Energy-Efficient Authentication and Anti-Replay Security Protocol for Wireless Sensor Networks

R. Rughiniș*, L. Gheorghe**

** Faculty of Automatic Control and Computers, University Politehnica of Bucharest, Bucharest, Romania. (Tel: +40 722 302 269; E-mail: razvan.rughinis@cs.pub.ro)*
*** Faculty of Automatic Control and Computers, University Politehnica of Bucharest, Bucharest, Romania. (E-mail: laura.gheorghe@cs.pub.ro)*

Abstract: This article discusses a security solution for Wireless Sensor Networks that provides strong mutual authentication, anti-replay protection, confidentiality, integrity, and semantic security, while being reliable and energy-efficient: the Energy-efficient Authentication and Anti-Replay Security Protocol (EAASP). Mutual authentication is performed during a three-step handshake that establishes an authenticated connection. The protocol includes an anti-replay mechanism that binds packets to their context so that they cannot be re-used by attackers. Reliability is warranted by a packet recovery mechanism using negative acknowledgements. The security protocol has been implemented in TinyOS using nesC components and its functionality was evaluated using TOSSIM. An extensive comparative evaluation of five solutions recommends EAASP for critical applications in military, industrial and medical utilization.

Keywords: wireless sensor network; security; authentication; anti-replay; integrity; energy efficiency

1. INTRODUCTION

Wireless Sensor Networks employ embedded devices with low power, limited resources, low cost, and small dimensions, which self-organize into a network in order to collect data about the environment and to relay it towards a central device identified as the base station (Zheng et al., 2009). Applications that run on top of sensor networks can be divided into monitoring and tracking applications (Yick et al., 2008). Monitoring applications include environmental monitoring, health care, industrial process control, disaster detection, and many others. Sensor networks can also be used to track animals, humans or objects.

Depending on the application, sensor networks are designed to meet a set of variable priority requirements, including security, reliability, robustness, self healing, and scalability (Westhoff et al., 2006). Low energy consumption is a crucial constraint for sensor networks (Walters et al., 2006; Akyildiz et al., 2002), particularly for WSNs including devices that are difficult to replace – such as sensors situated in hard to reach environments, or inside the human body.

Wireless Sensor Networks are used in critical applications in fields such as military, health, and habitat monitoring. Attackers may intercept traffic, may inject malicious packets into the network or may otherwise interfere with network devices, in order to obtain data or to disrupt normal network functionalities (Kavitha et al., 2010). Still, traditional security methods cannot be directly applied to Wireless Sensor Networks because of the limited resources available on sensor nodes (Walters et al., 2006; Pathan et al., 2006). Therefore, dedicated security solutions must be designed to address the required security profiles.

The article discusses the Energy-efficient Authentication and Anti-replay Security Protocol (EAASP), a security solution that provides mutual authentication, anti-replay protection, integrity, confidentiality, semantic security and reliability. Development versions of the protocol were presented in Gheorghe et al. (2010a, 2010b) and in Gheorghe et al. (2011). The EAASP protocol jointly optimizes protection and energy efficiency for critical uses such as military, law enforcement, security, industrial control and medical applications.

The paper is structured as follows: section 2 introduces a range of alternative solutions that address a comparable security profile; section 3 presents the main protocol features; section 4 discusses EAASP implementation; section 5 presents test results; section 6 includes a comprehensive comparative evaluation of EAASP performance on multiple metrics, section 7 discusses the practicability of EAASP for a variety of application domains with specific security profiles, and section 8 concludes the article.

2. RELATED WORK

Each application has a specific profile of security requirements, which must be taken into consideration in the design of a security solution. The main requirements to be taken into account when implementing a WSN are: authenticity, integrity, confidentiality, non repudiation, freshness, availability, intrusion detection, and key management (Zheng et al., 2009; Wang et al., 2006). Other requirements that should be taken into consideration when developing a security protocol for WSNs are: reliability,

resiliency, flexibility, scalability, fault-tolerance, self-healing, assurance, transmission time, delay, and, last but not least, energy consumption (Wang et al., 2006; Karlof et al., 2004; Anastasi et al., 2009).

This paper addresses the following attacks which are relevant for our comparative evaluation of EAASP, starting from their presentation in Wang et al. (2006): packet injection and alteration (including the Sybil attack), packet replay (including the wormhole attacks), selective forwarding, Sinkhole, Blackhole, de-synchronization, and flood attacks. These attacks occur in the network and transport layers. Other important attacks may take place in the link and physical layers: jamming, tampering, eavesdropping, node capturing, collision, exhaustion, and unfairness.

Significant dedicated state-of-the-art security solutions for Wireless Sensor Networks include SPINS (Perrig et al., 2002), TinySec (Karlof et al., 2004), MiniSec (Luk et al., 2007), and CLIFFs (Sharma et al., 2011).

2.1 SPINS

SPINS was developed in 2002 by Perrig et al., and it consists of two building blocks: SNEP and μTESLA (Perrig et al., 2002; Boyle et al., 2008). SNEP provides confidentiality, authentication, integrity and freshness. μTESLA provides authenticated broadcast and emulates asymmetry by a delayed disclosure of symmetric keys. SPINS was implemented in TinyOS.

2.2 TinySec

TinySec is a link layer security architecture, developed in 2004 by Karlof et al., which provides confidentiality, authentication, integrity and semantic security (Karlof et al., 2004; Boyle et al., 2008). The authors chose not to include anti replay protection, deeming that it should be implemented at higher layers of the communication stack.

TinySec allows for two different security options: TinySec-Auth, which provides only authentication and TinySec-AE, which assures authenticated encryption. TinySec was designed to replace SNEP, and it is included in the TinyOS distribution.

2.3 MiniSec

MiniSec is a security solution developed by Luk et al. in 2007, which aims at high security while being energy-efficient (Luk et al., 2007). The security requirements it addresses are authentication, confidentiality and anti-replay protection. It has two modes of operation: single-source unicast communication – MiniSec-U, and multi-source broadcast communication – MiniSec-B.

2.4 CLIFFs

The Cross Layer Integrated Framework for Security (CLIFFs) for WSNs has been developed by Sharma and Ghose in 2011 (Sharma et al., 2011). CLIFFs includes an adaptive security protocol that dynamically adjusts its functionality to the security level that is required by the current network state.

3. ENERGY-EFFICIENT AUTHENTICATION AND ANTI-REPLAY SECURITY PROTOCOL

The three main sources of security vulnerabilities in WSNs are their limited energy resources, unreliable communication, and unattended operation (Walters et al., 2006). The Energy-efficient Authentication and Anti-Replay Security Protocol (EAASP) is a security protocol that provides strong, mutual authentication and reliability, ensuring anti-replay protection and message confidentiality, integrity and semantic security, with low energy and latency costs. Its security profile is designed to efficiently prevent attacks from external malicious nodes.

