Enhancing Security In Nebulae Framework

Submitted By

Jenishkumar Maheshbhai Patel

16MECE15

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

INSTITUTE OF TECHNOLOGY

NIRMA UNIVERSITY

AHMEDABAD-382481

May 2018

Enhancing Security In Nebulae Framework

Major Project

Submitted in partial fulfillment of the requirements

for the degree of

Master of Technology in Electronics & Communication Engineering
(Embedded Systems)

Submitted By

Jenishkumar Maheshbhai Patel

(16MECE15)

Guided By

Prof. Akash Mecwan

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

INSTITUTE OF TECHNOLOGY

NIRMA UNIVERSITY

AHMEDABAD-382481

May 2018

Declaration

This is to certify that

1. This thesis consist of my original work towards the degree of Master of Technology in Embedded Systems at Nirma University and has not been submitted elsewhere for a degree.

2. Due acknowledgment has been made in the text to all other material used.

- Jenish M Patel

16MECE15

Disclaimer

“The content of this paper does not represent the technology,opinions,beliefs, or positions of System Level Solution corporation Pvt. Ltd.,its employees, vendors, customers, or associates."

Certificate

This is to certify that the Major Project entitled **"Enhancing Security in Nebulae Framework"** submitted by **Jenishkumar Maheshbhai Patel (16MECE15)**, towards the partial fulfillment of the requirements for the degree of Master of Technology in Embedded Systems, Nirma University, Ahmedabad is the record of work carried out by him under our supervision and guidance. In our opinion, the submitted work has reached a level required for being accepted for examination. The results embodied in this major project, to the best of our knowledge, haven't been submitted to any other university or institution for award of any degree or diploma.

Date: Place: Ahmedabad

Prof. Akash Mecwan
Internal Guide

Dr. N.P. Gajjar
Program Coordinator

Dr. D. K. Kothari
Section Head, EC

Dr. Alka Mahajan
Director, IT

Certificate

This is to certify that the major project entitled **"Enhancing Security in Nebulae Framework"** submitted by **Jenishkumar Maheshbhai Patel(16MECE15)**, towards the partial fulfillment of the requirements for the degree of Master of Technology in Embedded Systems, Nirma University, Ahmedabad is the record of work carried out by him under our supervision and guidance. In our opinion, the submitted work has reached a level required for being accepted for examination.

Mr. Apurva Patel
Software Engineer
SLS Corp.Pvt.LTD
Anand

Acknowledgements

I would like to express my gratitude and sincere thanks to **Dr. N.P.Gajjar**, PG Coordinator of M.Tech Embedded Systems for guidelines during the review process.

I take this opportunity to express my profound gratitude and deep regards to **Prof. Akash Mecwan**, guide of my internship project for his guidance, monitoring and constant encouragement.

I would like to thank **Mr. Apurva Patel** and **Mr. Nilesh Vora**, for the amazing opportunity to work under them at SLS Corp.Pvt.Ltd.,Anand. I have learned so much from this project. The guidance you have given me is greatly appreciated.

- Jenish M Patel
16MECE15

Contents

List of Figures

List of Acronyms

OS	Operating System
SSL	Secure Socket Layer
TLS	Transport Layer Security
DTLS	Datagram Transport Layer Security
IoT	Internet of Things
WSN	Wireless Sensor Networks
SHA-256	Secure Hashing Algorithm-256
AES	Advance Encryption Standard

Abstract

Sensor nodes are widely used for automation. These sensor nodes communicate wirelessly with each other to form a wireless sensor network. As the sensor network performs a critical task in automation. This arises a security issue for sensor nodes communication. As the data during communication could be manipulated and send to the destination which may lead to huge loss of resources and environment. This thesis discusses about securing communication data for Contiki OS as it is one of the operating system used for sensor nodes. Datagram Transport Layer Security (DTLS) which provides features such as cookie verification, authenticity, integrity, and confidentiality for data communication. Thus DTLS implementation would help to solve security issue upto some extent. Implementation of DTLS on Contiki operating system and working flow of DTLS is been discussed in this thesis. NXP JN5168 microcontroller has been used as sensor node for hardware implementation.

Chapter 1

Introduction

1.1 Introduction

Internet is used to connect devices and make the, communicate. this devices form a network of devices which could be of same or different type. Internet is widely used and is increasing. With the help of internet communication has become easy around the world. One can control devices at home while sitting in other country. Internet is considered a major component of automation as automation in every place is widely increasing. This automation uses sensors which communicate with each other and form a network of sensors.

Sensors such as thermo, light, IR sensors which are wirelessly connected to internet form a wireless sensor network and play a crucial role in automation. This wireless sensor network does not require any high performance computer to computing they send reading of the task assigned to it to the user which performs computing. Wireless Sensor Network communicating through internet is also known as Internet Of things IoT. Thus IoT making life easier by making any thing easily controllable from anywhere in the world.

With this many advantages there are also many disadvantages of IoT. The IoT devices access would be gained by some unauthorized person and can misuse the network such as if someone gets access to the automated house and break into it for misusing it or to rob it. Thus security for IoT devices has raised a major concern. This could me more dangerous when someone gets access to power plant and changes some data which could

lead to disaster. As a result securing the communication between IoT is more important. Transport layer security is widely used for communication over internet. Operating system such as Contiki OS which is widely used for IoT devices because of it small footprint still does not support TLS. Thus implementing of TLS on contiki OS would help to solve the problem to some extent. As TLS is used for transport control protocol which is a connection oriented protocol and IoT requires connection-less protocol which is User Datagram Protocol thus TLS could not be directly implemented on contiki OS.Datagram Transport Layer Protocol (DTLS) is made instead of TLS for connection-less protocol. Thus taking this idea we have to run DTLS on contiki OS.

1.2 Motivation

As the use of internet increases and automation has been introduced in daily life style to make life easier. Thus wireless sensor network in automation is wide to used in which small sensors are deployed to perform tasks. This bring vulnerability to daily life if a wireless sensor network data maybe accessed by mischief person. Thus security is to be introduced in wireless sensor network. Contiki is widely used for WSN thus implementation of DTLS is to be done. DTLS provides security for the WSN DTLS will provide security for transport layer by encryption of data to be sent using asymmetric way of encryption.

1.3 Objective

Implement DTLS on contiki to provide a secure communication between sensor node. This would include integration of DTLS library with contiki OS. Thus when a data is sent through a sensor node it must be encrypted and safe for transmission.

1.4 Organization of Thesis

First chapter includes introduction, motivation and objective of the thesis. Chapter second shows what Nebulae is and what does it do,it also includes some Nebulae products.

Chapter three shows difference between TLS and DTLS and why we prefer DTLS for WSN, it show algorithm for RSA protocol. Chapter four includes some platform for cooja simulation and basic steps for integration of contiki with DTLS and some API of DTLS library and hardware implementation of the project. Chapter five conclusion of the thesis.

