EXPERIMENTAL EVALUATION OF SECURE FAULT-TOLERANT SENSOR NETWORK COMMUNICATION PERFORMANCE

Project Reference No.: *45S_MTech_041*

College ***:*** *Manipal Institute of Technology, Manipal*
Branch ***:*** *Department of Computer Science and Engineering*
Guide(s) ***:*** *Dr. Radhakrishna Bhat*
Student(S) ***:*** *Ms. Pavithra Nayak K*

Keywords:

Wireless Sensor Network (WSN), Fault-tolerance, Brooks Iyengar algorithm, Security

Introduction:

WSN is gaining huge importance these days. When sensors are combined forming a wireless network, the results obtained are highly useful. There could be hundreds to thousands of sensor nodes in a wireless sensor network, forming a distributed system [1]. There are a variety of applications being developed using these networks. The sensor data plays a vital role in many IoT applications [2].

The widely known issue with these sensors is that the value obtained from sensors are not always accurate [1]. Once the sensor senses the data, it sends this information to its neighbours for further processing. A sensor node might fail to respond due to conditions like failure of link, energy depletion, radio interference, environmental calamities or desynchronization. When a faulty sensor node starts giving out erroneous data and the network is not aware of them, then it results in failure of estimation of sensor data or produces inaccurate data [3]. The popular algorithm for fault tolerance in WSN is the Brooks Iyengar algorithm. The Brooks Iyengar algorithm gained wide popularity because of its seamless approach in real time applications. Every sensor in the network runs this algorithm. The results from each such processing elements are put together and a weighted average over the midpoints of the region is found [1]. The practicality of this algorithm was such that the prominent defense agency, Defense Advanced Research Projects Agency (DARPA) has used it for collecting sensor data in real-time. Also, another prominent UK defense manufacturer, The Thales Group has used this algorithm. The Brooks Iyengar algorithm has made a great impact in WSN and is going to play a vital role in moving the systems towards automation [2].

As the applications are rapidly increasing, it is important to address the security concerns around this network. If left unaddressed then it calls out for many security attacks. Building a secure model for the WSN is restricted because of its resource constraint environment such as limited memory, CPU and energy. Also, the wireless channel creates an open platform for the attackers to get involved with the malicious intents. The designing of security model for these systems is challenging for the designer [4].

Objectives:

The fault tolerant network has become the essential feature for WSN. But in order to achieve a reliable communication, it's equally important to implement a secured layer around this fault tolerant network.

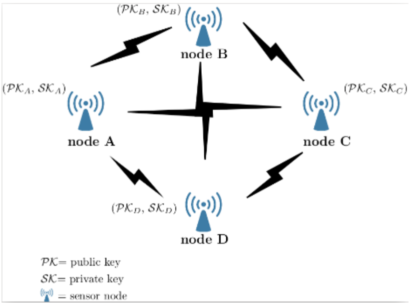
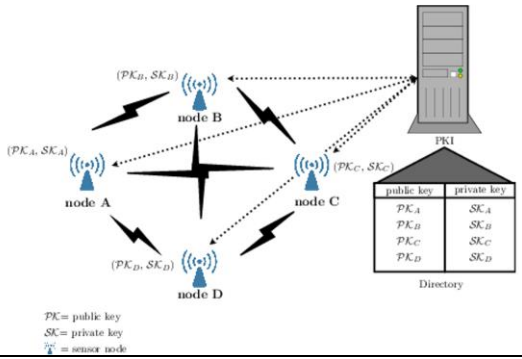
In Brooks-Iyengar algorithm, each sensor node communicates with every other sensor node. The sensor nodes running the algorithm are transmitting the data without any security mechanisms in place. To overcome this gap, the cryptographic algorithms are integrated with the Brooks-Iyengar algorithm to enable the security mechanisms around the network. In this project, we have the following objectives:

- To identify various secure communication methods for secure fault-tolerant sensor network communication
- To implement ECC based fault-tolerant Brooks-Iyengar algorithm
- To evaluate the performance of experimental results and highlight the suitable applications

Methodology:

Elliptic Curve Cryptography (ECC) is an efficient public-key cryptography best suitable for resource constrained system [5]. The Elliptic Curve Discrete Logarithm Problem (ECDLHP) shields with great security for the algorithm and imposes high challenge for attacker to retrieve the keys. Currently there is no efficient algorithm to break into ECC keys [6]. This feature of ECC protects the system from any attacker to regenerate the private key.