3.1 Data confidentiality

In EAASP, data confidentiality is provided by encrypting the packet payload using lightweight block cipher and operation mode. The payload is first concatenated with the sequence number and then encrypted, as represented in formula (1), in order to provide semantic security.

$$Payload_i = E(K, Msg_i||Seq) \quad (1)$$

3.2 Authentication, integrity and anti-replay protection

EAASP uses a single Message Authentication Code (MAC) in order to provide authentication and anti-replay protection.

The paper discusses an anti-replay method that consists in computing a MAC based on the contents of the previous message sent between the same source and destination and including this MAC in the current packet, as in formula (2). This way, we build chains of packets. We call this method 'binding the packet to its context'. If a packet is replayed, the MAC computed at the destination considering the previous packet is not equal to the MAC found in the packet, so the packet is dropped.

$$MAC_i = MAC(K, Msg_{i-1}) \quad (2)$$

The most efficient method to prevent packet alteration is to include in each packet a MAC computed as a function of the contents of the current packet itself. The attacker cannot re-compute a valid MAC after altering the packet because it does not know the secret key. The destination node computes the MAC based on the current packet contents and compares it with the one found in the packet. If the packet has been altered, the MACs are not equal and the packet is dropped.

In order to combine the anti-replay and the integrity protection methods, the MAC is computed based on the contents of both the previous and current packets. We also include the sequence number in the computation of the MAC in order to provide semantic security, as represented in formula (3).

$$MAC_i = MAC(K, Payload_i \,||\, Payload_{i-1}||\, Seq) \quad (3)$$

Therefore, packet *i* includes the MAC computed using message *i-1* and message *i*, as represented in Figure 1. When

receiving packet from node X, node Y computes the MAC based on the previous packet received from X and the current packet to verify the MAC found in the packet.

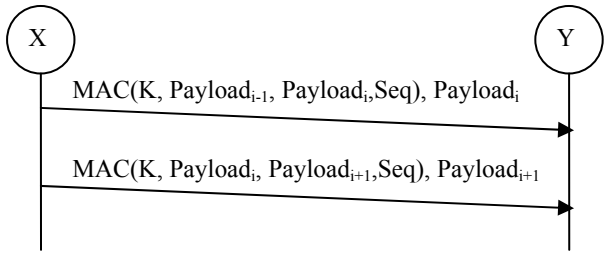

Fig. 1. EAASP anti-replay method.

This approach provides anti-replay protection since any replayed packet is rejected at the destination because of its invalid MAC. It provides integrity because altered messages are dropped at the destination. The method also provides authentication by using the secret key shared between source and destination in the MAC computation.

However, the first packet between source and destination is always accepted even if it is malicious. The solution we propose for addressing this problem is to perform mutual authentication prior to sending data packets.

3.3 Mutual authentication

Mutual authentication is completed by a 3-step handshake that establishes an authenticated connection, and it is required before any data packets can be exchanged. The handshake is represented in Figure 2. Node X initiates the connection by sending a H1 packet containing a random Challenge1. Node Y sends a H2 packet containing a random Challenge2 and the MAC computed from Challenge1 in order to authenticate itself. Node X also authenticates itself by sending the MAC computed from Challenge2.

When the mutual authentication is completed, the authenticated connection is established and data packets can be exchanged in both directions. The handshake and data packets include the MAC computed using formula (3).

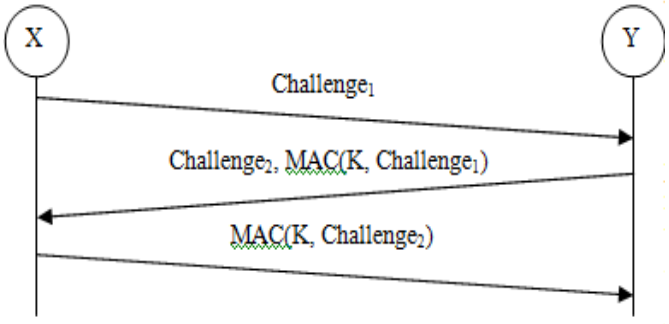

Fig. 2. Mutual authentication in EAASP.

During the mutual authentication process, if the other node does not respond for a predefined period of time, decided by the network administrator, the authentication fails and must be re-initiated. This provides protection against Denial of Service attacks based on open connections.

After mutual authentication is performed, a connection is created between the two nodes. The connection times out after a predefined period of time in which no packet is sent in any direction, and mutual authentication must be re-iterated when the two nodes want to communicate with each other.

The predefined timeout depends on the network dimension and the application and should be determined experimentally by the network administrator.

3.4 Communication Reliability

Packet loss is a common problem in WSNs, and it affects the security protocol because it de synchronizes the anti-replay mechanism. If packets are lost, the destination drops the subsequent packets because the MAC of the previous packet expected by the destination does not match the one found in the received packets.

EAASP accommodates two methods for providing reliability: positive and negative acknowledgements. However, for a more energy-efficient implementation of EAASP the use of negative acknowledgements is recommendable.

Positive acknowledgements (ACK) are used to announce that one or more packets have been received at the destination node. If the ACK is timely received, the next packet is sent. If not, the respective packets are marked as “lost” and they are resent; the procedure is then repeated.

Negative acknowledgements (NACK) are used when the destination detects packet loss. The destination detects a lost packet when it receives a packet with a sequence number greater than the expected one. The out-of-order packet is not dropped by the destination, but it is stored in order to be checked and used after the recovery of the lost packets.

When choosing between the two solutions, one must take into consideration energy consumption, reliability, and delay criteria. An evaluation of the energy efficiency of positive versus negative acknowledgments depends on the rate of packet loss; with low packet loss rates positive ACKs consume more energy than negative ACKs, while high packet loss causes the opposite effect.

As regards reliability, positive ACKs track the status of each packet or packet sets. Negative ACKs start tracking packet status only when packet loss is detected, and thus a large number of packets can be lost before the destination notices.

The delay introduced by the retransmission method depends on the number of lost packets. In the case of low packet loss, positive ACKs introduce a greater delay because they are sent more frequently and the source node has to wait for them. In the event that many consecutive packets are lost, negative ACKs can introduce a greater delay because the destination does not trigger the recovery action until it becomes aware of the packet loss.

In the next paragraphs EAASP with negative ACKs is discussed for efficiency reasons regarding energy consumption and delay in networks with low rates of packet loss. However, for a network with high packet loss, the use of positive ACKs is recommendable.

