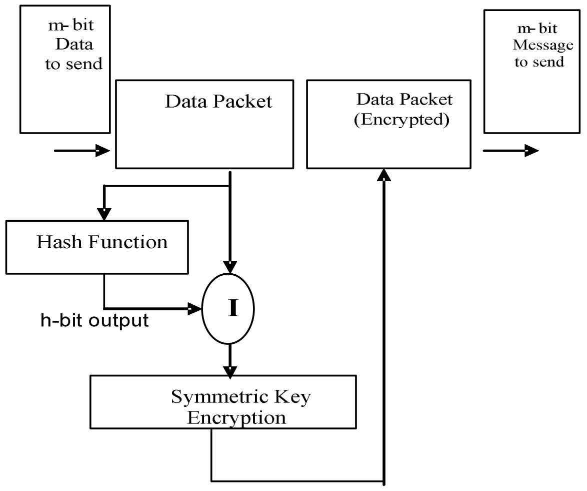

Figure 5: Symmetric scheme-steps executed by source node before it sends the encrypted message

$$E_{sn}(RC5) = E_{Tx}(m,d) + E_{enc-dec}(ECC) + E(SHA) \quad (20)$$

The encryption/decryption energy equals the number of instructions in RC5 (Equation 7) multiplied by the energy consumed by one instruction (Equation 11), Thus using Equations 7, 8 and 11 in 20 gives:

$$E_{sn}(RC5) = E_{elec}*(m+h)+\varepsilon_{amp}*(m+h)*d^2 + [\left\lceil\frac{m}{128}\right\rceil*45168*(\frac{E_{elec}+\varepsilon_{amp}*d^2}{1000})] + [\left\lceil\frac{m+65}{512}\right\rceil*124432*(\frac{E_{elec}+\varepsilon_{amp}*d^2}{1000})] \quad (21)$$

Note the key size and block size RC5 uses is a 60 bits long each (a total of 128 bits), and SHA-1 works on 512-bit block of the message (m).

As for the intermediate node(s), Figure 6 shows that each node will consume a total energy, Ein(RC5) on receiving m+h bits, encryption/decryption, hashing, and then transmitting m+h bits, is as in Equation 22.

$$E_{in}(RC5) = E_{Rx}(m+h) + 2*E_{enc-dec}(RC5) + 2E(SHA) + E_{Tx}(m=h,d) \quad (22)$$

$$E_{in}(RC5) = E_{elec}*(m+h) + [2*45168*\left\lceil\frac{m}{128}\right\rceil*(\frac{E_{elec}+\varepsilon_{amp}*d^2}{1000})] + [2*124432*\left\lceil\frac{m+65}{512}\right\rceil*(\frac{E_{elec}+\varepsilon_{amp}*d^2}{1000})] + [E_{elec}*(m+h)+\varepsilon_{amp}*(m+h)*d^2] \quad (23)$$

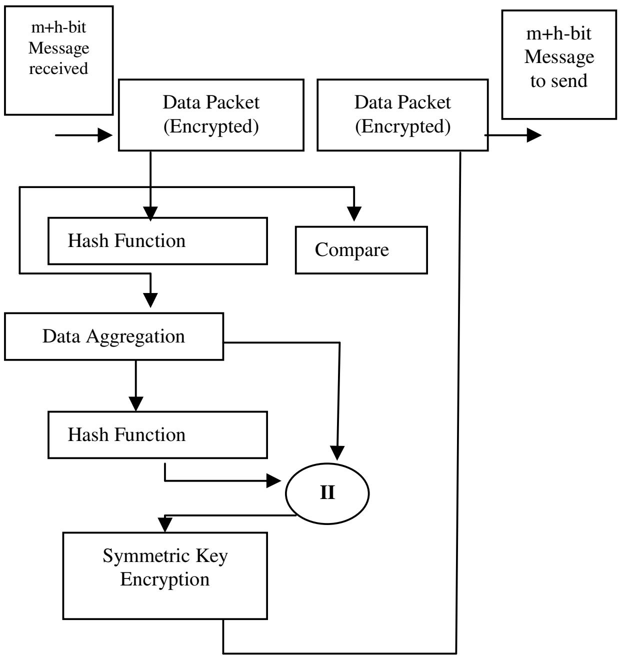

Figure 6: Symmetric scheme-steps executed by each Intermediate node after receiving the encrypted message

The operations at the sink node for the symmetric key algorithm are shown in Figure 7. The sink node consumes a total amount of energy, *Esnk(RC5)*, on only receiving, decrypting, and hashing a packet of m+h bits, is given in Equations 24 and 25.

$$E_{snk}(RC5) = E_{Rx}(m+h) + E_{enc-dec}(RC5) + E(SHA) \quad (24)$$

$$E_{snk}(RC5) = E_{elec}\times(m+h) + [45168\times\left\lceil\frac{m}{128}\right\rceil \times(\frac{E_{elec}+\varepsilon_{amp}*d^2}{1000})] + [124432\times\left\lceil\frac{m+65}{512}\right\rceil\times(\frac{E_{elec}+\varepsilon_{amp}*d^2}{1000})] \quad (25)$$

Therefore, the overall energy consumption of sending a packet (data or interest) from the sink node to the source node or vice versa, *Es-snk(RC5)*, assuming that there are *I (I<=m)* in-between intermediate nodes, can be given by Equation 26.

$$E_{s-snk}(RC5) = E_{sn}(RC5) + E_{in}(RC5) + E_{snk}(RC5) \quad (26)$$

Substituting Equations 21, 23, and 25 into Equation 26, the over all energy, *Es-snk(RC5)* is given as in Equation 27.