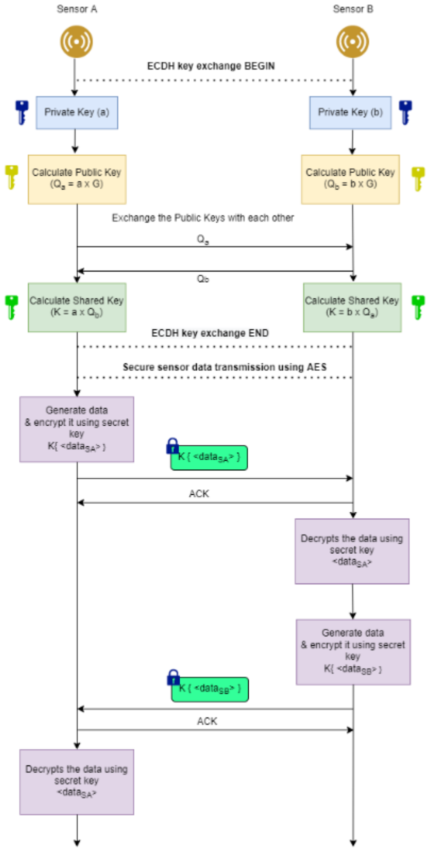
There are two techniques for the key (public and private key) generation and distribution:

Technique #1: Each sensor node is responsible for the generation of its own public key and private key pair. Once created, each sensor node should share its public key to all other remaining sensor nodes in the network.  $(\mathcal{PK}_B, \mathcal{SK}_B)$ node B $(\mathcal{PK}_A, \mathcal{SK}_A)$ $(\mathcal{PK}_C, \mathcal{SK}_C)$ node A node C $(\mathcal{PK}_D, \mathcal{SK}_D)$ node D $\mathcal{PK}$= public key $\mathcal{SK}$= private key = sensor node	**Technique #2:** A third-party application known as Public Key Infrastructure (PKI), is involved in managing the keys. PKI is responsible for generating required number of public key and private key pairs for all the sensors in the network. Also, it is the responsibility of PKI to distribute the public key of a sensor to all other sensor nodes.  $(\mathcal{PK}_B, \mathcal{SK}_B)$ node B PKI $(\mathcal{PK}_A, \mathcal{SK}_A)$ $(\mathcal{PK}_C, \mathcal{SK}_C)$ node A node C public key private key $\mathcal{PK}_A$ $\mathcal{SK}_A$ $\mathcal{PK}_B$ $\mathcal{SK}_B$ $\mathcal{PK}_C$ $\mathcal{SK}_C$ $\mathcal{PK}_D$ $\mathcal{SK}_D$ Directory $(\mathcal{PK}_D, \mathcal{SK}_D)$ node D $\mathcal{PK}$= public key $\mathcal{SK}$= private key = sensor node

ECC can be used in two ways:

(1) Shared Secret Generation:	**(2) Key and Encryption / Decryption:**
Here, same shared secret exists between any two sensors. The popular mechanism which can be used in this approach is Elliptic Curve Diffie-Hellman Key Exchange (ECDH). Once the secret key is available, it can then use symmetric cryptosystem such as Advanced Encryption Standard (AES) for encrypting and decrypting the data.	Here, ECC is the only cryptosystem responsible for the key generation as well as for the encryption and decryption. At first, each sensor nodes acquires the private key and public key pairs. Once the public key is distributed among all the sensor nodes, the sensor starts encrypting the data using its own private key. Receiving sensor will be able to decrypt the data using the corresponding sensor node's public key.

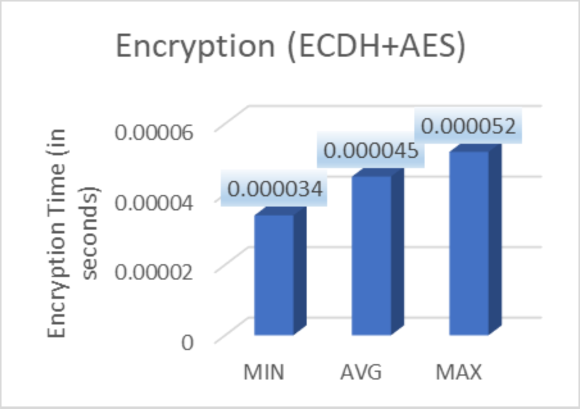
This project is developed on a prototype model of Brooks Iyengar algorithm. The suitable option is to choose Technique #1 and to use the symmetric cryptography. The ECDH approach is used for key generation and distribution and then AES for encryption and decryption. The implementation is shown in below diagram.

