Energy Efficient Encryption using Counter mode of operation in Wireless Sensor Network

A. Shish Ahmad, B. Dr. Mohd. Rizwan Beg
CSE Department, Integral University, Lucknow, India

Abstract- Sensor networks are deployed for various monitoring applications. The reporting should be in secure manner so that the adversary can’t forge the data and can be prevented from any disclosure. In this paper we have identified those calculations which are not necessary to be performed on sensor node for securing the network, shifted to base stations to save the energy at nodes. Using pre-processing approach saves energy consumption at nodes for producing Secured/Encrypted data. So here we are proposing EEECMO method Energy Efficient Encryption using Counter mode of operation in such a way that before the deployment, the base station process the part of the security algorithm that involve the key in the setup phase. In this fashion, we are getting secure communication network without the distribution of the key among the sensor nodes and also minimizes energy consumption at sensor node.

Key Words- Sensor network, Security, confidentiality, Block Cipher, Encryption and Decryption, Counter mode, pre-processing, Stream Cipher.

1. Introduction

Wireless sensor networks have become a promising future to many applications, such as smart houses, smart farms, smart parking, smart hospitals, habitat monitoring, building and structure monitoring, distributed robotics, industrial and manufacturing, and national security. In addition to common network threats, sensor networks are more vulnerable to security breaches because they are physically accessible by adversaries. Imagine the damage caused by compromised sensor network in sensitive military and hospital applications. These are often deployed in unattended environments, thus leaving these networks vulnerable to passive and active attacks by the adversary. The conversation between sensor nodes can be eavesdropped by the adversary. The adversary can be aware of the conversation between the sensors and can forge the data. Sensor nodes should be resilient to these attacks. Since sensor nodes are resource constrained and run on battery, energy consumption should be low to make it operate for many days.

Confidentiality is very much concern in battlefield scenarios, where fraudulent node impersonate as a legitimate sensor.

Security in ad hoc networks is an essential component for basic network functions like packet forwarding and routing and network operation can be easily jeopardized if countermeasures are not embedded into the basic network functions at the early stages of their design. The disclosure threat involves the leakage of Security information from the system to a party that should not have seen the information and is a threat against the confidentiality of the information.

Data confidentiality is a core security primitive for ad hoc networks. It ensures that the message cannot be understood by anyone other than the authorized personnel. With wireless communication, anyone can sniff the messages going through the air, and without proper encryption all the information is easily available. Due to the small battery backup of Sensor Node, security algorithm must consume low power. Symmetric key are efficient for providing security services like confidentiality.

Security Solutions should minimize the amount of computation and communication required to ensure the security services to accommodate the limited energy and computational resources of mobile, ad hoc-enabled devices.

Generally DES algorithm is a basic building block for providing data security. To apply DES in a variety of applications, four "modes of operations have been defined. These four modes are intended to cover virtually all the possible applications of encryptions of encryption for which DES could be used.

Here we are proposing Energy Efficient Encryption using Counter mode of operation (EEECMO) using DES algorithm for providing confidentiality in such a way that total energy consumption decreases at Sensor node. Using the property of property of pre-processing in CTR, we pre-process those steps which is not necessary to be performed at sensor node to base station.

The Proposed method uses the key at the base station only to produce the credential that play role to encrypt the sensed data at Sensor node/sensor aggregator, so there is no need to distribute the key in sensor network. It saves a lot of problem arises to distribute the key as well as saves energy to distribute the key.

2. Counter MODE (CTR)
Although interest in the counter (CTR) mode has increased recently with applications to ATM (asynchronous transfer mode) network security and IP sec (IP security). A counter equal to the plaintext block size is used. The only requirement stated in SP 800-38A is that the counter value must be different for each plaintext block that is encrypted. Typically, the counter is initialized to some value and then incremented by 1 for each subsequent block (modulo 2^b, where b is the block size). For encryption, the counter is encrypted and then XORed with the plaintext block to produce the cipher text block; there is no chaining. For decryption, the same sequence of counter values is used, with each encrypted counter XORed with a ciphertext block to recover the corresponding plaintext block. Thus, the initial counter value must be made available for decryption. Given a sequence of counters T1, T2, A, ..., TN, we can define CTR mode as follows:

- Encryption
  \[ C_j = P_j \oplus E(K, T_j) \quad j = 1, 2, \ldots, N-1. \]
  \[ C_N = P_N \oplus \text{MSB}_b \cdot [E(K, T_N)] \]

- Decryption
  \[ P_j = C_j \oplus E(K, T_j) \quad j = 1, 2, \ldots, N-1. \]
  \[ P_N = C_N \oplus \text{MSB}_b \cdot [E(K, T_N)] \]

For the last plaintext block, which may be a partial block of bits, the most significant bits of the last output block are used for the XOR operation; the remaining b-u bits are discarded. Unlike the ECB, CBC, and CFB modes, we do not need to use padding because of the structure of the CTR mode.

