RESEARCHERID
THOMSON REUTERS

[*Punde, 9*(1): January-March 2019] **ISSN 2277 – 5528**
Impact Factor- 5.085

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & MANAGEMENT

AN SECURED MESSAGE SENDING USING TINY ENCRYPTION ALGORITHM BASED ON WIRELESS SENSOR NETWORK

Anurag Punde[*1] & Mr. Rajesh K. Chakrawarti[2]

[*1]PG-Scholar, Department of Computer Science & Engineering Shri Vaishnav Institute of Technology & Science, Indore, INDIA

[2]Assistant Professor Department of Computer Science & Engineering Shri Vaishnav Institute of Technology & Science, Indore, INDIA

ABSTRACT

A wireless sensor network is an effective tool that contains the large number of sensor nodes. We can congregate the data in assorted conditions, with the help of Wireless Sensor Network. The Tiny Encryption Algorithm (TEA) is a cryptographic algorithm that minimizes the footprint and improves the system efficiency. In this work, we have given the implementation of the algorithm that not only increases the performance of the system but also increases the effectiveness and the efficiency of the computer. We have developed the android based application that securely send and receive the messages using Wireless Sensor Network in our research work.

Keywords: *Wireless Sensor Network (WSN), Tiny Encryption Algorithm (TEA), Encryption, Autonomous Sensor, Decryption.*

I. INTRODUCTION

Wireless sensor network (WSN) is highly distributed and self-configuring network that has small, light weighted nodes. We can use Wireless Sensor network in many application such as to constant monitoring and detection of events, in military, battlefield, surveillance, forest fire, flood detection, patient monitoring and etc.[1] Ad HOC wireless network also uses the wireless nodes but it is different from the wireless sensor network. There are certain limitations in the Ad HOC network such as resource limitation (Memory, Power and Processing), Now having unique global IDs, more prone to failure and etc. Currently, wireless sensor networks are beginning to be deployed at a hastened pace. It is not perverse to expect that in the next 10-15 years the world may be concealed with wireless sensor networks with the facility to access them using the Internet. This can be considered as the use of Internet becoming a physical network. A wireless sensor network is a collection of nodes organized into a cooperative network [2]. Each node comprises processing capability (one or more microcontrollers, CPUs or DSP chips), might include various types of memory (program, data and flash memories), have a RF transceiver (usually with a single omni directional antenna), have a energy source foundation (e.g., solar cells and batteries), and contain a range of sensors and actuators. The nodes commune wirelessly and regularly self-organize after being deployed in an ad hoc fashion. Systems of 1000s or even 10,000 nodes are estimated. Such systems can modernize the way we live and vocation. The Tiny Encryption Algorithm (TEA) is basically a symmetric (private) key encryption algorithm created by David Wheeler and Roger Needham of Cambridge University. They published their work in 1994. This particular Algorithm was designed for basically easiness in understanding and implementation while considering performance, while looking for encryption strength on equivalence with more sophistically complicated and resource-intensive algorithms such as DES (Data Encryption Standard). Wheeler and Needham summarize their whole work as follows: "it is anticipated that this Algorithm can easily be translated into most of the modern languages. It takes little set up time and does enough rounds to make it secure. It can replace Data Encryption Standard (DES) and is tiny enough to write into almost any programming language"[3]. Academic research, one can learn of TEA's relative merits.

II. PROBLEM STATEMENT

Wireless sensor nodes have strict power constraints, providing protection to wireless sensor networks is harder than conventional network security applications. For example like asymmetric cryptographic algorithms are not

applicable in wireless sensor networks due to the limited and restricted computation, power, and storage resources available on sensor nodes.

There is very less work done as we can see that in today's scenario that wireless sensor network has lesser security that the other conventional network system. In that phase it is facing plenty of security related problems. Due to the less overheads the data might not be secure. There are so many of intrusion chances for data stealing.

As we can see that the wireless network is an open communication network where two devices can communicate each other without having any particular security or governance body to the base. The data must be shared between two parties must be encrypted so that the intruder can't access the important information between the two of the working devices.

Distinctive characteristics of a WSN comprise:-

- Limited power that can be harvest or store
- Ability to hold up insensitive environmental conditions
- Ability to cope with node failures
- Mobility of nodes
- Dynamic network topology
- Communication failures
- Heterogeneity of nodes
- Large scale of deployment
- Unattended operation

III. PROPOSED ALGORITHM

In this section we are proposing the algorithm for encryption and decryption. As we send the data, we always expect that the data will be received at receiver end will be secure.