Received November 20, 2008

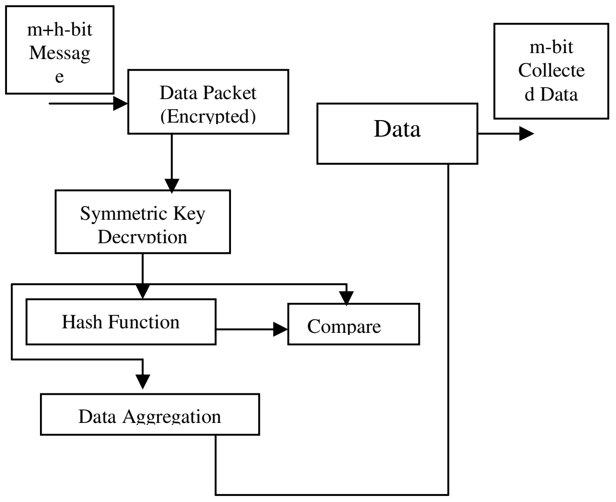

Figure 7: Symmetric scheme-steps executed by sink node after receiving the encrypted message

$$
\begin{aligned}
&E_{s-snk}(RC5) = \\
&E_{elec}\times(m+h)+\varepsilon_{amp}\times(m+h)\times d^2 \\
&+[\left\lceil \frac{m}{128} \right\rceil \times 45168 \times (\frac{E_{elec}+\varepsilon_{amp}\times d^2}{1000})]+ \\
&[\left\lceil \frac{m+65}{512} \right\rceil \times 124432(\frac{E_{elec}+\varepsilon_{amp}\times d^2}{1000})]+I\times \\
&\left\{\begin{array}{l}
E_{elec}\times(m+h)+[2\times 45168\times\left\lceil \frac{m}{128} \right\rceil \\
\times(\frac{E_{elec}+\varepsilon_{amp}\times d^2}{1000})]+[2\times 124432\times\left\lceil \frac{m+65}{512} \right\rceil \\
\times(\frac{E_{elec}+\varepsilon_{amp}\times d^2}{1000})]+ \\
[E_{elec}\times(m+h)+\varepsilon_{amp}\times(m+h)\times d^2]
\end{array}\right\}+ \\
&\left\{\begin{array}{l}
E_{elec}\times(m+h)+[45168\times\left\lceil \frac{m}{128} \right\rceil \\
\times(\frac{E_{elec}+\varepsilon_{amp}\times d^2}{1000})] \\
+[124432\times\left\lceil \frac{m+65}{512} \right\rceil\times(\frac{E_{elec}+\varepsilon_{amp}\times d^2}{1000})]
\end{array}\right\}
\end{aligned}
\tag{27}
$$

4.3 Implementing Hybrid Key Algorithm

In this scheme, if we limit the size of the "Aggregation Data " to be equal to the basic block of the symmetric key algorithm so that the symmetric key code will only be executed once in all nodes that the packet pass through. The Hybrid schemes may use the public key and the symmetric key implemented in the node.

As we did with the other sachems, let us start with the source node which performs the operations shown in Figure 8. The total amount of energy consumption at the source node using the Hybrid scheme, Esn,(H), to encrypt an m-bit packet and then send it, is given by:

$$
\begin{aligned}
&E_{sn}(H) = E_{Tx}((m+h),d) \\
&+E_{enc-dec}(ECC+RC5)+E(SHA)
\end{aligned}
\tag{28}
$$

The energy required to perform encryption, Eenc-dec (ECC+EC5), incorporate encryption using ECC and RC5 algorithm, and is calculate as:

$$
E_{enc-dec}(ECC+RC5) = \left\{\begin{array}{l}(number\,of\ instructions\ of\ ECC) \\ +(number\,of\ instructions\ of\ RC5)*E_c\end{array}\right\}
\tag{29}
$$

Substituting Equations 5, 7, 8, 9, 10, and 28 into 29, we get:

$$
\begin{aligned}
&E_{sn}(H) = \left\{\begin{array}{l}(4984615 + 45168) \\ *(\frac{E_{elec}+\varepsilon_{amp}*d^2}{1000})\end{array}\right\}+ \\
&\left\{124432 * \left\lceil \frac{m+65}{512} \right\rceil *(\frac{E_{elec}+\varepsilon_{amp}*d^2}{1000})\right\} \\
&+\left\{E_{elec}*(m+h,d)+\varepsilon_{amp}*(m+h)*d^2\right\}
\end{aligned}
\tag{30}
$$