As with the OFB mode, the initial counter value must be a nonce; that is T1, must be different for all of the messages encrypted using the same key. Further, all Ti values across all messages must be unique. If, contrary to this requirement, a counter value is used multiple times, then the confidentiality of all of the plaintext blocks corresponding to that counter value may be compromised. In particular, if any plaintext block that is encrypted using a given counter value is known, then the output of the encryption function can be determined easily from the associated ciphertext block. This output allows any other plaintext blocks that are encrypted using the same counter value to be easily recovered from their associated ciphertext blocks.

One way to ensure the uniqueness of counter values is to continue to increment the counter value by 1 across messages. That is, the first counter value of the each message is one more than the last counter value of the preceding message. Following are advantages of CTR mode.

A. **Hardware efficiency:** Unlike the three chaining modes, encryption (or decryption) in CTR mode can be done in parallel on multiple blocks of plaintext or cipher text. For the chaining modes, the algorithm must complete the computation on one block before beginning on the next block. This limits the maximum throughput of the algorithm to the reciprocal of the time for one execution of block encryption or decryption. In CTR mode, the throughput is only limited by the amount of parallelism that is achieved.

B. **Software efficiency:** Similarly, because of the opportunities for parallel execution in CTR mode, processors that support parallel features, such as aggressive pipelining, multiple instruction dispatch per clock cycle, a large number of registers, and SIMD instructions, can be effectively utilized.

C. **Pre-processing:** The execution of the underlying encryption algorithm does not depend on input of the plaintext or cipher text. Therefore, if sufficient memory is available and security is maintained, preprocessing can be used to prepare the output of the encryption boxes that feed into the XOR functions. When the plaintext or ciphertext input is presented, then the only computation is a series of XORs. Such a strategy greatly enhances throughput.

D. **Random access:** The ith block of plaintext or ciphertext can be processed in random-access fashion. With the chaining modes, block cannot be computed until the i-1 prior block are computed. There may be applications in which a ciphertext is stored and it is desired to decrypt just one block; for such applications, the random access feature is attractive.

E. **Provable security:** It can be shown that CTR is at least as secure as the other modes.

F. **Simplicity:** Unlike ECB and CBC modes, CTR mode requires only the implementation of the encryption algorithm and not the decryption algorithm. This matters most when the decryption algorithm differs substantially from the encryption algorithm, as it does for AES. In addition, the decryption key scheduling need not be implemented.

![Figure 1: Operation of CTR](image)

**3. PROPOSED METHOD**

Our proposed method is Energy Efficient Encryption using Counter mode of operation.
(EEECMO) for securing Sensor network is divided into 3-Phases.

3.1 Phase 1:-

This is the setup phase performed at base station before the deployment of the sensor node. We choose the incremental Counter of 64-bit and keys k1,k2......Kn, Where n is the number of sensor nodes. Apply the DES encryption as follows, as shift register and select s-bits depend on the size of sensed data to be encrypted.

Here we are using different keys to produce the credential at different respective nodes, that play role to produce the cipher text at nodes after deployment.

The shift register value (after selecting s-bits) store at each node and deploy it randomly.

3.2 Phase 2:-

This starts after the deployment of the sensor nodes. Using End to End encryption the sensor encrypt the sensed data with the help of the Credential loaded previously in setup phase as follows.

In the above figure each sensor node/ aggregator node perform the X-OR operation bit by bit to the calculated credential prior to the deployment. Here we can see that for encrypting the data at sensor node no key is required. Following shows how the cipher text calculated.

\[ C_j = P_j \oplus \text{MSB}_s(E_{K_j}(\text{Counter (address)}_j)) \]

where \( j = 1 \ldots N \)

After calculating the cipher text each node send that encrypted data to base station. After receiving the encrypted data from each node/ aggregator the base station perform the third phase of EEECMO.