Chapter 2

Nebulae

2.1 Introduction

Under sls corp Nebulae is the team and ecosystem of IoT. Nebulae is an ecosystem of IoT which provides networking hardware. Nebulae also provides cloud services. All the system in Nebulae designed and engineered to simplify the efforts for IoT development.

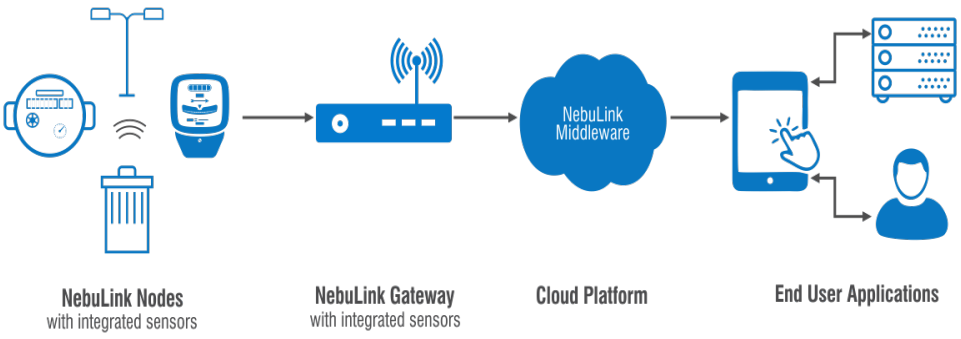

Figure 2.1: Nebulae[4]

Nebulae consists of three basic products which are

- Nebulink node
- Nebulink gateway
- Nebulink middleware

All the three component contribute for a complete Nebulae network.

2.2 Nebulink Node

Nebulink Node are the node which are created by SLS which are wireless devices which contribute to the development of wireless sensor nodes. This sensor form a network resulting in a wireless sensor network. This wireless sensor network are deployed in dense are and large with scalability ensuring minimum amount of maintenance cost. Nebulae provides a network which is self configuring, self healing and self sustaining network. Nebulae ensures detection and re-configuring of new routing path when a node is broken and grantee data delivery with high quality of service. Nebulae is designed to protect against any failure and if any failure occurs it would provide run-time recovery to the failure.

Features of nebulink nodes are

- it provides a network which configures by itself with fast network formation
- Nebulae network provides self healing feature with ensures safe delivery of message
- security of the network connection with the network devices is secured
- network synchronization is very accurate
- Very low power consumption
- low installation cost
- fast and wireless update feature
- Nebulae 2.4 Ghz nodes for HAN and sub-Ghz nodes for NAN
- SPI, I2C and UART could be used for sensor interface

2.3 Nebulink Gateway

Nebulink gateway used by user and IoT developer to access IoT functionalities such as home automation, industrial automation and smart cities. Nebulink gateway is used for integrating sensors and different networking protocols. Nebulink gateway is used to control embedded system, sensing and management of sensor nodes. Advances peripheral mechanism allows wirelessly controlling and configuring devices for remote loaction by the use Nebulae Cloud services. Nebulae cloud allows programmer to integrate Nebuling gateway to their cloud[4].

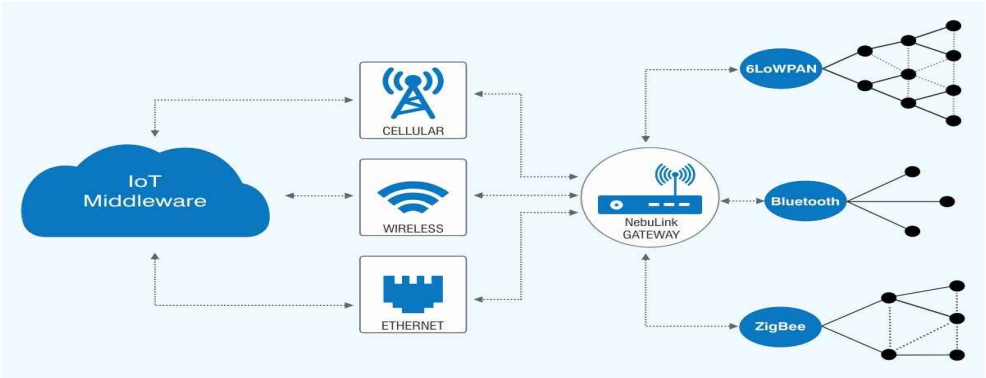

Figure 2.2: Nebuling Gateway[4]

Nebulink Gateway is unique due to its advanced features to provide hybrid network management and message transfering from one network to another network. This unique feature facilitated any sensor devices from two different frameworks to directly communicate with each other without using cloud this means sensor node from Zigbee can communicate to sub-Ghz sensor node. This advanced feature of Nebulink Gateway facilitate smart cities technology developers as the feature of many devices reside on one single Gateway as it allows sensor nodes to directly communicate to smart cities[4].

Nebulae has developed an industrial gateway which is Nebulink gateway which is a high performance based on OSGi standards IoT Gateway for industries and heavy duty applications. To fulfill industrial requirements it can support protocols as ModBus, CAN, RS-232, RS-485, I2C, SPI.

Features of Nebulink industrial Gateway are as follows

- software framework of gateway is modular and dynamic
- simple and easy to understand SDK
- devices could be wirelessly operated
- wireless updation for application
- gateway for industrial and home automation works independenly for designated application
- Advanced routing protocol for nebulae gateway
- WiFi, GSM(2G,3G), Ethernet,PowerLine, 802.15.4, ZigBee and Modbus can be used for communication

- communication is secured for cloud to gatway
- Nebulink gateway is able to connect to multiple cloud at a time
- Modbus, CAN, RS485, RS232 are supported
- Supports various interface options for Sensors like SPI, I2C, UART
- Devices agnostic gateway framework
- flexible NebuLink Gateway framework
- wireless Firmware update of NebuLink Gateway and NebuLink Nodes
- firmware updation could be manual or scheduled

2.4 Nebulink Middleware

Nebulink Middleware provide service for remote device management for Nebulae IoT devices. Nebullink middle-ware is a single platform that allows IoT devices such as Nebulink node to connect, operate and automate. Use of Nebulae Middle-ware SDK will help developers by reducing cost and time of development with its feature of rapid development for IoT application and products. Nebulink Middle-ware provides feature such as user friendly interface to setup and monitor IoT devices data.

Nebulink Middleware is for IoT provides large scalibility with failure proof connectivity. It also provides device interoperability, data consistency and security for all the devices which are connected to it. Data management and data analytic for machine learning and decisions making based on the data collected from sensor network on real time bases is done.

