IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 5, May 2017

A Survey: IoT based Vehicle Tracking System

Erabathini Chetan Laxmirajam[1], Dr. R. S. Kawitkar[2]

M.E. Student, Dept. of Electronics, Sinhgad College of Engineering, Pune, India[1]

Professor, Dept. of Electronics, Sinhgad College of Engineering, Pune, India[2]

Abstract: Present scenario of the vehicle tracking system is useful in many applications like trace the vehicle; find the current location of the vehicle, security of personal vehicle, public transportation system, traffic management and others. Focusing on current traffic scenario in major cities and user requirements, we are trying to implement such system which is useful in real time applications. This paper focuses on some of the techniques to help traffic management. One of the techniques is connecting traffic management with Internet of Things (IoTs). IoT is the system of physical objects or things installed with hardware, software, sensors, and system connectivity, which empowers these objects to gather and trade information. IoT uses different types of protocols to work with different objects. CoAP, MQTT protocols ESP8266 and Zigbee models and customized RFID to help traffic management.

Keywords: Custom RFID, CoAP, MQTT, AWS, mosquito paho.

I. INTRODUCTION

In today's technical era the vehicle tracking system becomes more popular in our day to day life due to its several advantages such as security, current location status, navigation. Presently all the vehicle tracking system which we are using now-a-days is based on GPS and GSM modules [1]. This tracking systems becomes more popular with in short period of time due to its advantages such as user friendly, cost effective, and trust worthy, but there are some limitations such as user have to continuously access his mobile data and in case of GSM some time network problems get arise. In this paper we are trying to discuss about tracking system which is mainly based on Internet of Things (IoT). Hera we are implanting system without using GPS and GSM technique. We are going to make simple and very effective traffic management system using the custom protocol having a wide range RFID which we will implement in the buses and differences with talk to each other.

In this paper we will be implementing transmitter and receiver's module which are install in vehicles. So that when was passed from each other they will explain its on state to other vehicle and hence the complete data of each vehicle will be updated in the server. When we come across in this situation we have to think about the speed and effectiveness of the solution.

In current scenario, there are different technologies such as Wi-Fi, Bluetooth, Zigbee but these protocols have its own disabilities so we cannot use them. In turn we are going to use ESP8266 module.

This paper divided into six main sections: section II addresses the CoAP in IoT, section III gives the details of the MQTT, The comparison of MQTT and CoAP, in the section IV, section V explain the hardware requirement , and section VI concludes and discussion of the survey.

II. COAP IN IOT

The constrained application protocol is an application layer protocol which is designed by Internet Engineering Task Force (IETF) for the purpose of to target constrained recourse devices. It possesses subset of HTTP methods which makes it interoperable with HTTP. CoAP is a lightweight protocol, it works on UDP platform. It utilizes the HTTP commands GET, POST, PUT, and DELETE to give resource-oriented interactions in a customer server architecture. CoAP is a request/response protocol that uses both synchronous and asynchronous reactions. The purpose behind planning a UDP-based application layer convention to deal with the assets is to eliminate the TCP overhead and reduce data transfer capacity necessities [4]. Additionally, CoAP supports unicast as well as multicast, as opposed to TCP, which is by its nature not multicast-oriented. Running on the problematic UDP, CoAP coordinated its own systems for accomplishing reliability. Two bits in the header of every packet express the sort of message and the required Quality of Service (QoS) level. There are 4 message types:

1. Conformable: A request for message that requires an ACK. The reaction can be sent either synchronously or in the event that it needs more computational time, it can be sent asynchronously with a different message.
2. Non-Conformable: A message that does not need to be acknowledged.
3. Acknowledgement: assurance about the reception of a conformable message
4. Reset: assures the gathering of a message that couldn't be handled.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 5, May 2017

III. MQTT

MQTT is a machine-to-machine (M2M)/"IoT" connectivity protocol. It was designed as very light weights publish/subscribe messaging transport. It is valuable for associations with remote areas where a little code footprint is required and additionally organise data transfer capacity is at a premium.