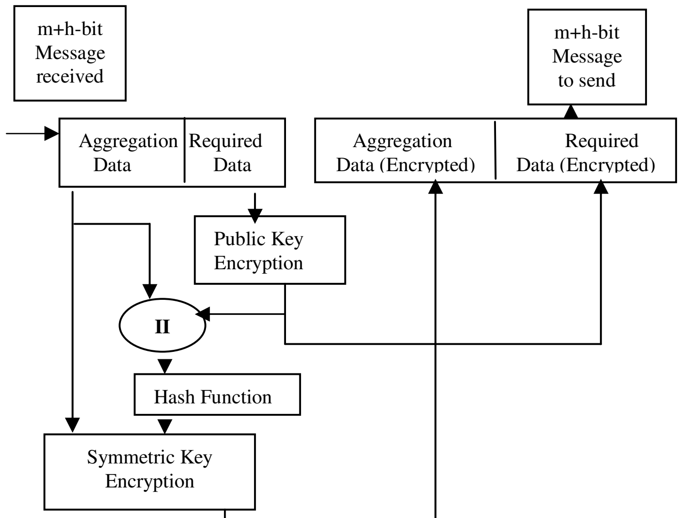

Figure 8: Hybrid scheme-steps executed by source node before it sends the encrypted message

For intermediate nodes, each node does not need to encrypt the part of the packet that is encrypted by the source/sink node using the public key, it rather needs to decrypt and encrypt the aggregation data using the symmetric key algorithm *(RC5)* and *SHA-1*, as seen in Figure 9. This is because the following fact about public key algorithm. Given two messages M1 and M2, if M1=M2 and the encryption and decryption keys are the same, then the cipher of both M1 and M2 are equal, i.e Ek[M1]=Ek[M2]=C. so the node will only check if the encrypted data is already existed in the data cache.

The energy consumption of the each intermediate node:

$$E_{in}(H) = E_{Rx}(m+h) + 2 * E_{enc-dec}(RC5) + 2E(SHA) + E_{Tx}(m = h, d) \tag{31}$$

Substituting Equations 7, 8, 9, 10, and 11 into 31, we get:

$$\begin{aligned} E_{in}(RC5) &= E_{elec} * (m+h) + \\ &[2 * 45168 * (\frac{E_{elec} + \varepsilon_{amp} * d^2}{1000})] + \\ &[2 * 124432 * \left\lceil \frac{m+65}{512} \right\rceil * (\frac{E_{elec} + \varepsilon_{amp} * d^2}{1000})] \\ &+ [E_{elec} * (m+h,d) + \varepsilon_{amp} * (m+h) * d^2] \end{aligned} \tag{32}$$

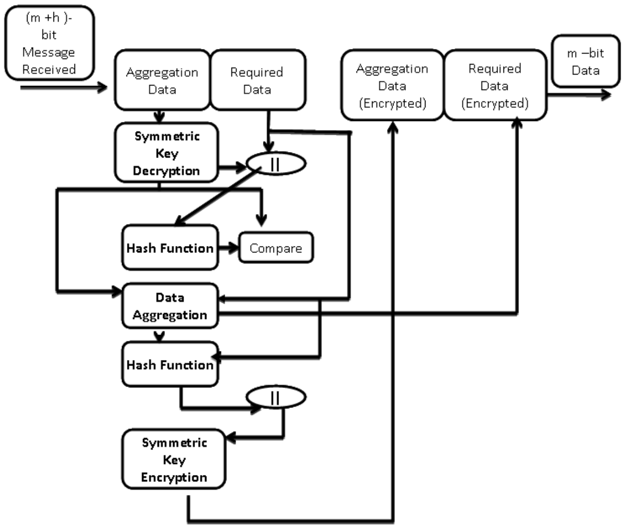

Figure 9: Hybrid scheme-steps executed by each Intermediate node after receiving the encrypted message and before it resend it again

The sink node for the Hybrid key algorithm is shown in Figure 10. The sink node consumes a total amount of energy, *Esnk(H)*, on only receiving *m+h* bits, decrypting the required data portion using the ECC public key algorithm, decrypting the aggregation portion using RC5 symmetric key algorithm, and hashing the data using SHA-1, as follows:

$$E_{snk}(H) = E_{Rx}(m+h) + E_{enc-dec}(ECC + RC5) + E(SHA) \tag{33}$$

Substituting Equations 7, 8, 9, and 11 into 33, we get Equation 34.

$$\begin{aligned} E_{snk}(H) &= E_{elec} * (m+h) \\ &+ \left\{ (4984615 + 45168) * (\frac{E_{elec} + \varepsilon_{amp} * d^2}{1000}) \right\} \\ &+ \left\{ 124432 * \left\lceil \frac{m+65}{512} \right\rceil * (\frac{E_{elec} + \varepsilon_{amp} * d^2}{1000}) \right\} \end{aligned} \tag{34}$$

The overall energy consumption of sending data or interest message from the sink node to source node, or vice versa, using the Hybrid key algorithm, *Es-snk(H)*can be calculated using Equations 30, 32, and 34, and assuming that there are I (I<=m) in-between intermediate nodes. Therefore, *Es-snk(H)* is given in Equation 35.

