ISSN: 2766 - 7715

Article

Encapsulate Sec: A Link-Layer Security Architecture for Wireless Sensor Networks

Bassam. W. Aboshosh[1]*, Rabie. A. Ramadan[2], Ayman El-Sayed[3], and Mohamed M. Dessouky[4]

[1] Department of Computer Engineering, Higher Institute of Engineering, Elshorouk Academy, Egypt; bassam.ahmed32@gmail.com
[2] Department of Computer Engineering, Menofia University, Egypt
[3] Computer Engineering Dept. Cairo University, Cairo, Egypt
** Correspondence: bassam.ahmed32@gmail.com

Received January 15th, 2019; Accepted: March 29th, 2019; Published: April, 5th, 2019

Abstract: Communication energy is the primary cause of power consumption for WSNs. All of the encryption protocols suggested for WSNs in the literature are designed to reduce security-related communication overheads in several ways to save communication resources. Therefore; all emerging energy-efficient protocols affect security resistance factors. However, maintaining communications security in WSNs is also critical. another challenge is the fact that WSNs suffer from limited bandwidth, memory, and computing capacities. They work in an unattended way in hostile conditions. An energy-efficient link-layer security framework is therefore necessitated. This paper proposes an energy-efficient link-layer security protocol called Encapsulate Sec. In addition to saving power consumption, Encapsulate Sec offers all security utilities. Encapsulate Sec provides node authentication, message integrity check, replay protection, and message semantic security at the reduced expense by minimizing overhead communication in the data packet and implementing only symmetric security algorithms. It also masks both the header fields and the MAC, making it more difficult to eavesdrop the wireless communication traffic. It also reduces the impact of message spoofing and replays attacks and MAC cryptoanalysis.

Keywords: WSNs; Encapsulate Sec; MAC; link-layer security protocol

1. Introduction

Data security and energy-conscious connectivity are core aspects of the design of the new security layer connexion architecture for wireless sensor networks. Sensor networks rely on wireless connectivity, which is by nature a transmitted medium and is more susceptible to security threats than its wired counterpart [1] due to a lack of a physical boundary. In the wireless domain, anyone with an acceptable transceiver will listen to, intercept, insert, and even modify the data transmitted. Protection facilities in wireless sensor networks are also desperately required to minimize the amount of damage and ensure efficient access control and confidentiality of information. However, WSNs are severally energy-constrained since many sensor networks are designed to operate unattended for a long time, and recharging or repairing batteries may be unsuccessful or impossible. Optimizing sensor nodes with minimal node capacities and the application-specific design of the networks, Security procedures have traditionally not been considered. This leaves WSNs under network security attacks, which could use even more battery power and reduce the lifetime of WSNs[2]. In the worst case, an opponent could be able to undetectably control certain sensor nodes, compromise the cryptographic keys, and reprogram the sensor nodes. Ensuring connectivity security in Wireless Sensor Networks (WSNs) is also critical; both because of the criticality of the resources in the sensor nodes and because of their pervasive and widespread distribution, with differing characteristics and degrees of security required [3].

Also, WSNs use data-centric multi-hop communication that, in essence, allows security support to be established at the connexion layer (increase the cost of security-related operations) rather than at the application layer, as in general networks. An energy-efficient link-layer security framework is therefore required.

There are several link-layer security architectures available that offer some combinations of security attributes desired by different WSN applications. Examples of WSN applications include tracking of environmental conditions (fluid, water level, temperature, stress, stress, strain, etc.), industrial automation in hostile settings, tracking of the activities of living beings in forests, sanctuaries, workplaces, classrooms, banks, etc., surveillance in war zones, enemy camps and a host of others [4][5]. Irrespective of the applications on which WSNs are implemented, the protection of the network nodes themselves and the data obtained and disseminated by them is of primary concern [6][7]. An energy-efficient link-layer security protocol (Encapsulate Sec) is proposed in this paper. Reach node authentication, message integrity checks, replay protection, block ciphers, and block cipher modes for semantic message security. The architecture proposed is aimed to offer the optimal level of security at the minimal overhead, thus saving the precious resources in the WSNs.