The protocol utilises a publish/subscribe design as an alternative to HTTP with its demand/response worldview. Publish/Subscribe is occasion driven and begins messages to be pushed to customers. The central correspondence point is the MQTT agent, it is accountable for dispatching all messages between the senders and the authorised clients Every customer that distributes a message to the agent, includes a topic into the message. The topic is the directing data for the agent. Every customer that needs to get messages subscribes to a specific topic and the agent conveys all messages with the coordinating topic to the customer. so that the customers don't need to know each other, they just convey over the topic. This architecture begins highly versatile solutions without conditions between the information makers and the information consumers.

Message Queue Telemetry Transport (MQTT) was released by IBM and targets lightweightM2M communications. It is an asynchronous publish/subscribe protocol that runs on top of the TCP stack. Publish/subscribe protocols meet better the IoT requirements than request/response since clients do not have to request updates thus, the network bandwidth is decreasing and the need for using computational resources is dropping.

In MQTT there is a agent (server) that contains topics. Every customer can be a publisher that sends data to the agent at a particular topic or/and an subscriber that gets programmed messages each time there is new update in a topic he is subscribed.

MQTT ensures dependability by giving the decision of three QoS levels:
1. Fire and forget: Once a message is delivered and ACK is not necessary.
2. Delivered at least once: Minimum once a message is sent and an ACK is needed.
3. Delivered exactly once: A four-way handshake mechanism is utilized to guarantee the message is delivered exactly one time.

Basically MQTT keeps running on TCP, it is intended to have low overhead contrasted with other TCP-based application layer protocols [10]. In addition, the publish/subscribe design that it utilized, is more appropriate for the IoT than request/reaction of CoAP, for instance, since messages do should be reacted. This implies bring down system data transmission and less message preparing that really broadens the lifetime of battery-run devices.

To guarantee security, MQTT agents may require username/secret key confirmation which is dealt with by TLS/SSL (Secure Sockets Layer), i.e., the same security protocols that ensure privacy for HTTP transactions all over the Internet. By comparing MQTT with the aforementioned CoAP, it is possible to see that the UDP-based CoAP has lower overhead than the TCP-based MQTT. However, due to the lack of TCPs retransmission mechanisms, packet loss is more likely to happen when using CoAP. According to a recent research study [10], MQTT experiences lower delays that CoAP for low packet losses, but CoAP generates less extra traffic for ensuring reliability.

However, results can vary depending on the network conditions. Additionally packet loss and delays depend on the QoS of the messages. In both protocols, packet loss degrades and delays increase when the QoS level is higher.

IV. COMPARISION OF MQTT AND COAP

	MQTT	**CoAP**
Architecture	Light weight messaging protocol uses publish/subscribe architecture	REST client approach uses request/response architecture
Bandwidth	Less bandwidth consumption hence suitable for bandwidth constraint environments.	Uses more Bandwidth as compared to MQTT
Power	Consumes 5 times less power as compared to HTTP	Consumes less power as compared to HTTP
Security	SSL/TLS provides encryption facilities.	SSL/TLS not able to provide security as built on UDP.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 5, May 2017

V. HARDWARE REQUIREMENT

A. ANTENNA DESIGN

Micro strip line is a channel for electromagnetic waves, which consists of three layers. The layers are conducting strip, lossless dielectric substrate and infinite ground plane as shown in Figure 1. The effective permittivity and characteristic impedance of micro strip line are determined by physical parameters such as width of conducting strip (W), height of substrate (h) and relative permittivity of substrate (E_r).
The discussion starts with the micro strip line design:

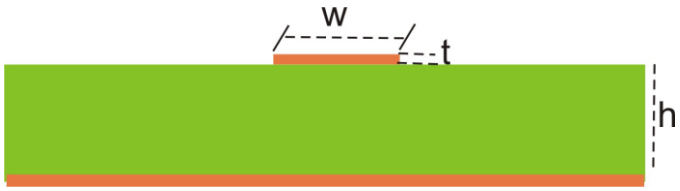

Fig 1. Substrate of the micro strip antenna

$$\mathbf{E_{reff} = [(E_r + 1) / 2] + [(E_r - 1)/2] [1 + 12(h / w)]^{-1/2}}$$

Where,

E_{reff} = Effective dielectric constant
E_r = Dielectric constant of the substrate
h = Height of the Dielectric Subtrate
w = Width of the patch

Width(w):

$$\mathbf{w = (C_0 / 2fr) * [2 / (Er+1)]^{1/2}}$$

Length (L):

$$\mathbf{\Delta L = 0.42\ h\ * \frac{[Ereff+03]\ [(\frac{w}{h})+0.264]}{[Ereff-0.258][(\frac{w}{h})+0.8]}}$$

Now,

$$\mathbf{L = [C_0 / 2fr\ (E_{reff})^{1/2}] - 2\Delta L}$$

By using this values we should make a chip less micro strip antenna.