Features of Nebulink Middleware are as follows

- unlimited devices handling capability
- Remote handling capability of devices
- Data analytic
- wireless software update
- fault detection
- wireless diagnose of endpoint devices
- Fault tolerance
- Data could be collected from other Nebulae network

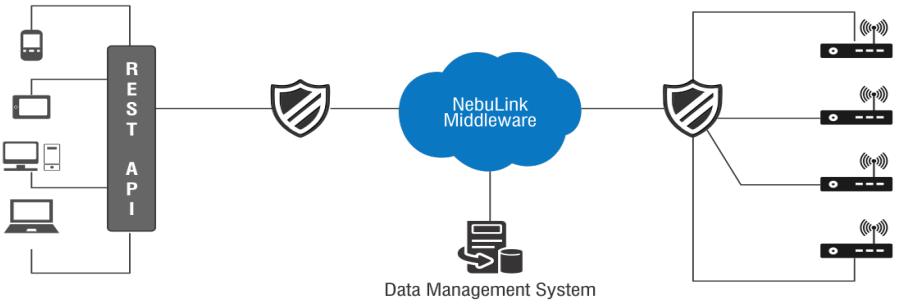

Figure 2.3: Nebulink Middleware

- nodes connected to the middleware could be controller easily
- Acknowledgment form endpoint for successful delivery of message

2.5 Nebulae Applications

Nebulae provides application in may field which are smart city, smart agriculture and smart manufacturing. With is vast applications Nebulae integrates received data from the network of wireless sensor network for achieving location, intelligent recognition, tracking, management and monitoring.

2.5.1 Smart City

Smart grid – Nebulae smart grid solution by providing smart meters in place of regular meters. Nebulae uses current and voltage sensors to achieving meter reading to eliminate human and mechanical errors.

Smart street light- Nebulae aims for smart street light to save energy and resources form being wasted.

Smart waste management – IoT network could help for waste management by creating a network of bins and which informs the user when the bin are fille with the location of the bin.

With this Nebulae has come through smart health, smart education, smart parking,

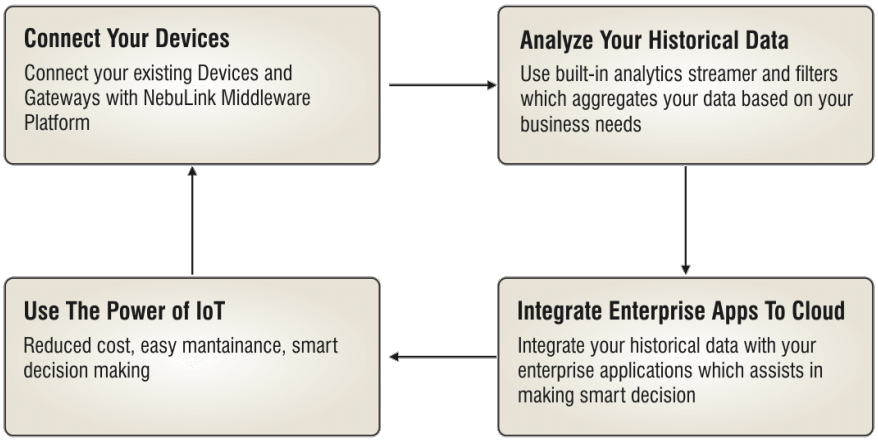

Figure 2.4: Nebulink Middleware Flow diagram[4]

security and public safety and smart water management where the IoT can be install to make the life easy and comfortable and to build a smart city.

2.5.2 Smart agriculture

As india is majorly depended on agriculture thus for farmer Nebulae provides solution for framers to increase the production using IoT devices. Installation of sensor on the field to reduce human errors and minimizing farmer efforts. Sensors for soil to maintain the moisture in the soil as per required.

2.5.3 Smart manufacturing

As the industrial automation has increased Nebulae has integrated IoT for the industries which are as per industry 4.0 requirement. Nebulae makes technologies and protocols for networking and easy manageability.

Nebulae gas, temperature, humidity, vibration and accelerometer sensor are integrated for industries to provide good quality of air, maintain and control temperature of the industry or a machine accordingly with environment. This is done to reduce human efforts and human errors.

Chapter 3

Security

3.1 Introduction

To overcome the IoT security issue discussed in chapter 1 for the Nebulae framework we have come across TLS/SSL type of security. As TLS/SSL would require larger footprint and IoT devices are of smaller footprint a new protocol is developed named Datagram Transport Layer Security (DTLS). DTLS is similar to TLS but it is used for small footprint devices. First we need to know what TLS/SSL is in section below.

3.2 TLS/SSL

Transport Layer Secuirty (TLS) or Secure Socket Layer (SSL) which is one of the best and widely used type of encryption. It is majorly used for web browsers, online transaction, Email etc. It is based on layed protocol. Record protocol layer in TCP is used to break the message into manageable blocks then compresses it if required, provides it with MAC address, encrypts it and then transmits it. TLS is used only for TCP based network. Record protocol encapsulate data on bases of four protocols which are as follows

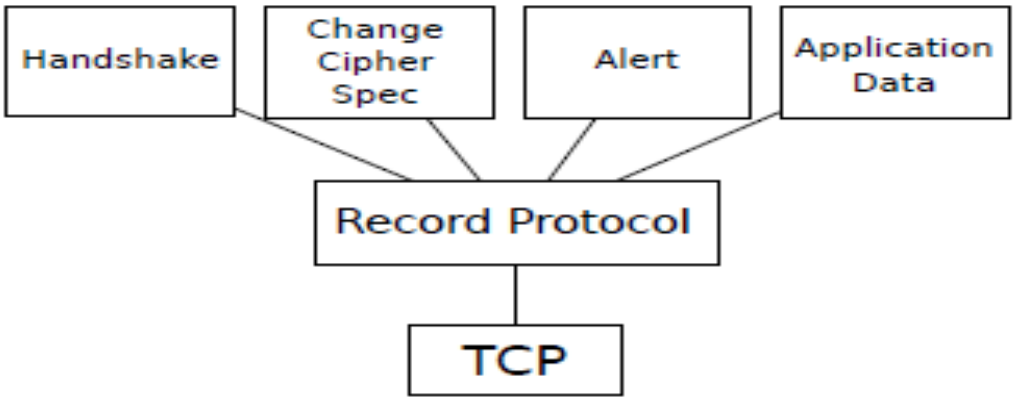

Figure 3.1: Structure of TLS Protocol[1]

3.2.1 Handshake Protocol

This protocol is responsible for deciding encryption algorithm and compression algorithm for the data to be transmitted securely. During this protocol key is decided and certificate id send to the other node for encryption of data for secure communication. Handshake protocol follows a simple handshake steps which as shown.