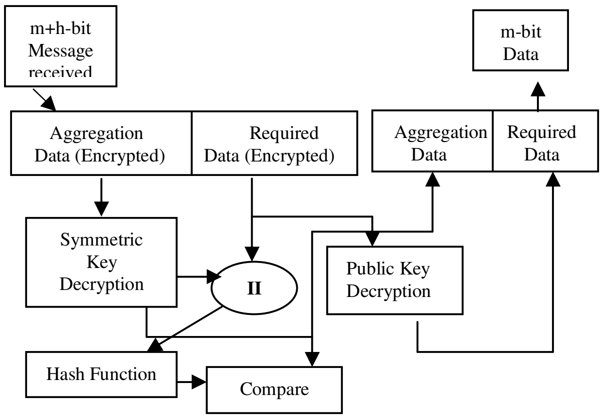

Figure 10: Hybrid scheme-steps executed by sink node after receiving the encrypted message

$$\begin{aligned} &E_{s-snk}(H) = \\ &\left(\begin{array}{l} \left\{ (4984615 + 45168) * (\frac{E_{elec} + \varepsilon_{amp} * d^2}{1000}) \right\} + \\ \qquad \left\{ 124432 * \left\lceil \frac{m+65}{512} \right\rceil * (\frac{E_{elec} + \varepsilon_{amp} * d^2}{1000}) \right\} \\ + \left\{ E_{elec} * (m+h,d) + \varepsilon_{amp} * (m+h) * d^2 \right\} \end{array} \right) \\ &+ I * \left(\begin{array}{l} E_{elec} * (m+h) + [2 * 45168 * (\frac{E_{elec} + \varepsilon_{amp} * d^2}{1000})] + \\ \qquad [2 * 124432 * \left\lceil \frac{m+65}{512} \right\rceil * (\frac{E_{elec} + \varepsilon_{amp} * d^2}{1000})] \\ + [E_{elec} * (m+h,d) + \varepsilon_{amp} * (m+h) * d \end{array} \right) + \\ &\left(\begin{array}{l} E_{elec} * (m+h) + \\ \left\{ (4984615 + 45168) * (\frac{E_{elec} + \varepsilon_{amp} * d^2}{1000}) \right\} \\ \qquad + \left\{ 124432 * \left\lceil \frac{m+65}{512} \right\rceil * (\frac{E_{elec} + \varepsilon_{amp} * d^2}{1000}) \right\} \end{array} \right) \end{aligned} \tag{35}$$

5. Results and Discussion

In this section we present and discuss results obtained from the energy analysis we made in the previous sections. To provide a valid and fair comparison we assume the three security schemes we considered in the above sections are executed on Atmega 128 16MHz 8-bit architecture AVR-instruction set, this microprocessor is widely used in many today's sensor nodes. We also assume that the sensor network consists of n nodes with I (I=10) intermediate nodes between the sink and any source node, and the distance between any two neighboring nodes is 1 meter. The number of intermediate nodes indicates how large the network is. Furthermore, the node uses the first order model for radio transmission (Equations 9 and 10) with transmitter electronics=receiver electronics=Eelec=50 nJ/bit, and the transmitter amplifier εamp=100 pJ/bit/m2. These parameters are consistent with many related works including [30], [31]. As for the parameters of the security schemes, we present the result in this sections using a message size of 1024 bits, a basic block size of 1024 bits for the SHA-1 hashing function with output size, h, of 160 bits, the basic block size in the RC5 symmetric key algorithm is 64 bits.

In our discussion we show and compare the energy consumptions at the sink, source, and intermediate node(s) for all security schemes under consideration assuming a message of size 1024 bits is traveling from the sink node to a source node (or vice versa) through I intermediate nodes, Figure 11 compares the energy consumes by the source node for all schemes using Equations 13, 21, and 30. In Figure 12 we compare the energy consumes by an intermediate node using Equations 15, 23, and 32. The energy consumption at the sink node is shown in Figure 13 using Equation 17, 25, and 34. The overall energy consumes in handle one message using each scheme is shown in Figure 14 using Equation. 35.