B. ESP8266 MODULE

ESP8266 is a low cost Wi-Fi chip with full TCP/IP stack and micro controller unit(MCU), it is an arrangement of elite, high coordination wireless network, intended for space and power constrained mobile based planner. It gives unparalleled capacity to install Wi-Fi ability within other framework or to work as a standard application with most reduced cost and negligible space required. Here ESP 8266p and Custom RF-ID is associate with AWS server and MYSQL information database [7].
Node MCU is an open source IOT platform. It includes firmware which runs on the ESO8266 Wi-Fi SoC from hardware which is based on the ESO-12 module. The term "Node MCU" by default refers to the firmware rather than the dev kits.

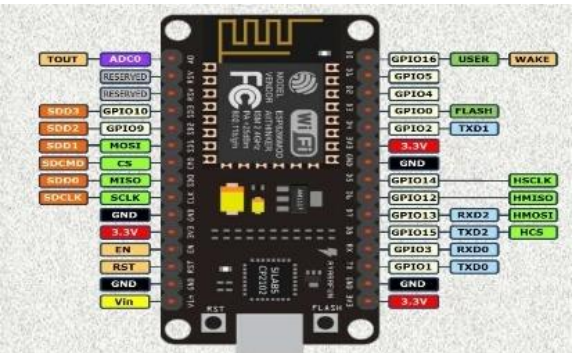

Fig2. ESP 8266 Module

ESP8266 is work on the base of IEEE 802.11b/g/n with integrated TCP/IP protocol stack and Wi-Fi 2.4 GHz support WPA/WPA2 [7].

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 5, May 2017

Some important parameter as given below:

Categories	Items	Values
Wi-Fi Parameters	Certificates	FCC/CE/TELEC/SRRC
	Wi-Fi Protocol	802.11 b/g/n
	Frequency Range	2.4GHz - 2.5GHz
	Tx Power	802.11b:+20dBm
		802.11g:+17dBm
		802.11n:+14dBm
	Rx Power	802.11b:-91dbm
		802.11g:-15dbm
		802.11n:-72dbm
	Type of Antenna	PCB Trace, External, IPEX Connector, Ceramic Chip
Hardware Parameters	Peripheral Bus	UART/ IR Remote control
		GPIO/PWM
	Operating Voltage	3.0 -3.6 V
	Package Size	5 x 5 mm
	External Interface	N/A
Software Parameters	Wi-Fi mode	Station/ softAP/ softAP+staion
	Security	WPA/WPA2
	Firmware Upgrade	UART Download / OTA
	Software Development	Supports cloud server development / SDK for custom firmware development
	Network Protocols	IPv4, TCP/UDP/HTTP/FTP
	User Configuration	AT Instruction set, cloud server, Android/ iOS App

C. AWS SERVER

-- Amazon web services (AWS) provides on demand computing resources and services in the cloud, with pay as you go pricing. For example AWS server can run by log on to, configure, secure and launch EC2.

Amazon elastic compute cloud(EC2) provides scalable computing capacity in the AWS cloud. So its provides easy and faster communication. Virtual computing environment known as instances.

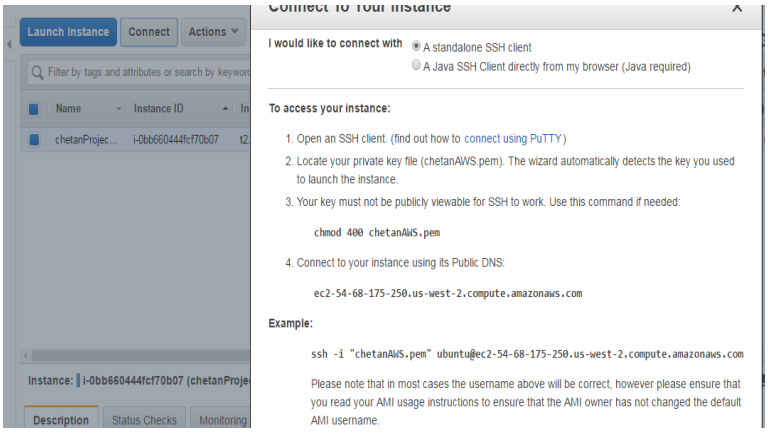

Fig 3. AWS .pem File

To launch the server first log on to the AWS console -> select EC2 then launch instant after that connect the server , then it provide .pem file to start server as shown in the above figure.