The rest of the paper is organized as follows. In Section 2, the literature review is presented. In Section 3, the motivation for Link Layer Security Protocols in Sensor Networks is produced. In Section 4, our proposed protocol is discussed in detail. Security analysis is explained in section 5. conclusions are discussed in Section 6.

2. Related Work

A various number of encryption protocols have been proposed for WSNs in the literature. The most widely recognized are SNEP[8], WSNSec[9], FlexiSec[10], TinySec[11], SenSec[12], Minisec[13], LLSP[14], and Zigbee[15]. Considering the limited packet size of WSNs (37 bytes on average, including headers and preamble). Since the TinyOS packet without encryption fields. Therefore; its frame format is considered the reference packet that is used to compute the security overhead due to the additional fields required to achieve the security services on each packet. Since the TinyOS packet has no security fields. Therefore, its frame format is considered to be the reference packet used to calculate the overhead security due to the additional fields required to provide security services for each packet. The figure1 shows the frame format of TinyOS. It is seen that the total header size for the TinyOS frame is just 13 bytes.

6-bytes	2-bytes	1-bytes	1-bytes	1-bytes	0-29-bytes	2-bytes
Preamble	Des	AM	Len	Grp	Payload	CRC

Figure 1. TinyOS frame

Figure 1 displays specifics of the additional packet size overhead in the headers related to the security services for these protocols in comparison to the TinyOS packet. Since MACs can detect transmission modification during the communication channel in transferred data, SNEP, WSNSec, FlexiSec, TinySec, SenSec, Minisec, LLSP, and Zigbee are used to detect transmission errors instead of CRC. Another security service is the assurance that the received data is fresh. The freshness counter is used to avoid intruders to repeat old messages. WSNSec, TinySec, Minisec, and Zigbee are used the freshness counter to detect such attacks. However, these security services add additional overhead contact with each packet. The other security services are given in Table 1.

Table 1. Header size comparison (Bytes)

Protocol	**Zigbee**	**TinySec-Auth**	**TinySec-AE**	**SNEP**	**MiniSec-U**	**MiniSec-B**	**LLSP**	**SenSec**	**FlexiSec HASH**	**FlexiSec AUTH_REP32**	**FlexiSec AUTH_ENC_REPP64**	**WSNSec**	**TinyOS**
Preamble	6	6	6	6	6	6	6	6	6	6	6	6	6
MAC	4	4	4	8	4	4	4	4	4	4	8	4	0
Src Add.	2	2	0	2	2	2	0	0	2	2	2	2	0
Des Add.	2	2	2	2	2	11 b	2	2	0	2	2	2	2
Length	1	1	1	1	5 b	5 b	1	1	1	1	1	1	1
Freshness Counter	4	2	0	0	3 b	1	0	0	0	0	0	2	0
CRC	2	0	0	0	0	0	0	0	0	0	0	0	2
Grp	0	0	0	0	0	0	0	1	0	0	0	0	1
AM	1	1	1	1	1	1	1	1	1	1	1	1	1
SCID	0	0	0	0	0	0	0	0	0	0	0	0	0
Random Number	0	0	0	0	0	0	0	3	0	0	0	0	0
Header Size	22	18	14	20	16	16	14	18	14	16	20	18	13

In our proposed protocol, we incorporate the advantages of these protocols by adding all security services, however; it's energy-efficient and time-efficient.