The payload of a NACK packet contains the sequence number of the lost packet L_Seq, and the sequence number of the received out-of-order packet R_Seq, as represented in Table 1.

Table 1. NACK packet structure

EAASP Header			Payload	
MAC	Type	Seq	L_Seq	R_Seq

The source node receives the NACK and resends the packet with the sequence number that is equal to L_Seq, and it does not resend the packet with the sequence number equal to R_Seq.

When the destination node receives a lost packet, it performs the integrity and anti-replay check and, if the packet passes the test, it sends it to the upper layers. The same process is repeated for the out-of-order packet in storage.

3.5 EAASP message format

The EAASP payload contains the initial encrypted data. The payload is represented in formula (3).

The EAASP message header contains the following fields: MAC (4 bytes), Type (1 byte) and Sequence (1 byte), as presented in Table 2.

Table 2. Header structure

Header structure	EAASP Header		
	MAC	Type	Seq
Number of bytes	4	1	1

The MAC field contains a Message Authentication Code that is computed from the contents of the previous packet, the contents of the current packet, the secret key and the sequence number, as represented in formula (4).

The Sequence number is used to detect lost packets and request them from the source node.

The Type field contains several other fields: Auth, Resent, H1/H2/H3/Data/NACK, and QoS, as represented in Table 3.

Table 3. Type field structure

Type field structure	Type field			
	Auth	Resent	H1/H2/H3/Data/NACK	QoS
Number of bits	1	1	3	3

The Auth field is 0 during the authentication handshake and 1 after the handshake; therefore, it indicates that the source and destination have successfully performed mutual authentication.

The Resent field is set to 1 only when the packet is re-sent through the packet recovery method. This field is used to mark packets that are transmitted in the recovery process.

According to EAASP design, a packet can be a handshake packet (H1, H2, H3), a Data packet or a NACK. This information is encoded on 3 bits.

The QoS field can be used to implement a QoS mechanism. The priority is represented on 3 bits; therefore, the administrator may define 8 levels of priority. For example, control packets, such as handshake, NACK, and Re-sent packets can receive greater priority than Data packets.

3.6 Collision resistance analysis

A collision attack occurs when a malicious message is generated with the same MAC as a legitimate message. The probability of having two different blocks of data with the same hash is 1 in 2^n, where n is the hash length. Therefore, the probability decreases exponentially with increasing hash length.

EAASP uses the HMAC method to compute 32-bit hashes. Therefore, the attacker should send packets with random MACs, on average, 232 times before forging a valid MAC. As the authors of TinySec state, it would take 20 months to send so many packets using a 19.2 kbps rate (Karlof et al., 2004). Therefore, EAASP is based on the consideration that a 4-byte MAC is sufficient for a resource constrained communication network. However, some critical applications may require a higher security level and hence a smaller collision probability, which can be provided by an 8-byte MAC.

4. EAASP IMPLEMENTATION

The Energy-efficient Authentication and Anti-Replay Security Protocol (EAASP) has been implemented in the communication stack of TinyOS, an open source, component based operating system that was especially developed for Wireless Sensor Networks (Levis et al., 2004).

The implementation was done using nesC components that have been connected with default TinyOS components, as represented in Figure 3.

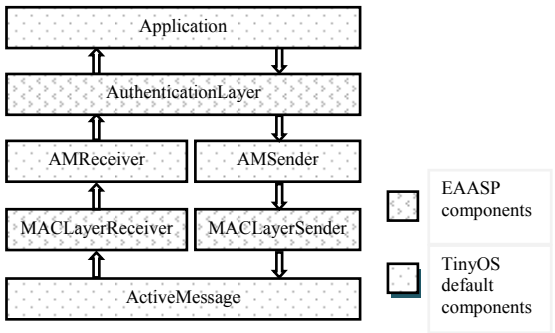

Fig. 3. EAASP implementation.

4.1 MAC Layer

The Message Authentication Code (MAC) Layer is implemented using the Communication System Hooks (CSH) that we placed between the Active Message layer and the AMReceiver and AMSender components.

CSH contain two main components: HookSender and HookReceiver. The HookSender component is able to analyze and alter any packet before it is sent by the current

node. The HookReceiver is able to inspect and modify any received packet received by the current node before reaching the Application layer. 4.1.1 MACLayerSender

In the HookSender component, packets are received from the AMSender component which is used by the Application layer to send packets to the destination. In the protocol implementation, the HookSender is designated as MACLayerSender, as observed in Figure 3.

When it receives a packet from the AMSender component, it first encrypts the payload. Then it computes a Message Authentication Code from the current payload, the payload of the previous packet, the secret key and the sequence number.

The current payload is then stored in this component in order to be used when sending the next packet.

After the MAC is computed and placed into the corresponding field, the packet is sent to the ActiveMessage component.

4.2 MACLayerReceiver

The HookReceiver component receives all packets that have as destination the current node from the ActiveMessage layer and sends them to the AMReceiver component. In the EAASP implementation, the HookReceiver is designated as MACLayerReceiver.

When a packet is received from the ActiveMessage layer, the validity of the MAC value and the sequence number contained in the packet are checked.

If the MAC is invalid and the sequence number is lower than or equal to the expected one, the packet is dropped. This way, the component rejects replayed and invalid packets.

If the MAC is invalid but the sequence number is greater than the expected one, the packet might be an out-of-order packet. This packet is stored in the component and it is checked and delivered to the upper layer after the expected packets are received.

If the MAC is valid and the sequence number is the expected one, the payload is decrypted and the packet is sent to the AMReceiver component. The payload of the current packet is stored in order to be used when verifying the MAC of the next packet.

The component maintains in a variable the sequence number of the last received valid message from each source node. This variable is used in order to detect the out-of-order packets.

4.3 Authentication Layer

The AuthenticationLayer performs the authentication handshake, keeps track of sequence numbers, buffers packets and sends ACK/NACK packets. It is placed between the Application and the AMSender/AMReceiver components.

When the Application wants to send packets, the AuthenticationLayer initiates the three step handshake with the destination. After the handshake has been performed and the authenticated connection has been established, the data packets can be delivered.

When an out-of-order packet is received, the Authentication layer builds and sends a NACK packet that contains the sequence number of the expected packet, as well as the sequence number of the out-of-order received packet.

When the AuthenticationLayer receives a NACK packet, it starts resending all the packets with a sequence number greater and equal to the expected value, but less than the out-of-order sequence number.

After all those packets are delivered to the AuthenticationLayer at the destination, the MACLayerReceiver checks and delivers the out-of-order packet.