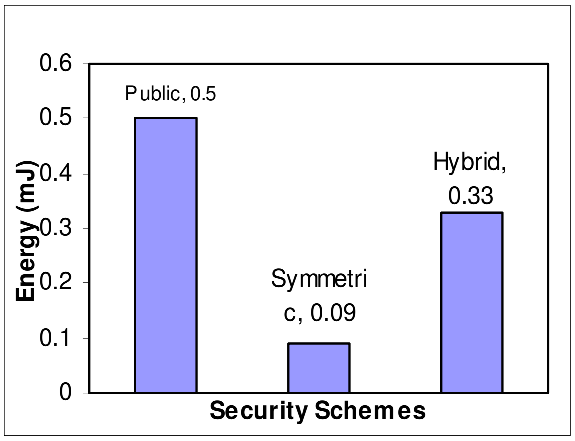

Figure 11. Energy Consumption at Source Node

5.1 Pure Public Key Scheme.

It is expected from the operation of the public key scheme in Figures 2, 3, and 4, that the energy consumption of the public key scheme, ECC, exceeds all of the other symmetric and hybrid schemes. The amount of energy consumptions in Figures 11 through 14 shows that source node uses ECC consumes energy 1.5 times more than that of the hybrid scheme and 5 times more than that of RC5. Intermediate nodes applying the ECC algorithm will die first since each one has to execute the public key functions many times to maintain the in-network processing feature of the routing protocol, and its obvious that the intermediate nodes will die before the source node since it performs more computations. Figure 12 clarifies that the intermediate node applying ECC consumes more than 6 times as it does in RC5, and more than 7 times as in the hybrid scheme. Additionally, the sink node consumes almost the same energy as the source node does for ECC algorithm. As a result the network energy when using ECC will be drained very quickly after a small number of interactions which makes this P the least favorite one for wireless sensor networks. The energy consumption of the public key scheme was expected because the cost of running its code is very high compared to other schemes.

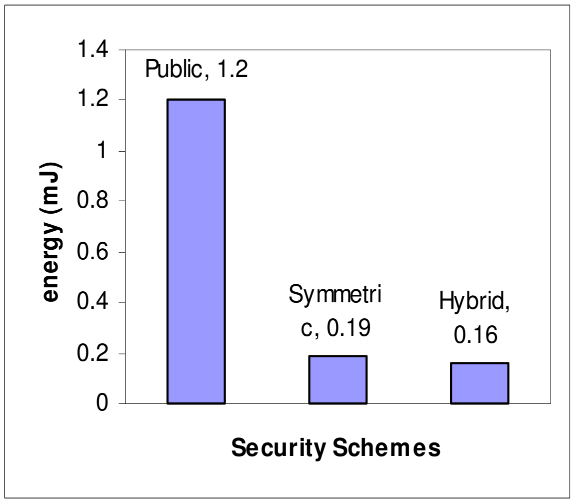

Figure 12. Energy consumption at intermediate node.

5.2 Pure Symmetric Key Scheme.

The energy consumption results from using the symmetric key scheme, RC5, was expected since it requires less number of computations than the other schemes do, this is obvious from the Figures 5,6, and 7. Although the symmetric key and the hash functions are executed many times to implement the three security services and still maintain the in-network processing of any routing protocol that is going to be integrated with this security scheme, but these execution do not require too many computations due to their relative simplicity. The energy consumptions depicted in Figure 11 show that source node using RC5 saves about 72% of the energy consumed by the hybrid scheme and 82% of the energy consumed ECC. Figure 12 indicates that the intermediate node consumes additional 19% of the energy consumed by the intermediate node using the hybrid scheme, and it saves more than 84% of the energy consumed by the intermediate node using ECC. Additionally, the sink node uses RC5 save about 72% of the energy consumed by the hybrid scheme and 82% of the energy consumed by ECC. As a result, the network energy in this scheme will be drained very slowly after a very large number of interactions which makes this the symmetric scheme, RC5, the most suitable and viable one for WSN.

5.3 Hybrid Scheme

Like the symmetric key scheme, the proposed hybrid scheme reduces the energy consumption through maximizing the use of RC5 over ECC as Figures 8, 9, and 10 show. Thus, we expect its performance to be close to the symmetric key scheme. The energy consumptions in Figure 11 show that source node in the hybrid scheme saves about 33% of the energy consumed by the public key scheme, but it consumes 3 times the energy consumed by the symmetric scheme. This is because the source node executes the ECC to guarantee end-to-end data delivery.

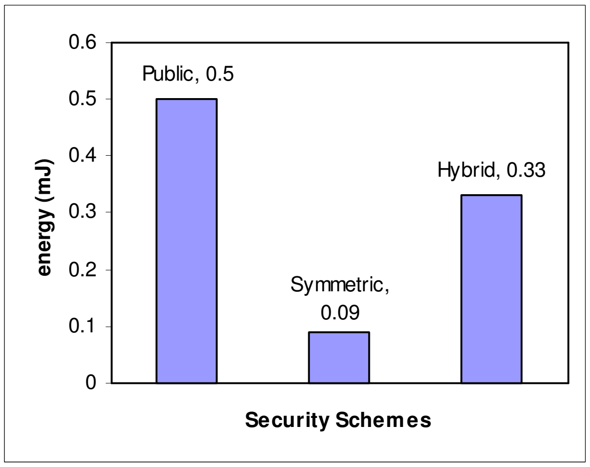

Figure 13. Energy Consumption at Sink Node

Figure 12 clarifies that the intermediate node in this scheme will save more than 86% of energy consumed by the intermediate node using the public scheme, and it saves more than 15% of the energy consumed by the intermediate node in the symmetric key scheme. This is expected behavior since not all parts in the received message should be decrypted to perform the data aggregation. Additionally, the sink node saves about 33% of the energy consumed by the public scheme, but it consumes 3 times more than the energy consumed when using the symmetric key scheme. This is because the sink node has to decrypt both the aggregation data and the collected data which means that the symmetric, public, and hash functions should be executed. As a result, the overall network energy in the hybrid scheme is equal to that of the symmetric key scheme as it is indicated in Figure 14 but with additional level of security. Therefore, the hybrid scheme is a suitable and viable algorithm for WSN like the symmetric scheme.