A. Encryption Routine

Figure 1 shows the structure of the TEA encoding routine. The inputs to this encoding routine are an original data block called Plaintext and a key K .The plaintext P = (Left [0], Right [0]) and the encrypted or cipher text C = (Left [64], Right [64]). The plaintext block is divided into two equal parts, Left [0] and Right [0]. Each half is used to encode the other half over 64 rounds of processing and then combined produce the cipher text block.

- Each round i has inputs Left[i-1] and Right[i-1], derived from the earlier round, as well as a sub key K[i] derived from the 128 bit overall K.
- The sub keys K[i] are different from K and from each other.
- The constant delta = $(\sqrt{5} - 1) * 2^{31}$ = 9E3779B9, is derived from the golden number ratio to ensure that the sub keys are distinct and its precise value has no cryptographic significance.
- The round function differs slightly from a classical Fiestel cipher structure in that integer addition modulo 2^{32} is used instead of exclusive-or as the combining operator.

Algorithm:

```
void code (long* v, long* k)
{
unsigned long y = v[0], z = v[1], sum = 0, /* set up */
delta = 0x9e3779b9, n = 32 ; /* a key schedule constant */
while (n-->0)
{
 /* basic cycle start */
sum += delta ;
```

```
y += (z<<4)+k[0] ^ z+sum ^ (z>>5)+k[1] ;
z += (y<<4)+k[2] ^ y+sum ^ (y>>5)+k[3] ; /* end cycle */
}
v[0] = y ; v[1] = z ;
```

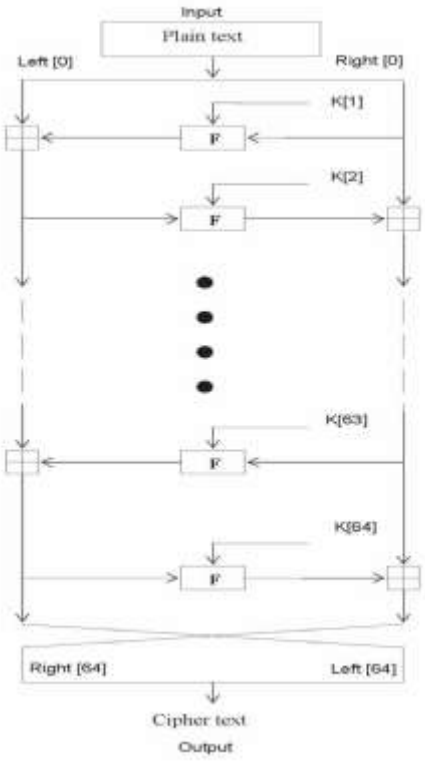

Figure 1: Abstract structure of TEA encryption routine

B. Decryption Routine

Decryption is essentially the same as the encryption process; in the decode routine the encrypted text is used as input to the Decode routine, but the sub keys K[i] are used in the reverse order. Figure 2 presents the assembly of the decryption routine. The in-between value of the decoding process is equal to the corresponding value of the encoding process with the two equal parts of the value swapped. For example, if the output of the n^{th} encryption round is ELeft[i] || ERight[i] (ELeft[i] concatenated with ERight[i]). Then the corresponding input to the $(64\text{-}i)^{th}$ decryption round is DRight[i] || DLeft[i] (DRight[i] concatenated with DLeft[i]). After the last iteration of the encryption process, the two equal parts of the output are exchanged, so that the cipher text is ERight [64] || ELeft [64], the resultant outcome of that round is the final encrypted or cipher text C. Now this cipher text is used as the input to the decryption routine. ERight [64] || ELeft [64] is the input for round 1, which is equal to the 2- bit swap of the output of the 64^{th} round.