The source node stores a predefined number of packets in a buffer, providing the opportunity for packet retrieval when packets are lost. When receiving a NACK packet, which requests a sequence number of a packet that is not in the buffer anymore, the missing packet cannot be retrieved and delivered to the destination any longer. At that moment the connection is closed and re-initiated, because it has been de-synchronized.

The AuthenticationLayer is responsible for creating and closing the authenticated connection, for requesting lost packets, and for storing sent packets and retrieving and delivering them upon request.

4.4 Cryptographic primitives

The MAC can be implemented by using a cryptographic hash function, for example HMAC, by using a block cipher algorithm, for instance CBC-MAC and CMAC, or by using universal hashing, such as UMAC and VMAC. Lee et al. compare several MAC methods and conclude that HMAC is the most efficient from the point of view of energy consumption and memory occupation (Lee et al., 2010). Therefore, HMAC was chosen, in this paper, for implementing the MAC in EAASP security protocol, for efficiency reasons.

Some of the most well-known block ciphers are AES, DES, RC5 and Skipjack. Lee et al. determined experimentally that Skipjack is the most efficient block cipher as regards energy consumption, RAM, encryption and decryption time (Lee et al., 2010). Skipjack is also the default algorithm used in TinySec and MiniSec. Karlof et al. consider that both RC5 and Skipjack are appropriate for sensor networks but they prefer Skipjack for its low key setup cost, and because RC5 is patented (Karlof et al., 2004).

The National Institute for Standards and Technology (NIST) recommends the following operation modes: Electronic codebook (ECB), Cipher-block chaining (CBC), Cipher feedback (CFB), Output feedback (OFB), and Counter (CTR). Lee et al. determined experimentally that CBC, OFB, and CFB are the most energy-efficient operation modes. CBC and CFB have the advantage of tolerating repeatable Initialization Vectors (IVs). However, CBC has the disadvantage of producing an output larger than the input and

has a decryption cost that is much higher than the encryption cost (Lee et al., 2010). Therefore, Skipjack was chosen in this work as the default block cipher and CFB as its operation mode, in order to efficiently encrypt data while minimizing energy costs.

5. EXPERIMENTAL EVALUATION

The Energy-efficient Authentication and Anti-Replay Security Protocol has been tested with TOSSIM, a simulator for TinyOS applications (Levis et al., 2003). Initial development versions of the protocol were evaluated in Gheorghe et al. (2010a, 2010b) and Gheorghe et al. (2011).

A single hop example of TOSSIM output for EAASP functioning is presented in Figure 4. Each line has the following format: The ID of the node, the component that generates the output, the type of packet sent or received, the fields, the source and destination of the packet. In Figure 4 the authentication handshake takes place, and a data packet is sent by node 3 and received by node 1. As it can be observed in the output, the Authentication layer is responsible for performing the handshake and building the authenticated connection. Afterwards, the Application layer is responsible for sending and receiving data.

```
(3): AuthenticationLayer: H1 packet sent [payload=234 type=8 seq=1 (3 ->1)]
(1): AuthenticationLayer: H2 packet sent [payload=57195 type=16 seq=1 (1 ->3)]
(3): AuthenticationLayer: H3 packet sent [payload=56185 type=24 seq=2 (3- >1)]
(3): AuthenticationLayer: Managed to authenticate myself to node 1
(1): AuthenticationLayer: Managed to authenticate myself to node 3
(3): ApplicationC: Data packet sent [payload=1235 type=32 seq=3 (3 ->1)]
(1): ApplicationC: Data packet received [payload=1235 type=32 seq=3 (3 ->1)]
```

Fig. 4. Handshake and data packets.

Figure 5 demonstrates communication reliability.

```
(3): ApplicationC: Data packet sent [payload=1243 type=32 seq=11 (3 ->1)]
(3): ApplicationC: Data packet sent [payload=1244 type=32 seq=12 (3 ->1)]
(1): AuthenticationLayer: Out of order packet received [payload=1244 type=48 seq=12 (3 ->1)]
(1): AuthenticationLayer: NACK packet sent [payload=11 type=40 seq=2 (1 ->3)]
(3): AuthenticationLayer: NACK packet received [payload=11 type=40 seq=2 0 (1 ->3)]
(3): AuthenticationLayer: Data packet resent [payload=1243 type=92 seq=11 (3 ->1)]
(1): ApplicationC: Data packet received [payload=1243 type=92 seq=11 (3 ->1)]
(1): ApplicationC: Data packet received [payload=1244 type=32 seq=12 (3 ->1)]
(3): ApplicationC: Data packet sent [payload=1245 type=32 seq=13 (3 ->1)]
(1): ApplicationC: Data packet received [payload=1245 type=32 seq=13 (3 ->1)]
```

Fig. 5. Response to lost data packets.

A packet with the sequence number equal to 11 is lost and the next packet is received by the destination. It is detected as an out-of-order packet and a NACK is built and delivered to the source node. The NACK is received by the source node, which retrieves and then delivers the lost data packet. The out-of-order data packet is also delivered by the MACLayer to the application. Afterwards, traffic continues normally; the packet with the sequence number equal to 13 is correctly delivered to the destination.

6. COMPARATIVE EVALUATION

This section compares EAASP with several other security solutions, taking into consideration security, energy consumption, delay, and control overhead. Four security solutions that provide comparable services were included in the evaluation: SPINS (Perrig et al., 2002), TinySec with its two versions TinySec-Auth and TinySec-AE (Karlof et al., 2004), MiniSec (Luk et al., 2007) and CLIFFs (Sharma et al. 2011). SPINS and TinySec are the most mature solutions for sensor networks, having been extensively tested in the academic environment and in industrial applications. Alternatively, MiniSec and CLIFFs are more recent solutions, but are more efficient and incorporating additional functionalities. Therefore, the analysis includes these solutions in the EAASP comparative evaluation, based on their similar security profiles and because they have variable maturity levels and diverse features.

In sensor networks there is always a trade-off between security and energy efficiency. For this reason, the first comparison refers to security metrics for analyzed solutions, and the second comparison refers to their energy consumption.

6.1 Security analysis

As Wang et al. propose in their survey, security should be evaluated using a number of requirements: authentication, integrity, freshness, confidentiality, non-repudiation, availability (Wang et al., 2006). Semantic security is included in this list, as it is implemented in several selected security protocols. An overview of the evaluation is presented in Table 4.

6.1.1 Authentication

SNEP is a component of SPINS that provides one way authentication through computing a Message Authentication Code (Perrig et al., 2002). The MAC has 8 bytes and is computed using the CBC-MAC method. The key used in the computation is generated from the Master Key that is pre-provisioned on the nodes and shared with the base station. The MAC is built from the encrypted packet concatenated with a counter.