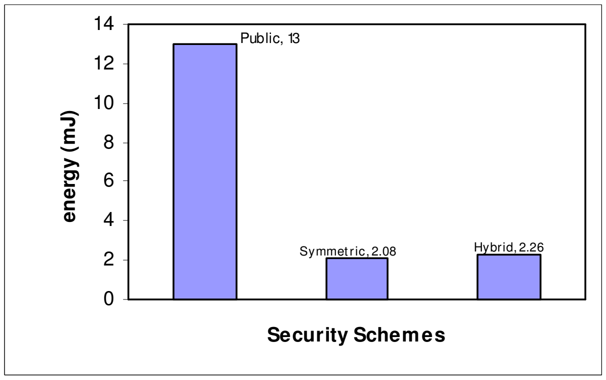

Figure 14. . Overall energy consumption

5.4 Impact of Network Parameters

It is obvious from the Equations from 12 through 35 that the main two parameters that affect the energy consumption of each scheme are network size (represented by I) and the message size (represented by m). Network size is the number of the sensor nodes (n) in the deployed the network. The size of the network has a direct effect on the number of intermediate nodes (I) between the source and sink; that is the larger n the larger I, and consequently the overall energy consumption of sending a message packet from source to sink and vice versa will increase, this is clearly obvious from the Equations 19, 27, and 35. Figure 15 shows that the overall consumption for all security schemes increases as the network size increases. The proposed hybrid scheme has shown slightly better performance than the symmetric scheme (RC5) in small and larger sensor networks. Therefore, it is a strong competitor to the symmetric scheme in wide range of applications and it is a scalable scheme.

Figure 16 shows the impact of varying the message size (m). This parameter has a direct effect on the number of times each encryption/decryption algorithm and hash function should be executed. This is because each algorithm works on a small basic block at a time so that if the message size is x times larger than the basic block size then the function should be executed x times.

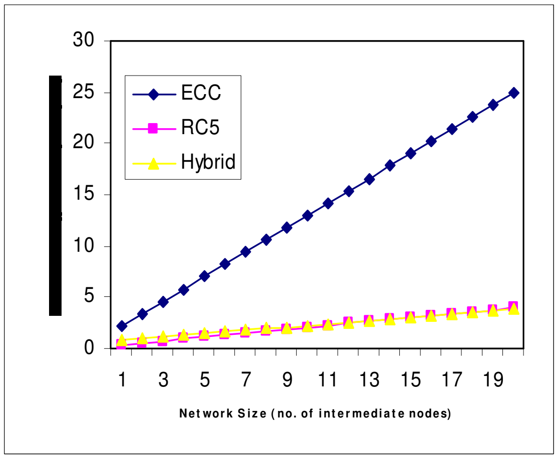

Figure 15. Impact of network size

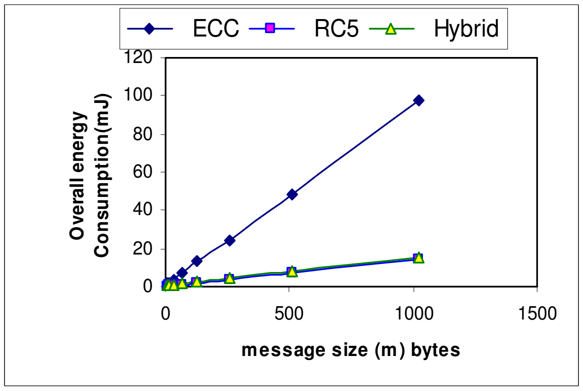

Figure 16. Impact of message size (I=10 nodes)

5.5 Illustrative Example

To complete the idea of the proposed security scheme in DD-like protocols, we present an explicatory example using the sensory network shown in Figure 1 that implements the animal detection system introduced in [4]. Assume the sink A issues an interest shown in Table 1, and the requested information could be satisfied by Sensor X (i.e. X is the source node, and I=20 nodes). In Section 2 above we explained how this message is divided into aggregation data and required data. The interest message will travel from the sink (A) through 20 intermediate nodes to reach Sensor X.

After the source node (X) receives the interest, it makes sure that it can satisfy the request, and decrypt the whole message; it starts collecting the data at the specified rate for the specified duration and then encapsulates the collected data in a reply message as shown in Table 1. The reply message will follow a reverse path selected according to the optimization process implemented in DD [1]. Let us assume that the selected path from the source node (X) to the Sink node (A) is through the nodes U,Q,M,I,D, and E.

