Software Stack for IoT Devices

A B. Tech Project Report Submitted
in Partial Fulfillment of the Requirements
for the Degree of

Bachelor of Technology

by

Kshitij Mahendra Ghodake and Garima Ranjan
(112101025 and 122101011)

under the guidance of

Dr. Vivek Chaturvedi

INDIAN INSTITUTE
OF TECHNOLOGY
PALAKKAD

COMPUTER SCIENCE AND ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY PALAKKAD

CERTIFICATE

*This is to certify that the work contained in this thesis entitled "**Software Stack for IoT Devices**" is a bonafide work of **Kshitij Mahendra Ghodake and Garima Ranjan** (**Roll No. 112101025 and 122101011**), carried out in the Department of Computer Science and Engineering, Indian Institute of Technology Palakkad under my supervision and that it has not been submitted elsewhere for a degree.*

Dr. Vivek Chaturvedi

Assistant Professor,

Department of Computer Science & Engineering,

Indian Institute of Technology Palakkad.

Contents

Chapter 1

Introduction

1.1 Objective of the project

The aim of this project is to work on the healthy and secure communication between the IoT devices and Cloud and designing an application for giving instructions to the IoT devices. For this purpose a research on MQTT and for secure communication, a research on the most suitable lightweight block cipher, Twine is included in the report.

1.2 Introduction to IoT devices and Software Stack

The number of IoT devices is increasing exponentially throughout the years, there are about 30 billion IoT devices that will be connected by 2030 and around 80ZB of data is being collected, and transferred to the cloud. An IoT network is a large network of IoT devices which are connected and share data among themselves and based on that data, it takes the appropriate actions. The components of the IoT network are Things or Devices (These are basically the sensors or actuators which collect data from the environment.) Gateway (It acts as a level of security for the network and data transmitted and also performs pre-processing of data.), Cloud (It stores the collected data and allows internet connectivity), Analytics (applying algorithms to the data), and User Interface (can monitor or control

data). A software stack is a collection of independent components that work together to support the execution of an application. The components, which may include an operating system, architectural layers, protocols, runtime environments, databases and function calls, are stacked one on top of eachother in a hierarchy. The communication layer and the software application layer are the major focus of this report. The communication between the IoT devices and the cloud is secured by MQTT protocol and the encryption by twine.

1.3 Software Design

The software for giving instructions to the IoT devices is designed using one of the python libraries, Tkinter. Tkinter is the most preferred library for creation of GUIs as it provides a variety of methods for better geometric management, provided it is easier to use. As the software is intended to connect to IoT devices via wireless connections like bluetooth, wifi, Zigbee and Zwave which have libraries in python which can be used to implement them in the software. The software consists of a registration page and log in page. When a new user registers to the application, the data is saved in file type on the operating system we are working on. The log in page looks for this data when a user tries to log in and if the data matches, the user enters the Dashboard.

Chapter 2

Communication layer

2.1 MQTT

This is a messaging protocol which involves three components: subscribers, the broker and publishers. The broker is the middleman for the subscribers and the publishers. An IoT node can publish its data under a topic-name, if some other node e.g. cloud wants to access this data, it has to subscribe to the same topic-name. MQTT has two types of encryption methods, payload encryption and SSL encryption. As this provides an independence between the nodes i.e. if one device stops working, it won't affect the others. The broker alerts all the devices if a new device is connected to the network or if it's disconnected. It also has birth, will protocols to take care of the edge cases of a device being unexpectedly disconnected. This also increases security of the system as if the hacker hacks one of the devices, he don't get access to the other devices, this is not the case in a normal system.