Both TinySec-Auth and TinySec-AE use a MAC to provide one way authentication (Karlof et al., 2004). The same method, CBC-MAC, is used, but the result is 4 bytes in length.

MiniSec-U is the unicast communication component of MiniSec and uses OCB method to provide one-way authentication (Luk et al., 2007). OCB computes a ciphertext from the initial message, key, and nonce. It also generates a 4-byte tag that is verified by the destination after decrypting the text. The tag provides the possibility to verify the authenticity of the packet.

CLIFFs assures data authentication through the use of a 4-byte MAC (Sharma et al. 2011). However, the authors do not specify the method used to compute this MAC.

EAASP provides authentication through the use of a 4-byte MAC, computed by the HMAC method. In order to strengthen the authentication and anti-replay protection, an authenticated connection is established by a 3-step handshake performing mutual authentication. The connection includes the flow of packets that are bound together. A packet cannot be injected or replayed in an already established connection, and the connection cannot be established until both sides have authenticated themselves.

6.1.2 Integrity

A CRC or MAC can be used to check the integrity of a message. The original TinyOS packet contains a two-byte CRC. SNEP, TinySec-Auth, TinySec-AE, CLIFFs, and EAASP use the MAC to verify message integrity. MiniSec-U uses the OCB tag to provide data integrity.

6.1.3 Freshness

Freshness is usually provided by using a counter or sequence number in cryptographic operations. SNEP provides weak freshness because it includes a counter in the MAC computation. The counter is not included in the message but it is known and incremented by both sides. However, this can cause de-synchronization problems when packets are lost. The authors use a protocol to synchronize the counters on both sides.

Neither TinySec-Auth, nor TinySec-AE provides any freshness because the authors decided that it should be implemented at higher levels. This decision makes TinySec more lightweight but less secure. CLIFFs does not specify any anti-replay protection either.

MiniSec-U provides weak freshness by including a counter in the OCB computation. The counter is kept synchronized by both the sender and the receiver. The receiver accepts only messages with a higher counter; therefore, it assures anti-replay protection.

EAASP provides anti-replay protection and weak freshness by binding the message to its context: it computes the MAC based on the contents of the previous packet sent between the same source and destination in a certain authenticated connection, based on the sequence number of the current packet and based on the contents of the current packet. This makes it very difficult for an attacker to find a packet with a valid MAC in order to replay it in the current conversation.

6.1.4 Confidentiality

The standard way to provide confidentiality is through encryption. Two cryptographic primitives used in the encryption process are the encryption algorithm and the operation mode.

SNEP provides confidentiality with the RC5 encryption method. However, this basic confidentiality is not sufficient in a context with repeated values, in which an attacker can easily map plaintext with the equivalent ciphertext. Therefore, semantic security is needed for sensor networks. Semantic security is provided by using the block cipher in counter mode (CTR mode) and a counter that is shared between the sender and the receiver. The counter is incremented after each message, and therefore the encryption of the same value is different every time. For security reasons, the key that is used for encryption is different from the key used for computing the MAC. This key is also derived from the Master key. A different key and counter are used for each communication direction.

TinySec-AE uses Skipjack as the default block cipher because, although RC5 is faster, Skipjack is more energy-efficient, has a lower memory footprint, and is not patented. The authors have chosen the CBC operation mode because it is more appropriate for block ciphers. They have slightly modified the functionality of the CBC by encrypting the IV before performing the actual encryption. Semantic security is assured through the use of non-repeating IVs. TinySec-Auth does not provide confidentiality or semantic security.

MiniSec-U also uses Skipjack as the underlying block cipher. OCB is a block cipher operation mode that provides authenticated encryption. The ciphertext is obtained from the message payload, counter, and secret key. The counter is used as non-repeating nonce and provides semantic security.

EAASP also uses Skipjack as the underlying block cipher and the CFB operation mode because of their high efficiency and performance. The protocol provides semantic security by using the sequence number when encrypting the payload and when computing the MAC.

CLIFFs uses four methods for providing confidentiality, associated with the four levels of security specified by the authors: Simple XOR, RC5/80/4, RC5/40/8 and RC5/40/12. The encryption algorithm is RC5, the key size is 80 bytes, and the number of rounds is 4, 8 or 12, depending on the security level. A larger number of rounds imply a higher security level.

6.1.5 Non-repudiation

Non-repudiation is usually provided by digital signatures or trusted third party (TTP). These security methods are not appropriate for low power devices, so they were not implemented in any of the selected protocols.

6.1.6 Availability

A large range of attacks target the availability of the sensor network, such as selective forwarding, Sinkhole attacks, Blackhole attacks, flood attacks, malicious data injection, radio jamming, etc.

Sinkhole, Sybil, alteration, and injection attacks are mitigated by every solution included in the evaluation. Replay and Wormhole attacks are blocked only by SNEP, MiniSec-U and EAASP. Blackhole attacks, selective forwarding and de-synchronization attacks are mitigated only by EAASP. Flood, jamming, tampering, collision and exhaustion attacks are not addressed by any of the compared protocols.

EAASP provides a high level of service availability by its authentication and integrity mechanisms, anti-replay protection, and packet recovery mechanism.

6.1.7 Other security metrics

Other metrics that are useful for comparing security protocols are: reliability, resiliency, flexibility, scalability, fault-tolerance, self-healing, assurance and energy efficiency.

SPINS, TinySec-Auth, TinySec-AE and CLIFFs do not provide any kind of reliability. The authors of MiniSec provide a possibility for counter re-synchronization that uses acknowledgements and packet recovery. EAASP implements packet loss detection and recovery in order to avoid de-synchronization, using negative acknowledgements.

Resiliency mainly depends on key management. If one single secret key is used for securing communications in the entire network, the protocol is not resilient to node capturing. However, if only the end nodes (source and destination) know the secret key used to secure the traffic between them, the security protocol is resilient to such attacks. SNEP uses a master key which is shared between the source and destination nodes and which is used to derive the encryption key and the MAC key. TinySec-Auth, TinySec-AE, MiniSec-U and EAASP use symmetrical keys that are shared between the source and destination nodes. CLIFFs uses 4 types of keys: Buddy key (Kb), My-Own-Key (Ko), Network key (Kn) and Broadcast key (Kbro). Therefore, all selected security protocols are resilient to compromised nodes.

None of the selected protocols depend on the network topology or other context factors, therefore they provide flexibility. For all selected security protocols, including EAASP, the number of hops does not influence the degree to which security requirements are satisfied by the protocol.

Both energy consumption, which is mainly determined by transmission costs, and latency depend linearly on the number of hops. Therefore, the selected protocols are scalable.