3.3 Phase 3.- After receiving the encrypted data from the network, the base station starts the decryption as follows. The Decrypted data is found as follows

\[ P_j = C_j \oplus \text{MSB}_s(E_{K_j}(\text{Counter (address)}_j)) \text{ where } j = 1 \ldots N \]

The plaintext is calculated by X-ORing the coming Cipher text with the shifted s-bits found by apply DES encryption with the same key on incremental counter.
4. RESULT AND ANALYSIS

Keys Plays its role in the first phase i.e. is the setup phase in which we apply the DES encryption using different keys at base station only. After encryption we are choosing only s-bits from 64-bit credential. The s-bit block size depends on the size of data send by the sensor aggregator/sensor node to the base station after deployment. The key does not participate in second phase of encryption process, so there is no need of key establishment in network.

Secondly because the encryption in first phase does not depend on the message, so we can pre-process that part on base station only.

Another advantage of using only s-bits is to perform stream encryption (may be a character of 8-bit) instead of block cipher technique. So instead of using block cipher technique like DES, we can choose any stream cipher technique like RC4. Following are the advantages of using stream cipher over block cipher.

1. Stream ciphers are often used in applications where plaintext comes in quantities of unknowable length where as in Block ciphers the length of the plaintext is known.
2. In Stream Ciphers the encryption and decryption process is done bit by bit where as in Block Ciphers the encryption and decryption process is done on the block of data.
3. Stream Ciphers are more faster than Block Ciphers because in Stream Ciphers the encryption/decryption is performed bit by bit while in Block Ciphers the encryption/decryption is performed on the block of data.
4. Block ciphers must be used in ciphertext stealing or residual block termination mode to avoid padding, while stream ciphers eliminate this issue by naturally operating on the smallest unit that can be transmitted (usually bytes).
5. Stream ciphers are often used in applications where plaintext comes in quantities of unknowable length—for example, a secure wireless connection.
6. If a block cipher were to be used in this type of application, the designer would need to choose either transmission efficiency or implementation complexity, since block ciphers cannot directly work on blocks shorter than their block size.
7. In cryptography, a stream cipher is a symmetric key cipher where plaintext bits are combined with a pseudorandom cipher bit stream (key stream), typically by an exclusive-or (XOR) operation. In cryptography, a block cipher is a symmetric key cipher which operates on fixed-length groups of bits, termed blocks, with an unvarying transformation.

So instead of apply DES for encryption, we can go for stream cipher algorithm like RC4.

So following are advantages of using EEECMO.

1. The Throughput, hardware & software efficiency of overall network system will increase due to the pre-processing of security algorithm.
2. Energy consumption at sensor node decreases, because of pre-processing of the part of the security algorithm carried out at base station. So it also saves the time and energy for key establishment process as shown in figure 6.

3. Counter selected at base station incremented for the next sensor hop also act as a authenticator for that sensor node.

4. Stream cipher is faster than block cipher. Also the computation time it takes to generate a cipher text is less than the block cipher for the same length of the output cipher text as shown in following figure 5.

![Figure 5: Encryption time of stream & Block cipher](image)

![Figure 6: Energy saved in µJ with varied number of nodes](image)

Figure 5 shows that the computation time for stream cipher is less than the block cipher, so selecting s-bits in shift register motivate us to use stream cipher encryption instead of block cipher. Figure 6 shows that the pre-processing approach used in counter mode of operation saves energy at sensor nodes. Figure indicates that as the network energy will be saved more by network if we increase number of nodes.

5. Conclusion
Using counter mode increases the hardware and software efficiency. Also due to the pre-processing we can save the energy by saving the computation cost of security algorithm at node. The Throughput of overall network system increases due to the pre-processing of security algorithm. As the key does not participate to produce the cipher text at sensor node, because the requirement of the key does not depend on the message, so here we are saving the effort, time and computation cost for distributing the key. Also using counter mode we can go for the stream cipher approach instead of block cipher which also minimizes the computation cost. So due to limited computational capabilities and limited battery power we can go for those approaches where part of the security algorithm can be computed at base instead of sensor node to minimize the energy consumption. Also if the part of security algorithm does not depended on key at sensor node i.e. not depended on message, no distribution of key is required in network.

Stream cipher algorithms can more preferred because of compatibility to the sensors.

6. REFERENCES