The overall consumed energy consumed in this process is shown In Table 3 for all security schemes. The result of the energy analysis shows that the proposed hybrid security scheme consumes energy approximately as the symmetric scheme does, but it achieves higher security level than the symmetric scheme does since the adversary must pass at least two additional walls than it should do in the symmetric scheme.

Table 3. Energy comparison for all schemes

	Overall Consumed Energy (mJ)		
	interest	reply	total
Pure ECC Public Key	27.4	8.2	35.6
Pure RC5 Symmetric Key	4.36	1.33	5.69
Hybrid (ECC+RC5)	4.18	1.62	5.8

6. Conclusion

In this paper, we presented a new method of applying cryptography techniques in WSNs. The new hybrid scheme uses PKC and SCK in very selective way. It maximizes the lifetime of sensor node's batteries through minimizing the use of PKC. PKC is used for end-to-end communications, and SKC is used in-network communications. We evaluated the energy consumption for the hybrid scheme and made a comparison with PKC and SKC schemes. The results showed that the hybrid scheme provides superior performance when compared to pure PKC with energy saving ranges from 33 to 86 percents. Moreover, It competes with SKC and provides slightly better energy saving. We showed that the new hybrid approach is scalabe and suitable for large WSNs. Finally, although the proposed security scheme had been evaluated on the Directed Diffusion routing protocol, but actually it could be applied to any other routing protocol, more precisely, for data-centric types [30-32].

References

[1] K. Kifayat, M. Merabti, Q. Shi, D. Llewellyn-Jones, "Group Based Secure Communication for Large-Scale Wireless Sensor Networks", Journal of Journal of Information Assurance and Security, 2(2), pp. 139-147, 2007.

[2] Alessandro Sorniotti, Laurent Gomez, Konrad Wrona and Lorenzo Odorico, "Secure and Trusted in-network Data Processing in Wireless Sensor Networks: a Survey", Journal of Journal of Information Assurance and Security, 2(3) (2007), pp. 189-199, 2007.

[3] M. AL-Rousan, Al-Ali, and A. Talaa, "TCRP: A Tree-Clustering Protocol for Tracking Mobile Targets In Wireless Sensor Networks," In Proceedings of the International Symposium in Mechatronics, Jordan, pp. 1-5, 2008

[4] C. Intanagonwiwat, R. Govindan and D. Estrin, "Directed diffusion: A scalable and robust communication paradigm for sensor networks", in the Proceedings of the 6th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'00), Boston, MA, pp. 56-67, 2006.

[5] Yi Cheng, Dharma P. Agrawal, "An improved key distribution mechanism for large-scale hierarchical wireless sensor networks" Ad Hoc Networks 5(1), pp. 35-48, 2007.

[6] W. Du, R. Wang, P. Ning: An efficient scheme for authenticating public keys in sensor networks. Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing. MobiHoc), Urbana-Champaign, IL, pp. 58-67, 2005.

[7] Y. Law, J. Doumen, and P. Hartel, "Survey and benchmark of Block Cipher for Wireless Sensor Neworks", ACM Transactions on Sensor Networks 2(1), pp. 65-93, 2006.

[8] D. Liu, P. Ning, "Improving key pre-distribution with deployment knowledge in static sensor networks" ACM Transactions on Sensor Networks 1(2): pp. 204-239, 2005.

[9] G. Gaubatz, J. Kaps, and B. Sunar, "Public keys cryptography in sensor networks" In Proceedings of the 1st European Workshop on Security in Ad-Hoc and Sensor Networks (ESAS), 2004.

[10] N. Gura, A. Patel, A. Wander, H. Eberle, and S. Shantz "Comparing elliptic curve cryptography and RSA on 8-bit CPUs", In Proceedings of the 6th International Workshop on Cryptographic Hardware and Embedded Systems (CHES 2004):, Cambridge, MA, pp. 119-132, 2004.

[11] D. J. Malan, M. Welsh, and M. D. Smith, "A public-key infrastructure for key distribution in TinyOS based on elliptic curve cryptography" In Proceedings of the First IEEE International Conference on Sensor and Ad Hoc Communications and Networks, Santa Clara, California, October 2004.

[12] H. Chan, A. Perrig, and D. Song, "Random key pre-distribution schemes for sensor networks", In Proceedings of the IEEE Symposium on Research in Security and Privacy, pp. 197–213, 2003.

[13] S.A. Camtepe and B. Yener, "Combinatorial design of key distribution mechanisms for wireless sensor networks", IEEE/ACM Transactions on Networking 15(2), pp. 346 – 358, 2007.

[14] J. Lee and D. R. Stinson, "Deterministic key pre-distribution schemes for distributed sensor networks," In Proceedings of the ACM Symposium on Applied Computing, vol. LNCS 3357, pp. 294-307, 2005.

[15] S. Zhu, S. Setia, and S. Jajodia, "LEAP: Efficient security mechanisms for large-scale distributed sensor networks" In Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS'03), pp. 62–72, 2003.

[16] J. Lee and D. R. Stinson, "A combinatorial approach to key predistribution for distributed sensor networks," in Proceedings of the IEEE WCNC, pp. 1200-1205, 2005.