EAASP is fault-tolerant when packets are lost or altered by hardware, software, or transmission faults, because of its packet recovery, authentication, and integrity mechanisms. MiniSec authors recommend the use of packet recovery based on negative acknowledgments but this is not actually integrated into the protocol.

All selected protocols are tolerant to packet altering caused by transmission faults. SNEP includes a counter synchronization protocol used in case of packet loss which re-configures the counters. The authors of MiniSec suggest that a packet recovery mechanism can be used to recover from packet loss. CLIFFs includes a re-election mechanism that is performed when a current cluster head is out of energy.

Table 4. Protocol comparison on security requirements

Metrics		**SNEP (SPINS)**	**TinySec-Auth**	**TinySec-AE**	**MiniSec-U**	**CLIFFs**	**EAASP**
Security Requirements	Authentication	One way, CBC-MAC, 8 bytes	One way, CBC-MAC, 4 bytes	One way, CBC-MAC, 4 bytes	One-way, OCB tag, 4 bytes	One-way, MAC	Mutual, HMAC, 4 bytes, authenticated connection
	Integrity	MAC	MAC	MAC	OCB tag	MAC	MAC
	Freshness	Counter	No	No	Counter	No	Previous packet, sequence number
	Confidentiality	Encryption, RC5, CTR	No	Encryption, Skipjack, CBC	Encryption, Skipjack, OCB	Encryption, RC5	Encryption, Skipjack, CFB
	Semantic Security	CTR, Counter	No	Random IVs	Counter	No	Sequence number
	Non-repudiation	No	No	No	No	No	No
	Availability	Partial	Partial	Partial	Partial	Partial	Partial
Other Requirements	Reliability	No	No	No	Possible	No	Yes
	Resiliency	Yes	Yes	Yes	Yes	Yes	Yes
	Flexibility	Yes	Yes	Yes	Yes	Yes	Yes
	Scalability	Yes	Yes	Yes	Yes	Yes	Yes
	Fault -tolerance	Altered packets	Altered packets	Altered packets	Altered packets	Altered packets	Lost and altered packets
	Self-healing	Counter synchronization protocol	No	No	No	Re-election of cluster head	Packet recovery, connection re-establishment
	Assurance	No	No	No	No	Yes	No
Energy-efficiency		Low	Good	Low	Medium	Medium	Medium

TinySec does not include re-configuration mechanisms. In EAASP, if a node loses the information regarding its authenticated connection, it re-initiates the 3-step handshake to perform mutual authentication and re-establish the connection. In our protocol, packet loss is handled using a packet recovery mechanism.

Only CLIFFs includes adaptive security services adjustable as regards security requirements, thus providing assurance. None of the other protocols accommodate diverse user preferences.

6.1.8 Attack mitigation

The comparative evaluation considers several frequent attacks presented above, following Wang et al.: packet injection and alteration, replay attack, selective forwarding, Blackhole attack, Sinkhole attack, Sybil attack, Wormhole attack, de-synchronization attack, flood attack, and physical and link layer attacks. An overview of the comparison is included in Table 5.

All selected protocols are able to mitigate packet injection and alteration. Defense is accomplished by authentication and integrity mechanisms: SNEP, TinySec, CLIFFs and EAFASP use MACs, while MiniSec-U uses the OCB tag for packet verification. The protocols use authentication and integrity mechanisms to mitigate Sybil attacks when they are generated by external attackers. All selected protocols can address the Sinkhole attack produced by a malicious node by the authentication of routing control packets.

Replay attacks, including Wormhole attacks, are mitigated by SNEP and MiniSec-U by authentication and freshness mechanisms: the MAC and OCB tag that are computed using a counter and the secret key shared between source and destination. EAASP uses a sequence number and the contents of the previous packet in the MAC computation to mitigate replay attacks. EAASP provides protection against Wormhole attacks by mutual authentication between source and destination and the anti-replay mechanism. TinySec and CLIFFs do not provide any protection against replay attacks.

Selective forwarding and Blackhole attacks cannot be mitigated by the selected protocols, except for EAASP, which addresses them by its packet recovery mechanism. When a packet is lost or dropped by a malicious node, it is re-sent by the destination. However, if the packet takes the same route, there is a possibility to be dropped again by the malicious node. A routing protocol should be used to take into consideration route failure and re-route the traffic in such instances.

EAASP protects effectively against a Denial of Service attack based on open connections by using time-outs. However, none of the selected protocols, including EAASP, is designed to mitigate flooding attacks. The problem can be easily solved for EAASP by including a Storm Control Mechanism in the MAC Layer (Rughinis and Gheorghe, 2010).

EAASP is the only one to establish a security connection between the source and destination nodes. A connection in EAASP can be terminated only when no messages are exchanged by the involved nodes. Therefore, other nodes cannot disrupt an already established connection, so EAASP mitigates de-synchronization attacks.

Table 5. Protocol comparison on attack mitigation

Attack Layer	Mitigated Attacks	SNEP (SPINS)	TinySec	MiniSec-U	CLIFFs	EAASP
Network and transport layers	Packet Injection	Yes	Yes	Yes	Yes	Yes
	Packet Alteration	Yes	Yes	Yes	Yes	Yes
	Sinkhole Attack	Yes	Yes	Yes	Yes	Yes
	Sybil Attack	Yes	Yes	Yes	Yes	Yes
	Replay Attack	Yes	No	Yes	No	Yes
	Wormhole Attack	Yes	No	Yes	No	Yes
	Selective Forwarding	No	No	No	No	Yes
	Blackhole Attack	No	No	No	No	Yes
	De-synchronization Attack	No	No	No	No	Yes
	Flood Attack	No	No	No	No	No
Link and physical layers	Collision Attack	No	No	No	No	No
	Jamming	No	No	No	No	No
	Tampering	No	No	No	No	No
	Exhaustion	No	No	No	No	No

The selected protocols do not target protection against jamming, tampering, collision and exhaustion. Mechanisms against such threats should be implemented in hardware or at the low level of the operating system (Wang et al., 2006).

As discussed in section 6.2 on energy consumption, SPINS and TinySec-AE provide low efficiency, TinySec-Auth provides good efficiency and MiniSec-U, CLIFFs and EAASP provide medium efficiency. However, the good efficiency of TinySec-Auth comes with the price of not providing confidentiality and semantic security.

6.2 Energy consumption, delay, and control overhead analysis

The following section analyzes the selected protocols in terms of control overhead, transmission-related energy consumption, transmission time, and delay (see Table 6 for an overview). For energy evaluation the results of Amiri (2010) are used, who determined that each received or sent byte consumes 0.12mJ on the Tmote Sky/TelosB. Thus, comparison requires first the determination of the protocol overhead in terms of number of bytes, and then the transformation of this value into energy.