3. The Motivation for Link-Layer Security Protocols in WSN

In conventional networks, security services such as message confidentiality, integrity, and authenticity are usually achieved by an end-to-end security mechanism; intermediate devices like routers need only to access the headers of the messages and it is neither necessary nor desirable for them to have view the contents of messages. In wireless sensor networks, the scenario is different; the topology of the WSNs can vary from a simple star network to an advanced multi-hop wireless mesh network. The propagation technique between the hops of the network can be routing or flooding. The traffic mode (type) is usually many-to-one in sensor networks, sensor nodes sense and gather data from their environment and exchange information through wireless connections to a sink node sometimes referred to as a Base Station. However, the communication of the sensed data causes high power consumption, and as a result, the lifetime of the sensor nodes, and consequently the WSN lifespan is limited. Whereas, processing data locally within the sensor node consumes much less than its communication. Thus, it is possible to reduce the power consumption of the sensor nodes by reducing the number and the size of the communicated data and duplicate elimination. This reduction of communicated data extends the network lifetime and is possible if the sensor nodes‟ data are aggregated instead of being sent directly to the sink (the base station in WSNs). the contents of messages have to be accessed, modified, and suppressed by the intermediate nodes, it can't be used end-to-end security mechanisms between sensor nodes and the sink (the base station in WSNs) to achieve the security services: confidentiality, integrity, and authenticity of the exchanged messages. If message integrity isn't checked at each hop and kept to be checked only at the final destination node, an injected message may route many nodes in the network and consume power and bandwidth before detected at the final destination. Injected messages can be detected by link-layer security architecture once they reach to the first neighbor node in the network. Therefore, we aim to design a link-layer protocol to perform the security services; the confidentiality, integrity, and authenticity of messages between neighboring nodes with minimum packet size.

4. Proposed Protocol Description

In many protocols, the MAC field is sent completely in the packet which leads to an increase the packet size (power consumption) such as Zigbee protocol [15]. However the Encapsulate-Sec has a small frame size because it merges both MAC and header fields to form Masked Header field. This will result in hiding the header fields and MAC and it does not affect the basic security services. The Encapsulate-Sec protocol adds security-related fields, such as the source address, the freshness counter, and the MAC to the packet header and trailer. Like most of the related protocols. As shown in Figure 2a, the header of the Encapsulate-Sec frame consists of the full security fields, such as the Source address two bytes Src, two bytes Destination address Des, two bytes freshness counter, one-byte Active Message AM, and the Length Len fields. The trailer consists of an eight-byte MAC field.

6-bytes	2-bytes	2-bytes	1-bytes	1-bytes	2-bytes	0-29-bytes	8-bytes
Preamble	Des	Src	AM	Len	Ctr	Payload	MAC

(a) The Encapsulate-Sec complete frame before the encapsulation process

6-bytes	8-bytes	0-29-bytes
Preamble	Masked Header	Payload

(b) The Encapsulate-Sec complete frame after the encapsulation process

Figure 2. The Encapsulate-Sec frame format

The Encapsulate-Sec protocol can be described with Equations given below. Where Hi is the header of the packet, Ti is the trailer of the packet, Mi is the payload, MHi is the masked header, Ci is the freshness counter, KeyAUTH is the authentication key, and KeyENCR is the encryption key.

To start the Encapsulate-Sec protocol, node authentication-and encryption keys have to be exchanged in advance. Communicating parties are assumed to have already exchanged those keys using any key management protocol for WSNs. First, encrypt the message using the encryption (session) key = $E_{Key_{(Encr)}}$ (Mi)

We concatenate the destination address (Des), source address (Scr), Active Message (AM), Length field (Len), and the freshness counter to perform the header (Hi).

$$Hi = Des : Scr : AM : Len : Ci \quad (1)$$

The MAC is computed by running a hashing function. The hash function has two inputs the authentication secret key $\mathbf{Key_{AUTH}}$, and the message **Mi**. Here, we apply the hash function with the secret authentication key on the encrypted message $\mathbf{E_{Key_{(Encr)}}}$ **(Mi)** to perform the trailer of the packet (MAC).

$$Ti = MAC_{Key_{(AUTH)}} \left[E_{Key_{(Encr)}} \; (Mi) \right] \quad (2)$$

On the sender side, the computed MAC is XORed with the header of the message $\mathbf{H_i}$ instead of appending to the message as a separate field as the nature of all other link layer protocols. An 8-byte Masked Header **MHi** is produced for the current message **Mi**.

$$\boldsymbol{MHi \;=\; Ti \;\oplus\; Hi} \qquad (3)$$

Node A sends the masked header concatenated with the encrypted message to node B.

$$\boldsymbol{A \;\rightarrow\; B \qquad MHi :} \quad \boldsymbol{E_{Key_{(Encr)}}\;(Mi) \qquad (Packet\;\; format)}$$

Figure 1b shows the frame format of the Encapsulate-Sec protocol. It consists of the preamble, Masked Header, and Payload. Figure 3 shows the frame format generation of the Encapsulate-Sec protocol at the sender side.