ISSN 2277 – 5528
Impact Factor- 5.085

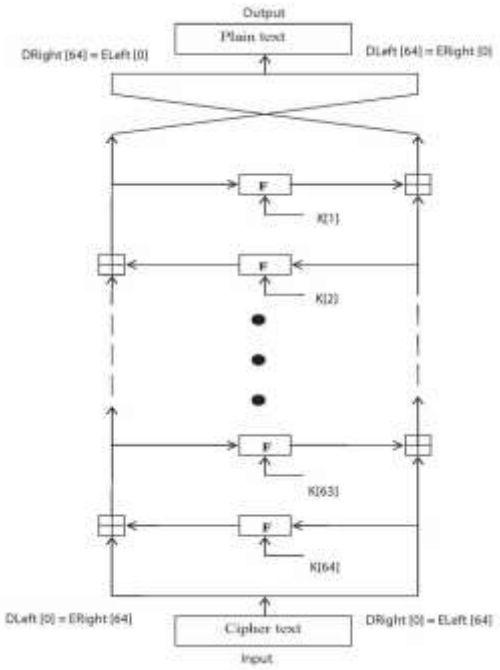

Figure 2: The abstract structure of TEA decryption routine

IV. COMPARISON

As today, each and every organization is working on the internet. Everything now we share with the help of internet. The two major concerns regarding the sharing of data and information is to maintain the data and information confidential and secure from unauthorized users to access it. As the eavesdropping is done usually while transmitting the confidential information by an attacker, from very beginning various algorithms concerning this issues were designed which are our main focus of comparing our algorithm with those in which we found that the key sharing between the two parties unlikely essential. In this comparative study, we are concerning about the algorithms that provide the confidentiality and security to our data. So that, the unauthorized users cannot easily fetches the information. In this manner we can prevent our data. A formula used to turn ordinary data, or "plaintext," into a secret code known as "cipher text." Every one of these algorithm uses a sequence of bits or string known as a "key" to perform the calculations. The larger the key (the more bits), the bigger the number of potential patterns can be produced, thus making it harder to break the code and descramble the contents. Most of the encryption algorithms uses the block cipher technique, which codes fixed blocks of input that are typically varies from 64 to 128 bits in size. Several Algorithms use the stream technique, which works with the continuous stream of input. Many of the algorithms were suggested by many researchers such as 3DES algorithm, Blowfish, IDEA, DES, summer and many more. All of these algorithms works on some dataset. Despite of it, they have their own way to generate the key though which they convert the plaintext or the original text in to some different form called cipher text. The process of converting the original text to some unreadable text is known as encryption. At the receiver ends, when receiver receives the unreadable message from source than it required same key to decode it. It means the same key will be used to decrypt the unreadable message to the original text. This type of process is known as decryption. Every algorithm will generate the key and it is definitely uses the different concept to not only generate the key but also they examine how to share the key.

Table 1: Comparison among various Encryption algorithm

TEA	3DES	Blowfish	IDEA	DES	Misty 1	Square	Summer
Tiny Encryption Algorithm is a faster and much secure algorithm developed by David Wheeler and Roger Needham of Cambridge Computer Laboratory. There is one known issue with the key schedule, so it is not recommended if utmost security is required. This algorithm is available in 16 and 32 round versions. More the rounds (iterations), more the security, but slower.	This is far better than DES. 3DES practices three applications of the DES in EDE (Encipher-Decipher-Encipher) mode with totally independent keys. This algorithm is assumed very secure (majority of the banks are using it to safeguard their prized transactions), but it is also very, very slow.	Blowfish is a high security cipher algorithm produced by Bruce Schneier, the writer of Applied Cryptography and founder of the company Counterpane. It is fast, is measured as secure and is resilient to linear and differential analysis. This is usually cipher of choice.	International Data Encryption Algorithm was produced by Xuejia Lai and James Massey. It is legitimately fast, is considered somewhat secure, and it is also resistant to both linear and differential analysis. To use this other than personal, a royalty must be paid to Ascom-Systec Ltd.	Data Encryption Standard was developed in the early 1970s by IBM with input from NSA, But a only key can be broken in three days by the EFF (Electronic Frontier Foundation), an ill funded organization. This algorithm was provided for completeness.	Misty1 was produced by M. Matsui of Mitsubishi. It is a reasonably fast encryption algorithm that is resistant to both linear and differential analysis. It is actually new though, so must be used with caution.	Square is a very fast and reasonably secure block encryption designed by John Daemen and Vincent Rijmen It hasn't been issued to as much peer review as Blowfish, 3DES, IDEA, etc., so it may be susceptible to attacks.	This is a proprietary stream cipher constructed by the author and it is planned for speed only. It is provided with backward compatibility with Version 1 of ScramDisk and is not recommended for practice on freshly created disks. As an alternative, use Tiny Encryption Algorithm or Blowfish, which are both reasonably fast.