The authors of TinySec use the byte time concept to evaluate the overhead of their protocol (Karlof et al., 2004). The byte time is the duration of transmission of a single byte over the radio, for example, amounting to 0.42 ms on Mica 2. Following this practice, the comparison relies on the conversion of protocol overhead in byte time to estimate the delay introduced by the protocols. The value for Tmote Sky/TelosB is used, which has a rate of 250 kpbs, so the byte time is 0.004 ms.

The first step of the analysis consists in determining the control overhead of the selected protocols. The authors of TinySec use TinyOS packets with a 24-byte payload and a 39-byte packet overhead that includes headers (7 bytes) and media access control information (a 28-byte start symbol and additional synchronization bytes). The default TinyOS packet contains a group address (1 byte) and CRC (2 bytes), which are removed in the implementation of TinySec, MiniSec, and EAASP, because they are not used.

TinySec-Auth adds only a MAC of 4 bytes, while TinySec-AE adds Src (2 bytes), Ctr (2 bytes) and MAC (4 bytes). The difference between TinySec-Auth and the default TinyOS packet is only 1 byte, while for Tiny-AE the difference is 5 bytes.

SPINS adds an 8-byte MAC so it does not need the CRC field, therefore the overhead is 6 bytes, as implemented by the authors.

MiniSec adds a SrcAddr (2 bytes) and a MIC (4 bytes). The difference between MiniSec and the default TinyOS packet is 3 bytes.

EAASP introduces the fields Type (1 byte), Seq (1 byte) and MAC (4 bytes). Therefore, EAASP introduces 3 additional bytes, when compared with the default TinyOS packet.

CLIFFs header has the following format: Dst (1 byte), Src (1 byte), Ctr (1 byte), MAC (4 bytes), while removing the 2-byte Dst and 2-byte CRC from the TinyOS packet and keeping the 1-byte AM and 1-byte Len fields. The difference between CLIFFs and the default TinyOS packet is 3 byes.

Table 6 presents the protocol overhead (bytes), total packet overhead (bytes), total packet size (bytes), the total energy (mJ), the protocol overhead (mJ, %), the total transmission time (ms) and latency (ms) for each compared protocol. The default packet overhead is considered to be 39 bytes.

The conclusion of this comparison is that TinySec-Auth is the most energy-efficient protocol and introduces the least delay. The next protocols in order of efficiency are MiniSec, EAASP and CLIFFs. They are followed by TinySec-AE and SNEP.

While TinySec-Auth has the lowest costs in terms of energy and time, it provides neither confidentiality, nor anti-replay protection. MiniSec and EAASP are efficient and provide both confidentiality and anti-replay. CLIFFs is efficient and includes adaptive security.

TinySec-AE is more efficient than SNEP, providing confidentiality, but not anti-replay protection. The least efficient protocol is SNEP, mostly because it uses an 8-byte MAC.

Table 6. Energy consumption, latency and protocol overhead introduced by the compared protocols

Security Protocol	Protocol Overhead (bytes)	Total Packet Overhead (bytes)	Total Packet Size (bytes)	Total Energy (mJ)	Protocol Overhead (mJ)	Protocol Overhead	Total Transmission Time (ms)	Latency (ms)
No protocol	-	39	63	7.56	-	-	0.252	-
TinySec-Auth	1	40	64	7.68	0.12	1.58%	0.256	0.004
MiniSec	3	42	66	7.92	0.36	4.76%	0.264	0.012
EAASP	3	42	66	7.92	0.36	4.76%	0.264	0.012
CLIFFs	3	42	66	7.92	0.36	4.76%	0.264	0.012
TinySec-AE	5	44	68	8.16	0.60	7.93%	0.272	0.020
SNEP	6	45	69	8.28	0.72	9.52%	0.276	0.024

If the cryptographic operations involved in the security protocol are also considered, the evaluation conclusions do not change because processing operations consume much less energy than packet transmission. SNEP and CLIFFs use RC5, while MiniSec, EAASP and TinySec-AE use Skipjack as block ciphers. For encrypting 64 bytes, Skipjack consumes 0.026 mJ and RC5 consumes 0.114 mJ according to Lee et al. (Lee et al., 2010). Although there is a significant efficiency difference between the two block ciphers, neither of them influences the previous efficiency classification.

7. DISCUSSION

Relying on Law and Havinga's work which systematically specifies security threats associated with each application domain (Law et al., 2005), it is possible to evaluate how appropriate EAASP is for specific application areas. Military, law enforcement and other security applications need to be protected against attacks that target service availability, confidentiality, integrity and authenticity: Denial of Service attacks, eavesdropping of classified information, injection of misleading information. Arora et al. add reliability and energy-efficiency as requirements for intrusion detection applications (Arora et al., 2004). EAASP meets the security profile of military, law enforcement, and security applications, and it is more adequate than the other selected protocols because it provides protection against selective forwarding and Blackhole attacks, which are critical Denial of Service attacks.

Industrial applications require service availability, confidentiality and integrity for protection against eavesdropping on commercial secrets, disruption of the manufacturing process, and misleading sensor readings. In some cases, such as sensors attached to workers that monitor the radiation level to which they are exposed, workers' privacy must also be assured. EAASP is adequate for the security profile of industrial applications because it provides protection against a large range of attacks that could affect the industrial process.

Disaster detection and relief, agricultural, and environmental monitoring applications require data authenticity and integrity. Their security requirements are met by simple and more energy-efficient security protocols, such as TinySec-Auth, and do not require a higher level of protection such as the one provided by EAASP.

Medical applications have a security profile that includes authenticity and integrity in order to assure valid sensor readings. They also require reliability, patient privacy, and energy efficiency. Reliability ensures that the data about the patient is delivered to the medical personnel, patient privacy protects against disclosure of personal information to other recipients than the specialized personnel, and energy efficiency is particularly necessary when devices are implanted inside a patient's body and batteries cannot be changed without surgery. EAASP is the most appropriate protocol for such applications within the range of the evaluated solutions, because it completely meets the security profile of such applications.

8. CONCLUSIONS

This paper discusses and evaluates an efficient security solution for Wireless Sensor Networks, the Energy-efficient Authentication and Anti-replay Security Protocol (EAASP), which economically provides high protection against specialized WSN attacks in the network and transport layers.