3.2.2 ChangeCipherSpec Protocol

This protocol informs during handshake protocol that the next data send will be encrypted and compressed according to previously send information and its key for decryption and decompression is also send during handshake protocol.

3.2.3 Alert Protocol

To prevent the connection to be closed or restart the certification processes by the receiver due to some error alert protocol is used. Alert Protocol is a message sent by the receiver as an error or a warning message to the sender.

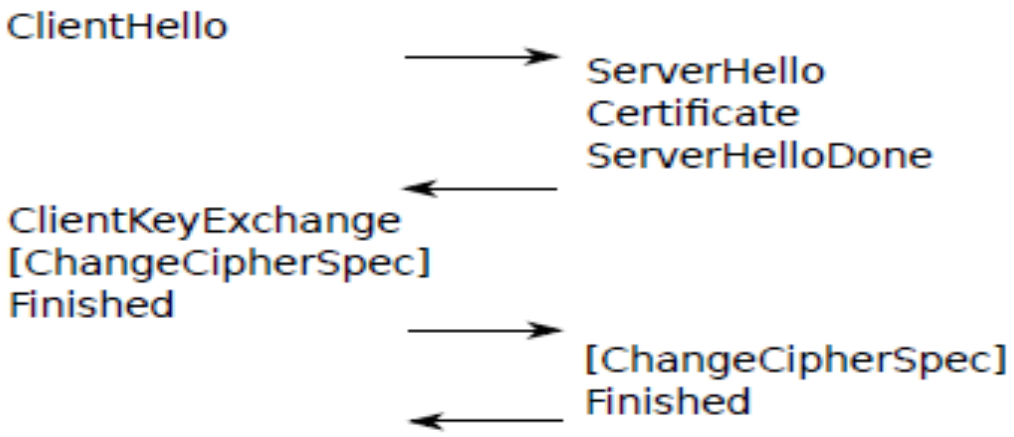

Figure 3.2: handshake Protocol[1]

3.2.4 Application Protocol

It is a protocol which transmits the encrypted data from the sender and ensure decryption on the receiver side.

3.3 Datagram Transport Layer Protocol (DTLS)

DTLS is similar to TLS. But TLS works only on TCP whereas we require security for wireless sensor network which has no stable link and connects on the bases of UDP. Thus DTLS is used for UDP network type. TCP is used where reliable network is required which ensures no retransmission of data and reliable delivery which is very necessary for TCP network as its application which are real time such as video call, audio calling,online gaming etc. This feature are not required for wireless sensor network as it requires security.

For DTLS all the protocol of TLS are been modified to eliminate unreliable networking for UDP type of networks. TLS is improved to make DTLS by modifying the header format and adding two more states in it which are epoch and sequence number as the new member of header format which were not present in TLS header format.

Sequence numbering the packets to identify the order of packets and arranging the

packets at the receiver on reception of out of order delivery of packets. Main advantage of sequence numbering of packets is that if any packet is lost it could be identified and only that particular packet is sent rather than sending the whole data which has be segmented. Epoch numbering specify that which cipher is used for encryption of data. TLS is vulnerable to DoS attack which has been overcome in case of DTLS by providing a technique which is a cookie exchange technique. In cookie exchange technique the server on receiving hello request does not start data exchange but the server ask the client for cookie whether the client is willing to communicate with the server. A regular handshake continues as the server get client reply on cookie verification.

DTLS also overcomes the disadvantages of UDP by sending the packet with sequence number and ensuring that all the packets are been received at the receiver and then the packets are rearranged to get the data.

Below is the figure which shows the modification made in DTLS and TLS. The one in box are extra parameter which are included in TLS and are the changes made to TLS.

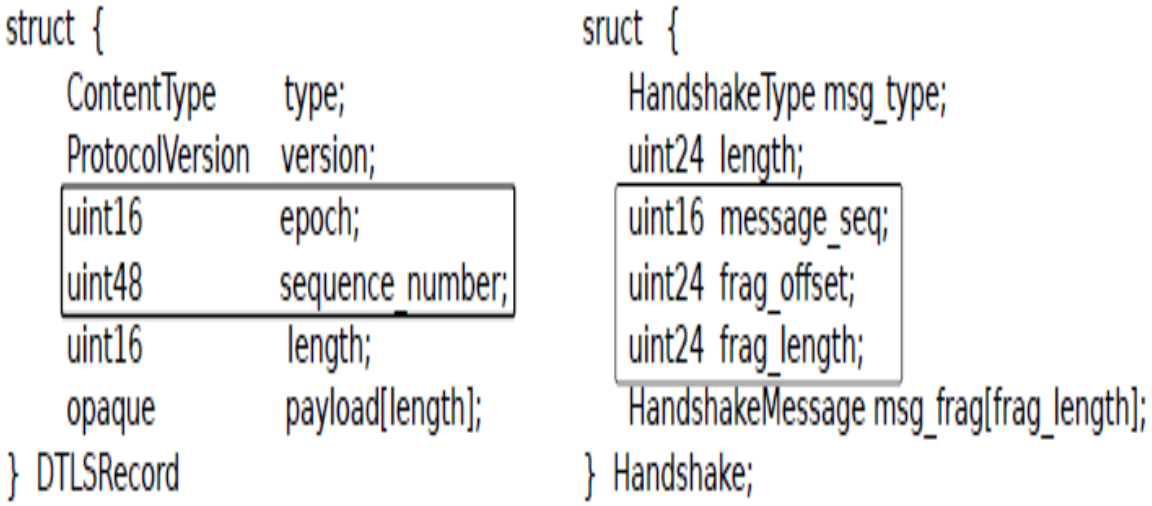

Figure 3.3: DTLS record format and DTLS handshake format[1]

3.4 Security Features

Major security features which are provided by SSL/TLS and DTLS are as follows

- Integrity
- Authenticity
- Confidentiality

3.4.1 Integrity

Integrity is provided by hashing of message. Hashing algorithms ensure that the data has not been manipulated or changed during transmission. Cyclic redundancy checksum (CRC) is used to check the integrity of the message. Examples of hash algorithm are SHA-256 and MD5. DTLS uses SHA-256 algorithm for hashing of message.