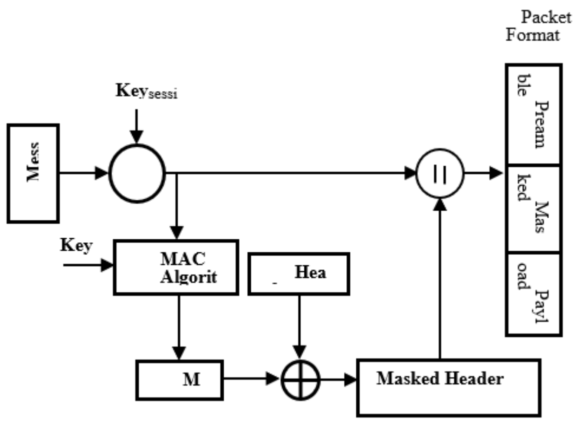

Figure 3. The frame format generation of the Encapsulate-Sec protocol

At the receiver side, once a packet is received at the destination, the MAC is generated and verified. If a match is achieved, the sender's address is looked up in the cache. If a fresh count is found, the packet is accepted, decrypts the payload using the session key, and stored, and then the freshness counter value has been updated. If the sender address is not found in the cache the authentication algorithm must be run to decide whether the packet is authentic. If so, the packet is accepted, a new entry is added for it in the cache, and the authentication is set for the sender. Otherwise, the packet is dropped. Figure 4 shows the Encapsulate-Sec protocol on both (transmitter/receiver) sides.

There are three main reasons for the packet "early dropped"; first, the received packet may be directed to another node. Second, it may have a bit of error due to transmission. Finally, it may be a false packet released by a malicious node. In Encapsulate-Sec protocol, if the receiver node receives a lost packet directed to another node, it first computes the hash and checks the destination address field, then it compares the destination address by its hold address. If they are not equal, it stops the receiving process and drops the packet immediately without complete receiving the rest of the packet.

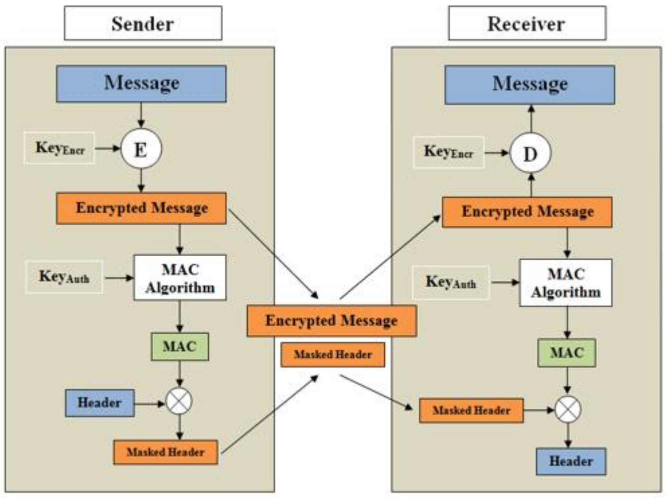

Figure 4. The Encapsulate-Sec protocol on both (transmitter/receiver) sides

Masking the header of the message with the MAC makes it confidential; as a result, it will be more difficult to trace the flow of wireless communication. To get the header information, an adversary needs to get the authentication key K_Auth to re-compute the MAC and then XOR it with the masked header MHi to resolve the original header (Hi). Figure 5 shows a flow chart of the behavior of the Encapsulate-sec protocol on the receiver side.