The comparative evaluation of EAASP in relation to four alternative solutions, SPINS, TinySec, MiniSec, and CLIFFs, indicates that it sustains a comparatively high security level by mutual authentication, anti-replay protection, confidentiality, integrity, semantic security and reliability, with significant efficiency in terms of energy consumption, latency and protocol overhead. Its comparative advantages consist in strong authentication, anti-replay protection, and reliability, as well as specific attack mitigation. Therefore EAASP is a protocol of choice for applications with concurrent high security demands and efficiency requirements.

EAASP provides protection against external malicious nodes, and it is not efficient against internal malicious or faulty nodes. EAASP may be integrated with a trust management solution in order to extend its security profile.

Future work will address a QoS mechanism that uses the reserved bytes in the protocol header. The mechanism could be used to give priority to critical data, control packets, or re-sent data packets. It is also possible to modify the EAASP solution so that it can take into consideration several local factors, including energy levels, in order to situationally adapt its security performance. When the energy level is high, the protocol would provide the maximum level of protection; the security level would be decreased as a function of available energy levels. Security could also be increased when threats are identified by an integrated intrusion detection system.

ACKNOWLEDGEMENTS

This work was supported by the research program "EXCEL – Excellence in research by postdoctoral programs in priority domains of a knowledge-based society" ("Excelenta in cercetare prin programe postdoctorale in domenii prioritare ale societatii bazate pe cunoastere"), Contract no. POSDRU/89/1.5/S/62557 and by the Sectoral Operational Programme Human Resources Development 2007-2013 of the Ministry of European Funds through the Financial Agreement POSDRU/159/1.5/S/134398.

We are grateful for the attentive and valuable comments received from the two anonymous referees.

REFERENCES

Akyildiz, I.F., Su, W., Sankarasubramaniam, Y., and Cayirci, E. (2002). Wireless sensor networks: a survey, *Computer Networks*, vol. 38, pp. 393–422.

Amiri, M. (2011). *Wireless Sensor Networks: Evaluation of Power Consumption and Lifetime Bounds*, LAP Lambert Academic Publishing, pp. 1-60.

Anastasi, G., Conti, M., Di Francesco, M., and Passarella, A. (2009). Energy conservation in Wireless Sensor Networks: A survey, *Ad Hoc Networks*, vol. 7, pp. 537-568.

Arora A. et al. (2004) A line in the sand: A wireless sensor network for target detection, classification, and tracking, *Computer Networks*, vol. 46, pp. 605–634.

Boyle D., and Newe, T. (2008). Securing wireless sensor networks: Security architectures, *Journal of Networks*, vol. 3, pp. 65-77.

Gheorghe, L., Rughiniş, R., Deaconescu, R. and Ţăpuş, N. (2010a). Authentication and Anti-replay Security Protocol for Wireless Sensor Networks, *5th International Conference on Systems and Networks Communications, (ICSNC 2010)*, pp. 7-13.

Gheorghe, L., Rughiniş, R., Deaconescu, R., and Ţăpuş, N. (2010b). Reliable Authentication and Anti-replay Security Protocol for Wireless Sensor Networks, *2nd International Conferences on Advanced Service Computing*, pp. 208-214.

Gheorghe, L., Rughinis, R., and Tapus, N. (2011) Energy-efficient Optimizations of the Authentication and Anti-replay Security Protocol for Wireless Sensor Networks, *The 7-th International Conference on Networking and Services (ICNS 2011),* pp. 201-207

Karlof, C., Sastry, N., and Wagner D. (2004). TinySec: a link layer security architecture for wireless sensor networks, *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys '04)*, pp. 162-175.

Kavitha, T, and Sridharan, D. (2010). Security Vulnerabilities In Wireless Sensor Networks: A Survey, *Journal of Information Assurance and Security*, vol. 5, pp. 31-44.

Law, Y.W., and Havinga, P.J.M. (2005). How to Secure a Wireless Sensor Network, *International Conference on Intelligent Sensors, Sensor Networks and Information Processing*, pp. 89-95.

Lee, J., Kapitanova, K., and Son, S.H. (2010). The price of security in wireless sensor networks," *Computer Networks*, vol. 54, pp. 2967-2978.

Levis, P., Lee, N., Welsh, M., and Culler, D. (2003). TOSSIM: accurate and scalable simulation of entire TinyOS applications, *Proceedings of the first international conference on Embedded networked sensor systems (SenSys 2003)*, pp. 126-137.

Levis P. et al. (2004). TinyOS: An operating system for sensor networks, *Ambient Intelligence*, W. Weber, J. M. Rabaey, E. Aarts (Eds), pp. 115-148.

Luk, M., Mezzour, G., Perrig, A., and Gligor, V. (2007). MiniSec: A Secure Sensor Network Communication Architecture, *6th International Symposium on Information Processing in Sensor Networks*, pp. 479-488.

Padmavathi G. and Shanmugapriya D. (2009). A Survey of Attacks, Security Mechanisms and Challenges in Wireless Sensor Networks, *International Journal of Computer Science and Information Security*, vol. 4, pp. 117-25.

Pathan, A.-S.K., Lee, H.-W. and Hong, C.S. (2006). Security in Wireless Sensor Networks: Issues and Challenges, *Proceedings of 8th IEEE International Conference on Advanced Communications Technology (ICACT)*, vol. 2, pp. 1043-1048.

Perrig, A. Szewczyk, R. Tygar, J. Wen, V. and Culler, D.E. (2002). SPINS: Security protocols for sensor networks, *Wireless Networks*, vol. 8, pp. 521-534.

Rughinis, R., and Gheorghe, G. (2010), Storm Control Mechanism in Wireless Sensor Networks, *9th RoEduNet IEEE International Conference*, pp. 430–435.

Sharma K., and Ghose, M.K. (2011). Cross Layer Security Framework for Wireless Sensor Networks, *International Journal of Security and its Applications*, vol. 5, pp. 39-52.

Walters, J.P., Liang, Z., Shi, W. and Chaudhary, V. (2006). Wireless sensor network security: A survey, *Security in Distributed, Grid, and Pervasive Computing*, Y. Xiao, Ed. Auerbach Publications, CRC Press, pp. 1-50.

Wang, Y., Attebury, G. and Ramamurthy, B. (2006) A survey of security issues in wireless sensor networks, *IEEE Communications Surveys and Tutorials*, vol. 8, pp. 2-23.

Westhoff, D., Girao, J., and Sarma, A. (2006) Security Solutions for Wireless Sensor Networks, *NEC Technical Journal*, vol. 1, pp. 2-6.

Yick, J., Mukherjee, B,. and Ghosal, D. (2008). Wireless sensor network survey, *Computer Networks*, vol. 52, pp. 2292-2330.

Zheng, J. and Jamalipour A. (2009). *Wireless Sensor Networks: A Networking Perspective*, Wiley-IEEE Press.