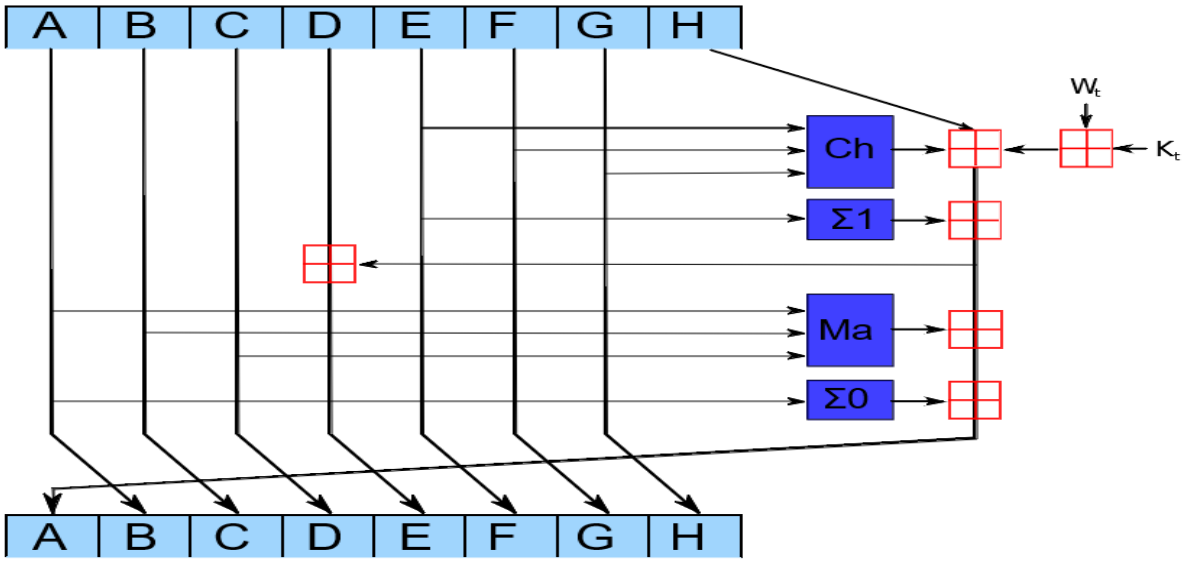

Figure 3.4: SHA-256 algorithm[5]

In SHA-256 algorithm the message or data to be hashed is padded and made multiple of 512 bits. The message is broken into blocks of data and then according the block diagram shown in figure 3.1 SHA-256 algorithm is applied. The output for any length message will be 256 bit data. If there is a minor changes between two input message then

it output data would be completely different. For example hash value for

hello is b5dbadee9efdf8173017d85134fd23112904176fde136a5e694edb125d637d5c then hello1 is 2e4ffc038bad379dd5b5ba6ce0fd6ba84bd139f55f62d251c13d146acc4abff9

3.4.2 Authenticity

Identity of a the communication nodes are checked under authenticity. The certificate of the sender is required to provide whether the sender is authentic or not. If the certificate fails to prove the authenticity of the sender then no further communication is made until a valid certificate is received. DTLS uses eclipse curve digital signature algorithm (ECDSA) is use to checking the authenticity of a node.

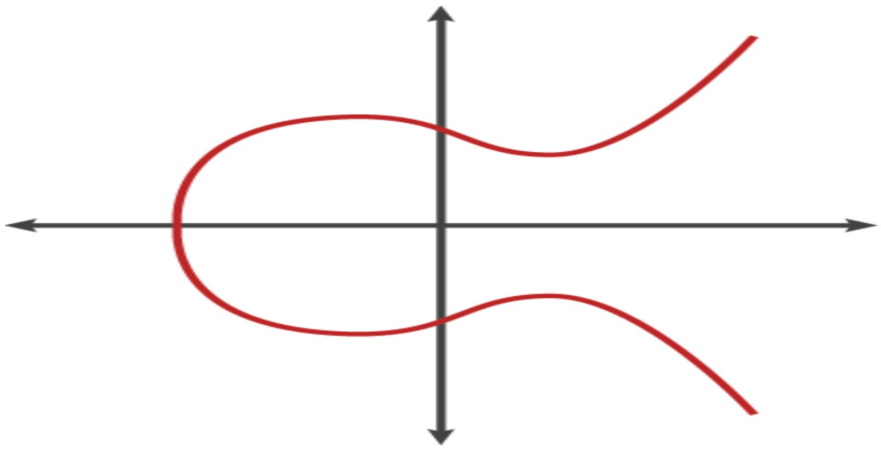

Figure 3.5: Eclipse Curve for ECDSA[6]

A certificate in ECDSA consists of public key and private key will be with the authenticator. To provide authentication first variable length data is converted into a fixed length message. The fixed length signature message consists of two integers. These two integer are computed according to the curve equation and if it matches the curve equation sender is validate and further communication takes place else sender is disconnected.

3.4.3 Confidentiality

Confidentiality means the message sent through open communication link must be meaningless for all other receiver except for the designated receiver. The designated receiver would have a special key to open the message and make it readable and meaningful. Advanced Security Encryption (AES) is used as encryption algorithm for DTLS. AES is a 128 bit encryption algorithm which has 128 bit encryption key.

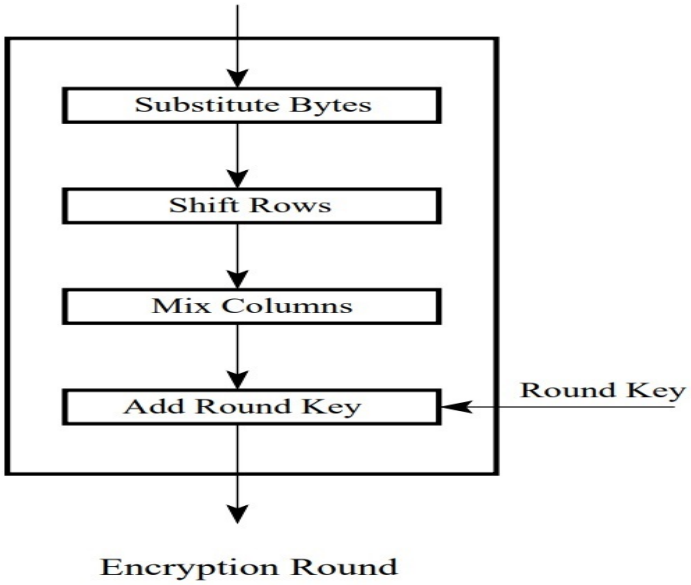

Figure 3.6: AES encryption[7]

AES key is first substituted into a 4x4 matrix then S-box values are substituted accordingly. Second row of the matrix is shifted onces to scatter all the elements then a random column is selected and mixed in the matrix. Finally round key is added to the matrix this process is repeated 10 times for AES-128bits.

Chapter 4

Implementation

4.1 Introduction

In this chapter talks about the implementation of DTLS on contiki. Different functions of DTLS library and platform of contiki. Initially I used Ubuntu 16.04 64bit OS when I run contiki and DTLS codes it would show relocation truncated to fit error the I came to know that DTLS should be run on Ubuntu 32bit OS as DTLS is compatible with 32bit OS other problem faced by me was due to the platform of contiki which has small footprint and DTLS require large than it. We could use native and minimal-net platform of contiki which are for testing and has no limit of footprint.