D. ECLIPSE PAHO

Eclipse Paho is an MQTT broker, its main responsibility is to provide a communication channel between publishers and subscribers. If any publisher, using the Eclipse Paho MQTT client API can publish the messages to an MQTT Broker. It is the responsibility of the broker to deliver all the messages arriving on a topic to all interested clients.

The eclipse paho provides open source user implementations of MQTT messaging protocol aimed at new, existing and emerging applications for the Internet of Things.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 5, May 2017

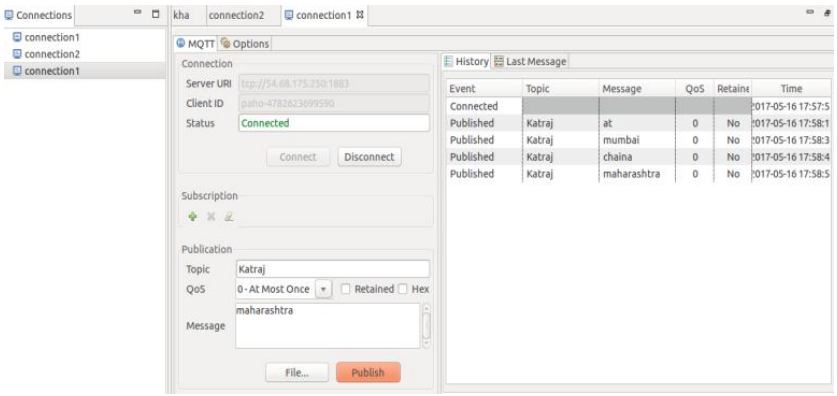

Fig4. Paho testing of publish and subscribe

VI. CONCLUSION AND DISCUSSION

In this paper, we discussed about the protocols which we are using in IOTs i.e. CoAP and MQTT. The details about CoAP and MQTT is elaborated, also comparison between CoAP and MQTT is discussed here. The hardware which we are using in our project is explained in detail and also discussed about the server which we are using for our tracking system.

REFERNCES

1. Pham Hoang Dat, Micheal Drieberg and Hguyen Chi Cuoung (2015) IEEE Conference On open System "Development of Vehicle Travcking System Using GPS and GSM Modem" Pages: 89 – 94.
2. Rohit Dhall, Vijender Solanki (2016) Internanational Journal of Interactive Multimedia and Artificial Intelligence "AN IoT Based Predictive Connected Car Maintaintenance Approach" Pages: 16 – 23.
3. B.Sobhan Babu, T. Ramanjaneyulu, K. Srikanth, D. Hema Sindhu (2016) International Journal of Emerging Trends and technology in computer science(IJETTCS) "Smart Vehicle Management through IoT" Pages: 26 – 31..
4. Yuang Chen, Thomas Kunz (2016) 2016 International Conference on Selected Topics in mobile and wireless Netwotks (WoWNeT) "Performance Evaluation Of the IoT Protocols under a Constrained Wireless Access Network"
5. VasileiosKaragiannis, PeriklisChatzimisios, Francisco Vazquez-Gallego, Jesus Alonso-Zarate (2015)Transaction on Internet of Things(IoT) and Cloud Computing "A Survey on Application Layer Protocolsfor the Internet of Things" Pages: 1 – 9.
6. SharvariRautmare, Dr. D. M. Bhalerao (2016) IEEE International Conference on Advances in Computer Applications (ICACA) "MySQL and NoSQL database comparison for IoT application" Pages: 235 – 238.
7. Manan Mehta (2015) International Journal of Electronics and communication Engineering and Technology(IJECET) " ESP 8266 A breakthrough in wireless sensor network and internet of things" Pages: 7- 11.
8. Houdawerfelli, KhaoulaTayari, MondherChaoui, MongiLahiani, HamadiGhariani (2016) International conference on Advanced Technologies for signal and Image Processing(ATSIP) " Design of Rectangular Microstrip Patch Antenna" Pages: 798- 803