5. Security Analysis

Confidentiality is achieved by encrypting the data with a secret key that is only known by the intended receivers. Semantic security ensures that the adversary cannot learn anything about the plaintext from the ciphertext. This is usually achieved by using block cipher modes of operation that support semantic security [16][17]. The E-GOST [18] encryption algorithm is chosen for Encapsulate-Sec. Enhanced Version of GOST Algorithm (E-GOST) algorithm is proposed to be easily implemented in both software and hardware E-GOST has been designed to be a general-purpose algorithm and to handle most of the limitations and constraints associated with other algorithms such as key size, speed, complexity, software and hardware implementations. E-GOST has been designed to mitigate some of the previous work limitations and constraints such as key size, block size, speed, lookup tables, S-boxes, P-boxes, complexity, software, and hardware implementations. The main purpose of designing a new encryption algorithm is to be efficiently utilized in wireless networks. Also, E-GOST has rigid features against different cryptanalysis techniques in addition to its simplicity which is the used s-boxes that can be implemented in a bit-slice manner. This feature is very important to the encryption in wireless networks, especially Wireless Sensor Networks (WSNs), to avoid the side-channel attack. Encapsulate-Sec protocol provides semantic security by running the E-GOST algorithm using Cipher block chaining (CBC) mode.

As Figure 4 shows, the encryption of the next block will require an XOR operation of the precomputed output with the message without running the E-GOST algorithm.

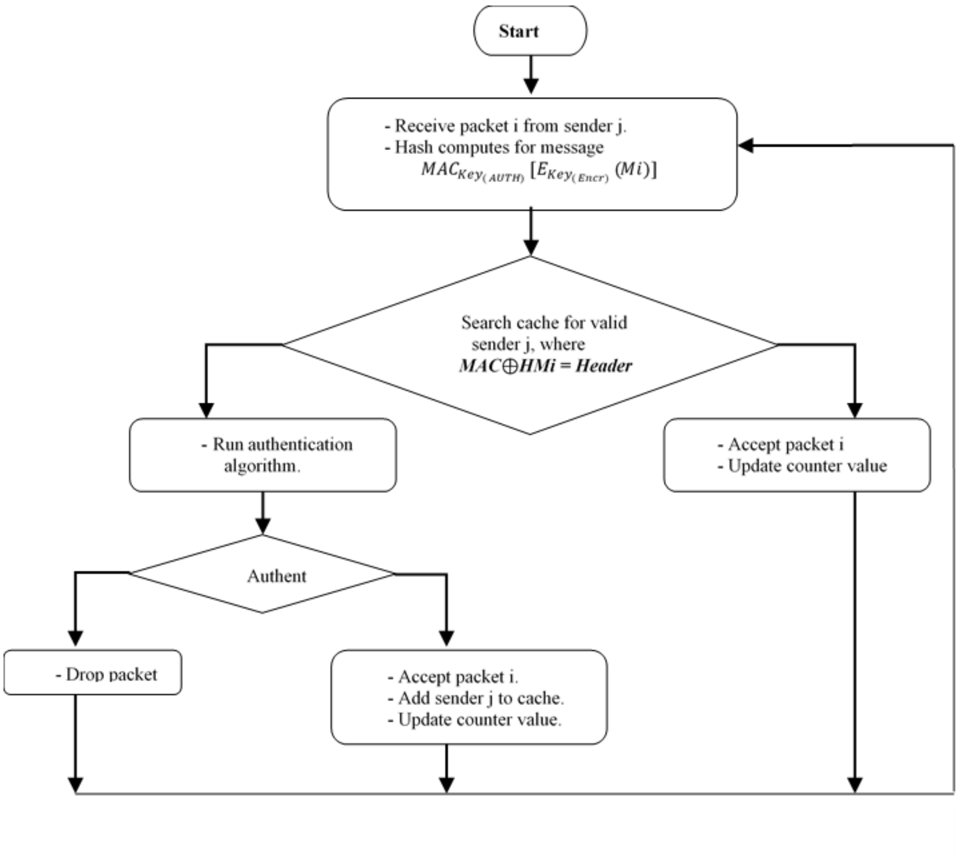
Start
- Receive packet i from sender j.
- Hash computes for message $MAC_{Key_{(AUTH)}}[E_{Key_{(Encr)}}(Mi)]$
Search cache for valid sender j, where $MAC \oplus HMi = Header$
- Run authentication algorithm.
- Accept packet i
- Update counter value
Authent
- Drop packet
- Accept packet i.
- Add sender j to cache.
- Update counter value.

