IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-6, Issue-7, July 2018 E-ISSN: 2347-2693

Protocol Analysis in Internet of Things for deployment strategies

Sumanashree Y.S[1]*, Suresha[2]

[1]*Department of studies in Computer Science, Manasagangothri,University Of Mysore, Mysuru
[2] Department of studies in Computer Science, Manasagangothri,University Of Mysore, Mysuru

Corresponding author: sumanashreeys@gmail.com

Available online at: www.ijcseonline.org

Accepted: 19/July/2018, Published: 31/July/2018

Abstract- With the inventions and improvement in the technology, billions of devices continuously produce or generate data. On this basis, we have designed the technique, through which data was extracted or collected from different sources. The data from the respective source was standardized and utilized to respective situations to act upon. The proposed system uses Hyper Text Transfer Protocol (HTTP) and Message Queue Telemetry Transport (MQTT) as base protocols for communication across network. The performance of both the protocols were compared for respective action and situation across varied load conditions. Thus, based on the output, the protocol would selected for different cases.

Keyword s- *HTTP, Internet of Things (IOT), MQTT.*

I. INTRODUCTION

Different sensors like temperature sensors, humidity sensors, radiation sensors, pressure sensors, produce very large amount of data within a short time. These data are voluminous and difficult to organise them in a useful way. There are many platforms like Xively and Thing speak which focus on different features on data to organise them in a proper way [1].

The concept of IOT is to connect things to internet and communicate. IOT leads to big data and transfer that large voluminous data to information and react through actuators [2]. For example, in smart city different sensors are used [3]. In a traffic control different sensors which give indication that more vehicles are ahead, take turn or go slow [4]. Connected devices also generate massive amount of Internet traffic, including loads of data that can be used to make the devices useful, but can also be mined for other purposes. All the new data, and the Internet accessible nature of the devices raise both privacy and security concern.

It has been forecast that around 50 billion devices will be connected smart through IOT by 2020 in a City, Home, College Campus and in an industry [5]. It is difficult to collect data and monitor them manually, so it is necessary to design an IOT platform, which handles huge amount of data, analyses and easily frames the decisions automatically based on the analytical system. Sensors in city infrastructure can help reduce road congestion and warn us when infrastructure is in danger of crumbling. Gadgets, out in the open can monitor for changing environmental conditions and warn us of impending disasters [6]. Sensors can sense the Gas leakages around the industrial areas and situations can be handled with alerts [7].

It is necessary to design a model which will handle billions of devices, visually give an overview of the data and their resources as well as the ways data is in which analysed.

Further the rest of the paper is organized as follows, Section I contains the introduction, Section II contains the related work, Section III contains the architecture, Section IV explain implementation, Section V describes results and discussion, Section VI concludes research work with future directions.

II. RELATED WORK

The important IOT platforms provide the internet connectivity for devices (things) and ability to analyze the data generated from these devices [1].

In the Literature Survey, architectures for integrating sensors with cloud and IOT platforms have been discussed [8-10]. In this research, we are collecting heterogeneous data and analyze it, using appropriate protocols with the Internet Of Things (IOT). IOT platform allows analyzing heterogeneous data and give the results.

The large data will be communicated using MQTT [11], but here in this paper we use both HTTP and MQTT protocols to analyse and to extract meaningful information within a given

framework. Here, we advice what and when to use HTTP and MQTT protocols.

III. PROPOSED SYSTEM ARCHITECTURE

The design is partitioned as layers as in the Figure 1, Each layer has its input and processed output as well as input for the above layers respectively.

The layers are namely 1. Resource Layer, 2. Communication Layer, 3. Message Layer, 4. Real Time analysis Layer, 5. Storage Layer, and 6. Perception Layer, They are further described briefly as below:

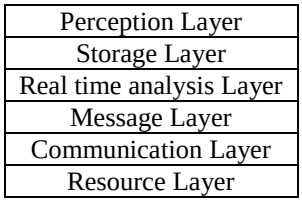

Figure 1. Overview of Proposed IOT Platform

3.1 Resource Layer

This layer is a combinatory of sensors, actuators, web scrapping agents and other host of data pool possibilities. The data generated by these systems either actively or passively are studied and to some extent are standardised at the root level for optimisation the state machines at upper layers and also for conserving the bandwidth constraints as applicable on the host network. Hardware implementation includes various digital/ analog sensors, acting as smart sensors which have a pre-processing for its nonstandard data and actuator with state indication. Software implementation using stub, dummy data was created and used to simulate the performance for real world like situation.

Figure 2, represents the architecture of the resource layer.