4.2 Contiki OS

Contiki OS was developed by Swedish Institute of Computer Science. Contiki operating system is an open source small foot print software. Contiki provides features such as high performance and multi tasking. Due to the above mentioned features contiki can be used for IoT devices which has limited amount of memory. Contiki 3.0 is used for implementation of DTLS. Contiki 3.0 provides platform for NXP-JN5168. But does not provide cooja simulation for JN516x.

Contiki OS is an open source operating system. Contiki OS can be download into two forms one is pre installed contiki known as instant contiki and other is contiki file

which must installed into Ubuntu. Contiki has many version of which latest is Contiki-3.0. Contiki package comes with a simulator and all the development tools requires for the nodes. Contiki platform folder consists of all the supported platforms.

Contiki programming start with declaring a process than that process must be started using autostart process. A contiki process is defined using process thread and process begins with process start and ends with process end. In contiki OS process are created to perform tasks. Contiki OS has three types of Stimer, Etimer and Ctimer. Stimer is a simple timer, Etimer is an event timer and Ctimer is a callback timer.

4.3 Cooja Simulation

Cooja simulation is started by using command ant run in tools/cooja folder. Once cooja is started new simulation is created as shown in figure 5.1.

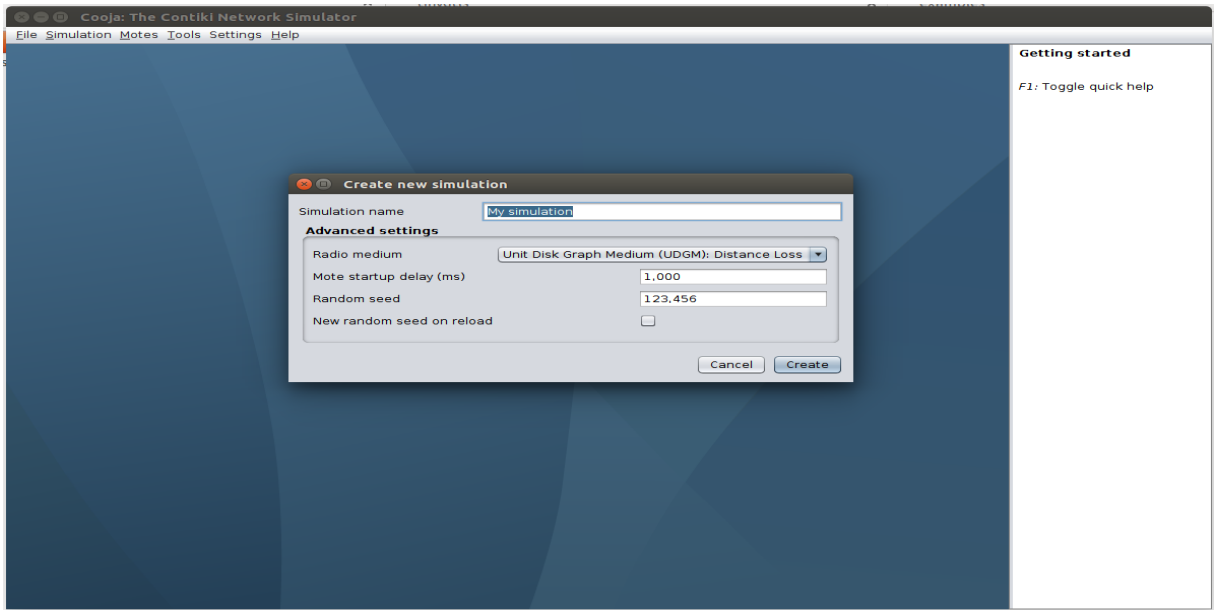

Figure 4.1: Cooja simulator

While creating a new simulation radio medium, mode startup delay and random seed are to be selected for mote creation. Once a simulation is created different types of windows opens in which one shows simulated nodes and its transmission range and direction

of flow of data other window shows the output of all the nodes with time stamp of debug print of nodes. Prints of different nodes are separated by different colors as seen in figure 5.3. Simulation could be started and stopped at any moment. Speed of simulation could also be varied accordingly. To start simulation different types of motes are selected as per requirement form motes tab and program required to dump is complied as shown in figure 5.2. Cooja mote is a general mote used for testing purpose only. When required nodes are created and their positions are set then simulation could be started and the output of the nodes can be seen on mote output window. Direction of flow of data could also be seen in figure 5.3 and simulation can be stopped and started at any moment. Led output can also be seen using led window.

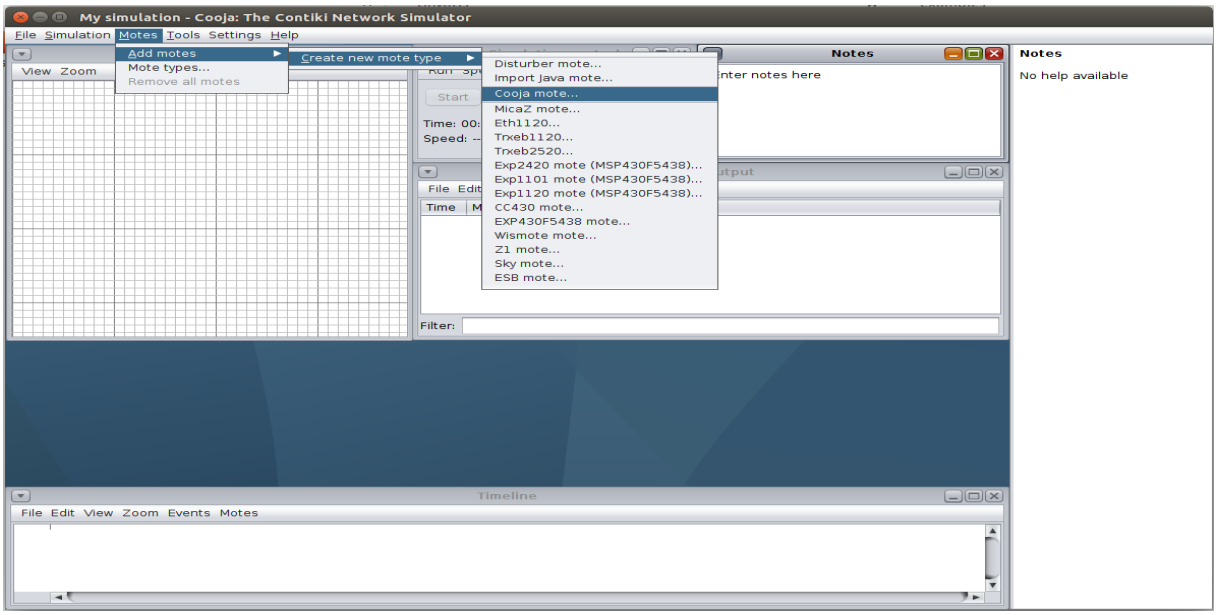

Figure 4.2: Mote selection for Cooja

4.4 DTLS in contiki

Contiki does not come with the builtin library of DTLS we have to seperately download and integrate both the files. Some basic steps are given on DTLS library by which we could link the two files which are 4.3.1 place tinydtls in the apps folder of contiki 4.3.2