Figure 5. The behavior of the Encapsulate-Sec protocol on the receiver side

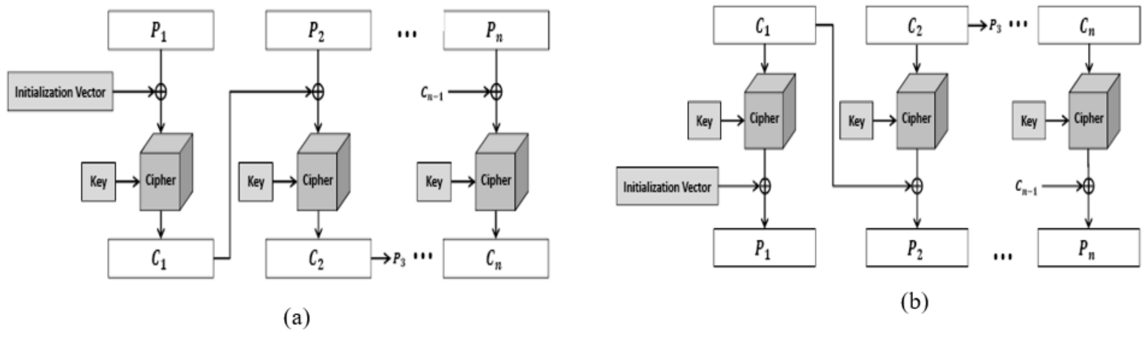

Figure 6. (a) Encryption with CBC mode of operation, (b) Decryption with CBC mode of operation

The same procedure applies to the decryption process. The initial vector used to initialize the CBC mode is defined as:

$$\boldsymbol{IV} = \boldsymbol{MAC(K_{Encr})}: \boldsymbol{MAC(K_{AUTH})}$$

5.1. Data Authentication and Integrity

Data integrity helps the receiver to ensure that the received data is not altered by an adversary during transmission. Data authentication allows verifying that the data is sent by the claimed sender. In Encapsulate-Sec protocol, these properties are achieved by computing a message authentication code using the secret authentication key KAUTH that is only known by the sender and the receiver. When the receiver verifies the correctness of the MAC of the received message it ensures that the message was sent by the claimed sender who has the authentication key KAUTH, and it was not altered during transmission. As shown in Table 4.2, most of the protocols use an encryption algorithm with cipher block chaining CBC-MAC to produce the MAC. This method does not require implementing a separate MAC algorithm.

5.2 Data Freshness and Replay Protection

Data freshness is used to prevent the adversary from playing old messages. This is achieved by ensuring that the sent data is recent through the maintenance of a freshness counter on both sides of the communication. The counter-based algorithm for replay protection is shown below.

Algorithm 1

```
Boolean FreshnessCheck(CounterReceived,NodeID)
{
id = o;
for id = 1 to lastValidId
{
  if (id == NodeID)
  {
    if (CounterReceived <= LastCount[NodeID])
    return false;
    else
    {
    LastCount[NodeID]=CounterReceived;
    return true;
    }
  }
 }
}
```

6. Conclusion

An energy-efficient link layer protocol called Encapsulate-Sec is produced as a compact rigid protocol for WSN. Encapsulate-Sec provides the same security services that Zigbee provides and achieving the well-balanced trade-off between performance and security. Encapsulate-Sec smartly minimizes the packet size besides hiding the packet header and trailer which provide immunity against eavesdropper to flow the wireless communication. Our proposed protocol is evaluated and compared with other related protocols in terms of a lifetime which enhances the network lifetime, end-to-end delay, throughput ratio, and energy consumption compared to Zigbee.