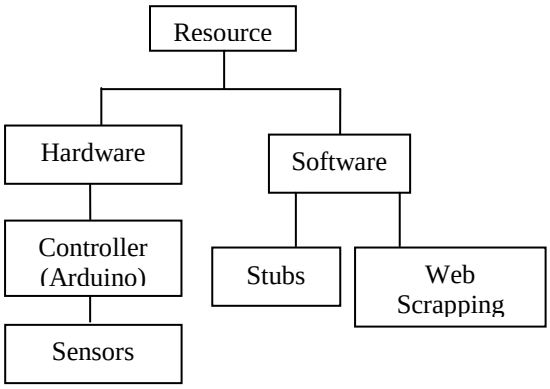

Figure 2. Different resources of data

3.2 Communication Layer

This layer is responsible for communication of devices across the Internet Protocol (IP) network including M2M. The IOT Gateway is used to combine varied protocols devices IP or Non-IP based to IP network. Gateways abstract and encapsulate the sensor platform, aggregate the data from various resources/devices and forward the data to the destination end point. Thus, Gateways create a pipeline with n to 1 mapping.

The two dominant data exchange protocol architectures are Broker based and Bus based.

Broker based architecture: In this architecture, Broker controls the distribution of information. It stores, forwards, filters and prioritizes public requests from the publisher client to the subscriber clients. Clients switch between publisher role and subscriber roles depending on the desired functionalities. Examples of broker based protocols are AMQP-Advanced Message Queuing Protocol, CoAP-Constrained Applications Protocol. MQTT-Message Queue Telemetry Transport, JMS: Java Message Service API.

Bus based architecture: In this architecture, clients publish messages for a specific topic which are directly delivered to the subscribers of that topic. There will not be any centralized broker or any broker based services, Examples of bus based protocol are DDS: Data Distribution Service, REST: Representational State Transfer, XMPP: Extensible Messaging Presence Protocol.

As more and more devices being connected to the Internet and millions of devices interacting with each other and with the server, the need for communication protocol is critical. Based on above communication model different protocols are used. Here, we use and compare HTTP, and MQTT communication protocols.

3.2.1 Hypertext Transfer Protocol (HTTP)

The Hypertext Transfer Protocol (HTTP) is an application-level protocol for distributed, collaborative, hypermedia information systems. It is a generic and stateless request/response protocol based on the client/server based architecture, in which clients request services from the server and the server provides respective response. The basic features that make HTTP a simple but powerful protocol are-HTTP is connectionless, HTTP is media independent. It uses normal IP header for routing of packets and data are not encrypted before transmission.

3.2.2 Message Queue Telemetry Transport (MQTT)

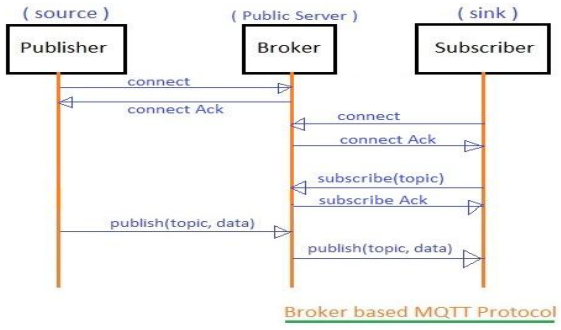

Figure 3. MQTT Protocol Architecture

Figure 3 shows MQTT Protocol Architecture, which is of the form publish and subscribe [12]. MQTT protocol is used at session layer in IOT protocol stack. In order to utilize IOT paradigm, interconnected devices need to communicate using lightweight protocols which do not need extensive use of CPU resources. The features of MQTT protocol are, it is over TCP, uses SSL/TLS for security, it encrypts payload and username/password is used in 'connect' message. MQTT protocol uses many messages such as CONNECT, PUBLISH, SUBSCRIBE, DISCONNECT etc. It uses 1883 port. The three important actors are Publisher (MQTT Client), Subscriber (MQTT Client), and Message Broker.

3.3 Message Layer

It is one of the most significant highly dependable layers. Since every system generates data from different resources, which has to be analyzed, summarised and collected at the central data collection node and used for various custom application functionality based on situations. Java Script Object Notation (JSON) is a perfect form to store things data, since such data can be diverse i.e. different devices produce dissimilar data, data structures will change over the time as different generations of device will give different data sets and certainly up-gradation of all devices including the old ones isn't justifiable due to the applicability of limitations of hardware or software. JSON is so flexible that different devices/versions data can be accommodated, standardized based on versioning techniques for custom application use cases such as alerts, maintenance dues, analytics etc. JSON is completely language independent and is basically a string of formatted ASCII characters i.e. a text format which can be easily interpreted, sent to and from the server, and used as a data format by any programming language.