[17] Y. Cheng, D.P. Agrawal, "Efficient pairwise key establishment and management in static wireless sensor networks" In Proceedings of the Second IEEE International Conference on Mobile ad hoc and Sensor Systems, Washington, DC, 2005.

[18] C. Karlof, D. Wagner, “Secure routing in sensor networks: attacks and counter measures”, in: Proceedings of the 1st IEEE Workshop on Sensor Network Protocols and Applications, pp. 1-15, 2003.
[19] C. Karlof, D. Wagner, “Secure routing in sensor networks: attacks and countermeasures”, Ad Hoc Networks 1(1), pp. 293-315, 2003.
[20] R. Di Pietro, L. V. Mancini, Y. W. Law, S. Etalle, and P. Havinga. “LKHW: A directed diffusion-based secure multicast scheme for wireless sensor networks” In Proceedings of the First International Workshop on Wireless Security and Privacy (WiSPr’03), pp. 397-406, 2003.
[21] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci , "A Survey on Sensor Networks", IEEE Communications Magazine, August , pp. 102-114, 2002
[22] IF. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “Wireless sensor networks: a survey” Computer Networks 38(4), pp. 393-402, 2001.
[23] K. Akkaya and M Younis “A survey on routing protocols for wireless sensor networks” Ad Hoc Networks 3(3), pp. 325-349, 2005.
[24] R.L. Rivest, A. Shamir, and L.A. Adleman. A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM 21(2), pp. 120–126, 1998.
[25] A. Wander , N. Gura , H. Eberle , V. Gupta, S. Shantz, “Energy Analysis of Public-Key Cryptography for Wireless Sensor Networks”, In Proceedings of the Third IEEE International Conference on Pervasive Computing and Communications (PERCOM), March 08-12, pp. 324-328, 2005.
[26] P. Ganesan, R. Venugopalan, P. Peddabachagari, A. Dean, F. Mueller, M. Sichitiu, “Analyzing and modeling encryption overhead for sensor network nodes”, in Proceedings of WSNA '03, September, pp. 151-159, 2003.
[27] ATMEL Corporation, available at http://www.atmel.com, accessed May, 2009.
[28] A. Perrig, J. Stankovic, D. Wagner, “Security in wireless sensor networks”, Communications of the ACM 47(6), pp. 53-57, 2004.
[29] Heinzelman W, Chandrakasan A, and Balakrishnan H. “An Application-Specific Protocol Architecture for Wireless Microsensor Networks”, IEEE Transactions On Wireless Communications, 1(4), pp. 660-670, 2002.
[30] Heinzelman W, Chandrakasan A, and Balakrishnan H. “Energy-Efficient Communication Protocol for Wireless Microsensor Networks”, In Proceedings of the 33rd Annual Hawaii International Conference on System Sciences (HICSS 33); pp. 1-10, 2000.
[31] Heinzelman W., Kulik J., and Balakrishnan H. “Adaptive Protocols for Information Dissemination in Wireless Sensor Networks”, In Proceedings of the 5th ACM/IEEE Mobicom, August, pp. 174–85, 1999.
[32] W. Stalling. Cryptography and Network Security. Third Edition, Prentice Hall, New Jersey, USA, 2003.
[33] D. Braginsky and D. Estrin, "Rumor Routing Algorithm for Sensor Networks," in the Proceedings of the First Workshop on Sensor Networks and Applications (WSNA), Atlanta, GA, October, pp. 1-12, 2002.
[34] C. Schurgers and M.B. Srivastava, “Energy efficient routing in wireless sensor networks”, In Proceedings of the MILCOM on Communications for Network-Centric Operations: Creating the Information Force, McLean, VA, pp. 357-361, 2001.
[35] M. Chu, H. Haussecker, and F. Zhao, "Scalable Information-Driven Sensor Querying and Routing for ad hoc Heterogeneous Sensor Networks," The International Journal of High Performance Computing Applications 16(3), 293-313, 2002.

Author Biographies

Mohammad AL-Rousan (Corresponding author) is an associate professor at Computer engineering, he received his BSc. in Computer Engineering from King Suad University, 1985, and his M.S. from University of Missouri-Columbia, USA, 1992. He earned his Ph.D. from Brigham Young University, 1996, USA. Since then, he has been a professor of computer engineering, Jordan University of Science and Technology. He has published more than 40 articles in international journals and conferences. His research interests include wireless networking, system protocols, intelligent systems, Nanotechnology, and Internet Computing. He is the founder and director of AI and Robotics Laboratory at Computer Engineering Department, Jordan University of Science and technology.

Abdoul Rjoub is assistant professor at Jordan University of Science and Technology (JUST), His research interests include low power design system, high level design architecture, computer aided design and vary large scale integrated system (VLSI). Dr. Rjoub has more than 40 publications in international journals and conferences, where most of them have been cited in journals and conferences, as well as patents, books, technical reports and dissertations. Dr. Rjoub awarded best the reviewer award from International Arab Conference of Information Technology 2009

Received November 20, 2008