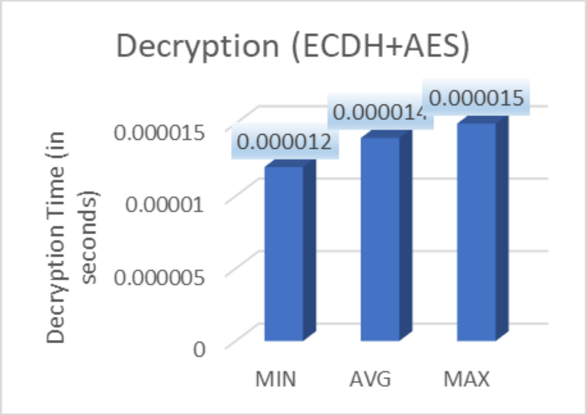


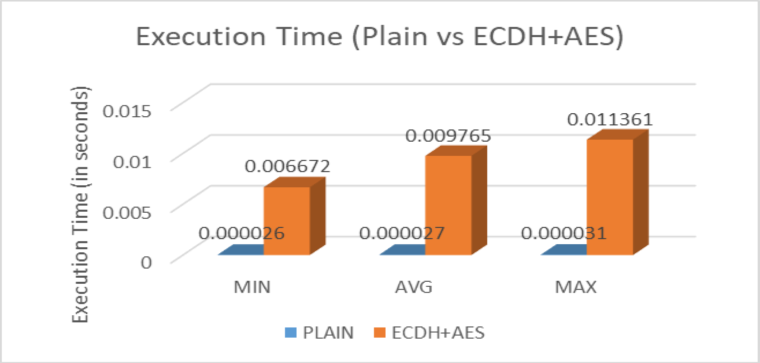
Sensor A
Sensor B
ECDH key exchange BEGIN
Private Key (a)
Private Key (b)
Calculate Public Key ($Q_a = a \times G$)
Calculate Public Key ($Q_b = b \times G$)
Exchange the Public Keys with each other
Q_a
Q_b
Calculate Shared Key ($K = a \times Q_b$)
Calculate Shared Key ($K = b \times Q_a$)
ECDH key exchange END
Secure sensor data transmission using AES
Generate data & encrypt it using secret key K{ $<data_{SA}>$ }
K { $<data_{SA}>$ }
ACK
Decrypts the data using secret key $<data_{SA}>$
Generate data & encrypt it using secret key K{ $<data_{SB}>$ }
K { $<data_{SB}>$ }
ACK
Decrypts the data using secret key $<data_{SA}>$

Conclusion:

The focus of this project is towards the performance analysis of Brooks-Iyengar algorithm when integrated with security algorithms. Below graph represents the analysis conducted. It can be seen that the time taken is comparable with the existing system.







Adding security around the sensor network, protects it from malicious attacks. In recent years, real-time security has become a fundamental requirement for the overall performance of the many of the safety critical system applications.

Scope for future work:

Further the project can be extended for the performance analysis using the pure ECC based approach where both key generation and encryption/decryption can be implemented by asymmetric cryptography.

Reference:

1. Sniatala, P., Amini, M.H. and Boroojeni, K.G., 2020. *Fundamentals of Brooks–Iyengar Distributed Sensing Algorithm*
2. Kumar, V., 2013. Impact of brooks-iyengar distributedsensing algorithm on real time systems. *IEEE Transactions on Parallel and Distributed Systems*, *25*(5), pp.1370-1370.
3. Panigrahi, T., Panda, M. and Panda, G., 2016. Fault tolerant distributed estimation in wireless sensor networks. *Journal of Network and Computer Applications*, *69*, pp.27-39.
4. Manavendra S, Rohan V. and Sarthak Mittal, *Security in Wireless Sensor Networks*
5. Vanstone, S.A., 2003. Next generation security for wireless: elliptic curve cryptography. *Computers & Security*, *22*(5), pp.412-415.
6. “ECC keys,” Elliptic Curve Cryptography (ECC) - Practical Cryptography for Developers. [Online]. Available: https://cryptobook.nakov.com/asymmetric-key-ciphers/elliptic-curvecryptography-ecc