V. RESULT & DISCUSSION

It is being observed that at firstly the sensor nodes weren't able to achieve the required security as the mechanism for the same was highly costly as per system resources. After analysis it is found that TEA algorithm is satisfying the required parameters by following which it can be applied to the sensor nodes without slightest difficulty. The cipher text is strong enough and requires less computational power so it will not be burden on the sensor nodes having low power issue. The Android system was taken due to their high availability as well as it will be cheaper to maintain and illustrate the implementation as we might needed to modify the code during development. Taking Java is adding another advantage like security which is highly recommended. The most important factor is the tiny encryption algorithm plays a crucial role as we can see that it is providing good results in terms of security. As this whole process is running on the server side there will be no computational power consumption issue on the client side sensor nodes.

As per the result extracted from the implementation here we can see that right now only 16-18 digits of data can be decrypted. As it may be required that sometimes large amount of data or longer data must be sent from one sensor node to another sensor node in this kind of condition it is necessary that we have to modify the code so that the requirement can be fulfilled. As this algorithm is available it can be easily hard coded to the sensor node at chip level. As this whole process is making communication between two sensor devices it is necessary that communication medium should also be reliable for which separate mechanism should be deployed. Furthermore for raising the security for such devices this algorithm can be infused with any other lightweight encryption algorithm. If possible than this solution can be tested on various other platforms.

VI. BENEFITS & LIMITATIONS

This Research work has some limitations like when applying this project or deploying it, it is necessary that both the systems must be connected to the server for effective communication. As the project is still in its early phase there is a limitation on how long or lengthy messages can be send. Particularly as this Application is developed taking Android Platform in consideration some time might be given for developing it for other platforms. Last but not the least this is not currently deployed hard-coded as still some changes required but this can also be done easily.

VII. CONCLUSION & FUTURE WORK

The simple conclusions are: yes, TEA is easy to implement, fast, efficient. If implemented and applied properly, TEA can be an excellent choice, particularly for encryption and decryption of small, short-term data in resource constrained devices. I have personally seen simple XOR algorithms used for "encryption" in scenarios where the programmers "do not have sufficient time, assets, or necessity for elaborate encryption algorithms." Tiny Encryption Algorithm would undoubtedly be a better substitute. Grounded on this research, I propose the following recommendations for using Tiny Encryption Algorithm:

- Choose algorithms carefully. Before choosing any algorithm we must study the benefits and characteristics of that algorithm. TEA is a private key cipher algorithm; Microsoft's mistake of using it as a hash function should be avoided. Many security weaknesses have arisen from a simple fact that a decent algorithm was used for the wrong purpose.
- Choose keys carefully and protect them. TEA is a private key algorithm, and its key must be protected. Avoid Microsoft's blunder of letting a critical private key be easily detected.
- Seek to understand the true implication of "weakness" statements. A "weakness" may have no real practical impact other than spreading "fear, hesitation, and doubt." Cryptanalysis algorithms should be distributed and freely tested and verified.
- Encryption authors should publish test suites to verify applications, to accommodate complete reference of implementations.
- Test/verify an encryption algorithm against known expected results before putting it to use.

REFERENCES

1. *Vikram Reddy Andem, A Cryptanalysis of the Tiny Encryption Algorithm, year 2003.*
2. *John A. Stankovic, Wireless Sensor Network, published in the year June 19, 2006.*
3. *Derek Williams, The Tiny Encrytion Algorithm, published in the year April 26, 2008.*
4. *Scheier, Bruce. A Self-Study Course in Block-Cipher Cryptanalysis. Cryptologia, Vol. 24(1). January 2000.*
5. *Steil, Michael. 17 Mistakes Microsoft Made in the Xbox Security System. October, 2005.*
6. *Steil, Michael. The Hidden Boot Code of the Xbox. Xbox-Linux. August, 2005.*
7. *A. Cerpa, J.Wong, L. Kuang, M. Potkonjak, and D. Estrin, Statistical Model of Lossy Links in Wireless Sensor Networks, IPSN, April 2005.*
8. *J. Elson, L. Girod, and D. Estrin, Fine-Grained Network Time Synchronization Using Reference Broad-casts, OSDI, December 2002.*
9. *S. Ganeriwal, R. Kumar, and M. Srivastava, Timing-sync Protocol for Sensor Networks, ACM SenSys, November 2003.*

10. *Rajesh K. Chakrawarti, Anurag Punde, Wireless Sensor Network, various issues and need of security-A Review,IJESRT October 2015.*
11. *Priyanka Mangal, Roopesh Kumar, Anurag Punde, Secure Communication in Wireless Sensor Network, year 2011.*