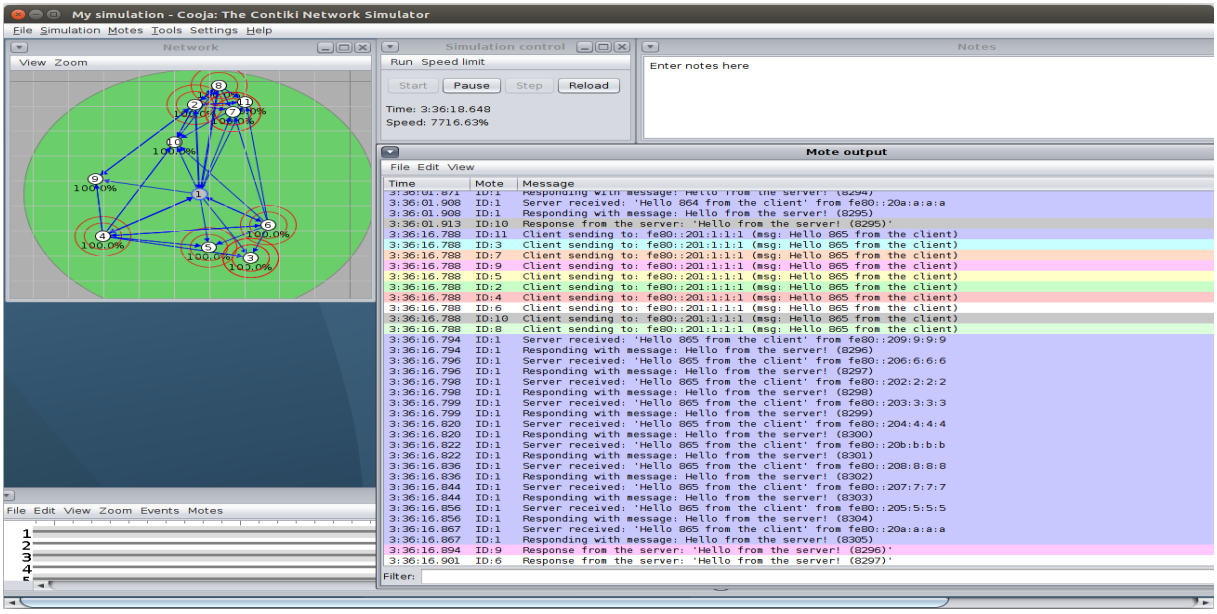

Figure 4.3: Mote output on Cooja

then run git submodule update 4.3.3 open tinydtls folder and run autoreconf 4.3.4 then ./configure –with-contiki 4.3.5 and finally we could run our code using make file there are sample server and client code that could perform DTLS based communication on cooja simulation.

4.5 DTLS API functions

Following are the API functions in the DTLS library • dtls_init():- this is to initialize memory management • dtls_handle_message():- incoming DTLS message is handled with this message and it takes data, data length, session and context details as an argument • dtls_connect():- to establish connection between the provided peer this API takes session details and peer details • dtls_close():- it is used to close the DTLS connection • dtls_write():- writes the data to the specified peer from the buffering • dtls_check_retransmit():- it check the sequence of the packets if missing then re-transmit that packet • dtls_new_context():- creates a new context object • dtls_set_handler():- sets the callback handler object to the provided data above shows some basic functions

of DTLS to setup a network connection.

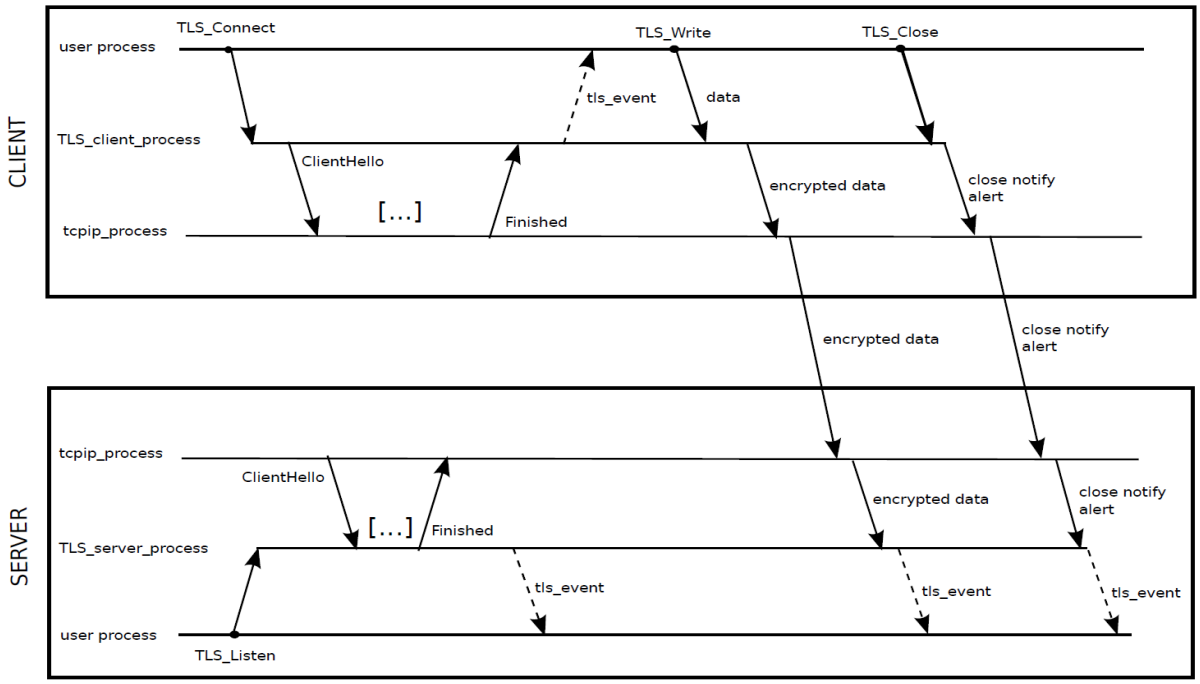

Figure 4.4: DTLS API Flow[1]

4.6 DTLS Flow

To start DTLS based communication first a DTLS handshake is to be made. To start DTLS handshake dtls_connect() function is called with the recipient address. When dtls_connect is called the node calling the function becomes a client node and checks whether node is already connected to the given address if not connected client transmits client hello packet and adds the server node to its peer. This process is also done when dtls_write() is sent when the recipent address is not added to peer of client. A receiver node receives the client hello packet and sends it to dtls_handle_read() function. As all received DTLS packets are handled by dtls_handle_read() function.