2.2 Encryption

According to the proposed IoT network structure, the encryption between the devices to gateway and gateway to cloud is needed to be secured. The connection from Raspberry Pi to the cloud already uses AES encryption, which is known to be extremely secure. The

crux of the problem lies in the connection from the IoT device to the gateway device, as the IoT device is assumed to have very less computation power, hence, already known-cryptographic algorithms like AES with large keys, also to encrypt/decrypt a file using the AES algorithm, the file must undergo a set of complex computational steps. Therefore a software implementation of AES algorithm would be slow and consume a large amount of time to complete. The immense increase of both stored and transferred data in recent years has made this problem even more daunting when the need to encrypt/decrypt such data arises. After doing some research in the lightweight cipher domain, we found Twine which can be successfully implemented on a device with 1.5KB of ROM. Twine is slower than AES in execution but due to its scalability on IoT devices, we are inclined to use this for the encryption of the data being sent from the IoT nodes to the gateway device.

2.3 AES vs Twine

Comparison between two block ciphers, Twine and AES was done, it was found that in AES, Mix Column Transformation is the most expensive operation where the input matrix is multiplied with an MDS (Maximum Distance Separable) Matrix. This transformation plays an important role with respect to the wide trail strategy of the cipher and guarantees a high number of active S-boxes over a round. Twine uses less resources in terms of memory, size and processing speed compared to AES, it requires less hardware and has no bit permutation or uses of Galois field which requires a lot of heavy computations. AES and Twine were implemented in python to encrypt the text using the inbuilt libraries in python for AES and twine (AES and xtwine), and it was found that Twine takes less time than AES, this is because the processor in this case can easily execute AES so executing Twine which involves less computation is easier, but in the case of IoT devices, due to the restraints on the processing power, AES can't be implemented on IoT but Twine can, the tradeoff is time for better encryption and processing speed.

Chapter 3

Conclusion and Future Work

3.1 Future Scope of the project

Actual implementation of the ideas involving a small network of IoT devices to test the network and all the mentioned protocols, completion of the software and using it for connecting the IoT devices via bluetooth, wifi, Z Wave and Zigbee. Making the software capable of fetching the data from the IoT node in a secure way as mentioned in the report, sending that data to the cloud and getting the relevant action based on the AI algorithms which will be executed on the cloud. We will be implementing our research on Raspberry Pi, The feature (buttons) for connecting and disconnecting bluetooth of IoT devices with the Raspberry Pi bluetooth will be added, this can be done using the bluetooth library in python on raspberry pi and providing the bluetooth address in the code(this address can be fetched using Raspberry Pi itself. Twine is vulnerable to attacks like differential attack, developing the algorithm in such a way that it becomes hard to hack, and can be reliably used. We have come up with a time based encryption algorithm which is a stream cipher having a 64 bit key which will be constantly changing according to a key-sized counter which will be installed with the hardware, we want to implement it and compare its performance with the other prominent algorithms and find the attacks for what it is prone to.

3.2 Conclusion

The overall conclusion of this semester's work can be broken down into four parts: the devices communication layer facilitated by protocol, MQTT, research on lightweight cryptographic (LWC) algorithms and the comparison between AES and Twine, and the time based encryption algorithm. Research was done on IoT devices and IoT networks on making the data transfer along the network secure. As IoT devices are constrained on processing the heavy cryptographic algorithms, research on LWC algorithm was done, later, after going through many algorithms, Twine was finalized and then a comparison between Twine and AES was done. A time based algorithm idea was developed which uses variable keys depending on the key-sized counter of the device. Research on various new areas like using specific hardware components only for encryption, possible attacks and disaster management protocols in the case of a breach was done.

References

https://mqtt.org/

https://www.hivemq.com/mqtt-essentials/

https://www.techtarget.com/iotagenda/definition/Internet-of-Things-IoT

https://en.wikipedia.org/wiki/Internet$_of_things list = PLu0W_9lII9ajLcqRcj4PoEihkukF_OTzA$

$https://www.nec.com/en/global/rd/tg/code/symenc/pdf/twine_LC11.pdf$

$https://www.simplilearn.com/tutorials/cryptography-tutorial/aes-encryption$

$https://www.techtarget.com/searchsecurity/definition/Advanced-Encryption-Standard$

$https://ieeexplore.ieee.org/document/6304791$

$https://www.ijert.org/research/area-optimized-architecture-for-aes-mix-column-operation-IJERTV4IS090602.pdf$