Author Contributions: The authors contributed equally to this research.

Funding: The authors receive no fund for this research.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Akyildiz, I. F., & Vuran, M. C. (2010). Wireless sensor networks (Vol. 4). John Wiley & Sons..
2. Niewiadomska-Szynkiewicz, E., Kwaśniewski, P., & Windyga, I. (2009). Comparative study of wireless sensor networks energy-efficient topologies and power save protocols. Journal of Telecommunications and information technology, 68-75.

3. Tiwari, A., Ballal, P., & Lewis, F. L. (2007). Energy-efficient wireless sensor network design and implementation for condition-based maintenance. ACM Transactions on Sensor Networks (TOSN), 3(1), 1-es.
4. Sharma, K., Ghose, M. K., Kumar, D., Singh, R. P. K., & Pandey, V. K. (2010). A comparative study of various security approaches used in wireless sensor networks. International journal of advanced science and technology, 17(2), 31-44.
5. Ullah, F., Ahmad, M., Habib, M., & Muhammad, J. (2011, March). Analysis of security protocols for Wireless Sensor Networks. In 2011 3rd International Conference on Computer Research and Development (Vol. 2, pp. 383-387). IEEE.
6. Perrig, A., Szewczyk, R., Tygar, J. D., Wen, V., & Culler, D. E. (2002). SPINS: Security protocols for sensor networks. Wireless networks, 8(5), 521-534.
7. Lighfoot, L. E., Ren, J., & Li, T. (2007, May). An energy efficient link-layer security protocol for wireless sensor networks. In 2007 IEEE International Conference on Electro/Information Technology (pp. 233-238). IEEE.
8. Zhu, S., Setia, S., & Jajodia, S. (2006). LEAP+ Efficient security mechanisms for large-scale distributed sensor networks. ACM Transactions on Sensor Networks (TOSN), 2(4), 500-528.
9. Shukur, M. I., Chyan, L. S., & Yap, V. V. (2009). Wireless sensor networks: Delay guarentee and energy efficient MAC protocols. World Academy of Science, Engineering and Technology, 50(2009), 1062-1066..
10. Baronti, P., Pillai, P., Chook, V. W., Chessa, S., Gotta, A., & Hu, Y. F. (2007). Wireless sensor networks: A survey on the state of the art and the 802.15. 4 and ZigBee standards. Computer communications, 30(7), 1655-1695.
11. Karlof, C., Sastry, N., & Wagner, D. (2004, November). TinySec: a link layer security architecture for wireless sensor networks. In Proceedings of the 2nd international conference on Embedded networked sensor systems (pp. 162-175).
12. Kalita, H. K., & Kar, A. (2009). Wireless sensor network security analysis. International Journal of Next-Generation Networks (IJNGN), 1(1), 1-10.
13. Singh, S. K., Singh, M. P., & Singh, D. K. (2010). A survey of energy-efficient hierarchical cluster-based routing in wireless sensor networks. International Journal of Advanced Networking and Application (IJANA), 2(02), 570-580.
14. Singh, S. K., Singh, M. P., & Singh, D. K. (2010). Energy-efficient homogeneous clustering algorithm for wireless sensor network. International Journal of Wireless & Mobile Networks (IJWMN), 2(3), 49-61.
15. Alliance, Z. (2005). ZigBee Specification v1. 0. New York, USA, 320.
16. Sastry, N., & Wagner, D. (2004, October). Security considerations for IEEE 802.15. 4 networks. In Proceedings of the 3rd ACM workshop on Wireless security (pp. 32-42).
17. Struik, R., & Rason, G. (2002). Security and security architectural recommendations for the IEEE 802.15. 4 Low-Rate WPAN. Certicom Corp.
18. Aboshosha, B. W., Dessouky, M. M., Ramadan, R. A., El-Sayed, A., & Galal, F. H. (2019). Enhanced Version of GOST Cryptosystem for Lightweight Applications. no. July.