Once the receiver node gets client hello it becomes server node and sends hello verify request with cookie data. Once client receives hello verify request packet it extracts cookie data and retransmits client hello with the cookie data in it. On receiving client

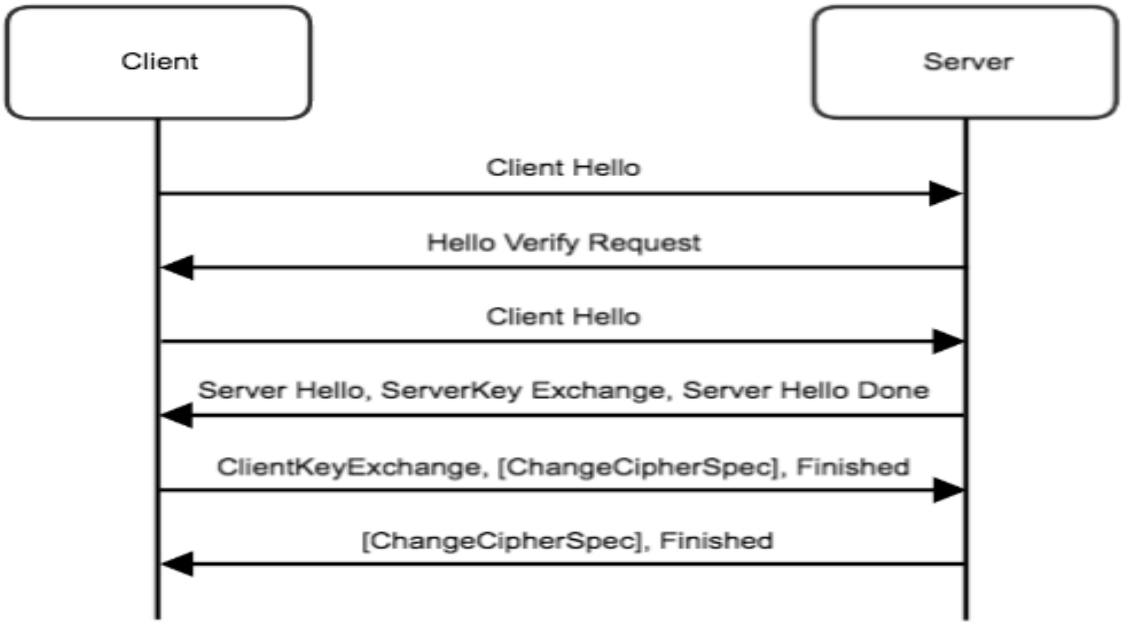

Figure 4.5: Original DTLS handshake Flow

hello with cookie server extracts its cookie and compares it with previously sent cookie if the received cookie matches with the sent cookie further communication is processed else if the cookie does not match the connection is terminated. After successful cookie verification server send server hello server key and server hello done packet. When client node receives server hello done packet it sends its client key exchange packet, cipher change packet and encrypted finish packet. Server node upon successful reception of finish message sends server finish message. After successful reception of finish message on both the sides now encrypted communication takes place. This encryption is done based on the encryption keys exchanged during handshake. This flow contains some data packets which are transmitted simultaneously dues to which some packets are not been received properly. Thus all the packets must be retransmitted. This would lead to degradation of handshake time. Thus new flow is created.

In this flow only one packet is sent at a time if a packet is dropped or degraded during communication only one packet needs to be retransmitted. This would provide faster handshake with less latency. In this client node send client hello. Upon successful

Figure 4.6: DTLS Flow for Nebulae

reception of client hello packet server node sends server hello. Client node on receiving server hello send client key exchange message. Server node sends server hello done on receiving client key exchange packet. Then change cipher spec message is exchanged. Once both the nodes receives change cipher spec message it changes its modes for the specific node to receive only encrypted message. Thus client send encrypted finish message to which server node send encrypted finish message. After finish message is received on both the side now the encrypted data transmission could take place.

4.7 Hardware – JN5168

NXP-JN5168 microcontroller is used for hardware implementation of contiki OS with DTLS based communication. JN516x is a series of ultra low power NXP microcontroller. JN5168 is a 32 bit RISC devices with 256kb flash, 32kb RAM and 4kb EEPROM. Communication is done using 2.5GHz IEEE802.15.4 ZigBee which is used for automation and smart energy as it consumes less amount of power. JN5168 provides high performance

computing and networking. JN516x platform requires BA-ELF-GCC and its SDK which are requires to install separately to compile code for JN516x. After successful compilation of code for JN516x in Contiki OS a binary file is generated. The programming of JN616x is done using NXP IDE named Beyond Studio.

Figure 4.7: NXP-JN5168 [8]

4.7.1 Beyond Studio

Beyond studio is an Eclipse IDE for NXP-JN51xx microcontrollers. Beyond studio is used to develop programs for NXP-JN51xx microcontrollers. It is also used for programming the microcontrollers.

Figure 4.8: Beyond Studio

Chapter 5

Conclusion

Initially the data transmitted by sensor node running contiki OS was not secured and open to vulnerability. Thus to secure data communication for sensor nodes running contiki operating system was needed.

DTLS library was implemented on Contiki OS as it provided secure data communication. DTLS provides secure data communication by encrypting the data to be sent in a networking.

Contiki OS with DTLS library was then ported to NXP based JN5168 microcontroller board developed by System Level Solution.

Bibliography

[1] [1] Vladislav Perelman,“Security in IPv6-enabled Wireless Sensor Networks: An Imple- mentation of TLS/DTLS for the Contiki Operating System”, thesis for conferral of a Master of Science in Computer Science, June 29,2012.

[2] [2] M.Suresh, P.Saravana Kumar, S.M.Ramesh, Dr.T.V.P.Sundararajan and S.Muthusamy,“An Efficient Implementation of RSA Data Encryption Algo- rithm In 8051”, International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE) Volume 3, Issue 11, November 2014.

[3] $htts : //www.memsic.com/info/aceinnaâlanding.cfm?nu = /wirelessâsensorânetworks/MPR2400CBmpn = MPR2400CB.$

[4] $https : //www.nebulae.io/$

[5] $https : //en.wikipedia.org/wiki/SHA - 2$

[6] $https : //blog.cloudflare.com/ecdsa - the - digital - signature - algorithm - of - a - better - internet/$

[7] $http : //www.contiki - os.org/start.html.$

[8] $http : //fcs.futureelectronics.com/2014/08/nxp - jn5168 - 001 - myy - jennet - ip - zigbee - pro - and - ieee802 - 15 - 4 - module/.$