3.4 Real-time analysis

As the world is progressing from summary savvy to real-time computing, it is a need to understand the requirement and design to process billions of data units generated by billions of things installed across the world. Observing the situation, it is the function of real-time analysis system to collect that large amount of data and support parallel processing to process all such collected data. Most prominently, the system should be fault tolerant and should pose low latency to actions.

3.5 Storage Layer

There is a need to store intermediate outcomes from each layer which might contain raw, pre-processed or processed data.It Makes arrangement for the user to analyse and attain the required result using different available tools. Different caching mechanism can be employed for temporary data storage than storing to the main database reducing latency and also increasing the
overall throughput of the system, this can be achieved by reducing the data traffic within the private network. The bulkier transactions are achieved through the separations of the logical blocks across systems or across the internal storage bus. Thus less data is appreciated.

3.6 Perception Layer

The results are visualized through graphs, which gives real perception to how bandwidth is utilised in both HTTP and MQTT.

IV. IMPLEMENTATION

The components used in this work are shown in Figure 4.

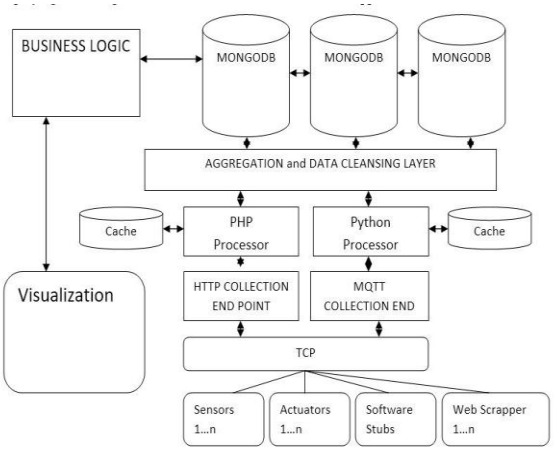

Figure 4. The components of the proposed system

As we know, we are dealing with different data sources generating dissimilar data either via implementation using actual hardware or by software simulation. Analysing the situation, first step is to understand the data source outcomes and to standardize them. Then it is decided as to what is the method to place the communication across the network, and how do we process multiprotocol aware situations and store the data in the database. Considering the above planning, in terms of communication, HTTP and MQTT communication

protocols are most suited for the communication layer. For example, consider a situation like a chat application which has more data as text and occasionally multimedia. Whenever the user tries to send a text message and if we consider an HTTP request as a use case, every text would be needing encapsulation of HTTP header and passed across the network. We know deducting all multimedia features, header size of HTTP would be around 30 bytes on average. Now considering MQTT in the same situation, since this protocol keeps connection alive to the end point, every message sent would contain an header size of approximately 2 bytes per packet, which is concluded with the above context, is 93.33% decrease in bandwidth consumption. On other hand, if we consider multimedia, HTTP has better capability due to the advanced headers which can ease the decoding issues in text based environment but further increase the header size depending on the payload type. Considering the above discussion, we come to a conclusion that HTTP is well suited for one-time operation in a larger interval and MQTT is suitable for continuous data transmission excluding multimedia.

The HTTP data processing unit is built using PHP Technology which utilizes HTTP requests such as GET and POST depending on confidentiality of the data. The whole payload of data has to be communicated through a standard JSON which is converted into PHP data type through authentication using API key for further processing. Using PHP-Mongo connector, the data is segregated, processed and placed at the respective collections as explained in Section 3. If the data belong to any category, then it will be placed at unknown data collection for post processing as required. PHP language is focused for non-textual and from rare data collection devices due to the HTTP protocol.

For Textual based communication which also needs bidirectional socket like communication, we are focussing on MQTT protocol developed by IBM in late 90s. The messaging format is still JSON to standardize and interoperate between HTTP and MQTT on service failures. PyMongo extension in Python along with PAHO Client Library and requests are utilized for implementation.

Storage

After Real time analysis by either of the languages, we visualize the result and take respective action as necessary such as alerts, notification or reactive approaches. All complex analysis is either done by using aggregation framework provided by MongoDB or Pandas in Python with data store as MongoDB. Sharing Technique in MongoDB is utilized for scalable operations.

Implementation units such as sensors and actuators generate disparate data at root level, using Arduino as sub root level processor, we have attached each sensor data unit with globally unique ID and format the data bits into JSON readable structure. We used software stubs acting as sensors to simulate and measure the load taken by respective servers for dynamic load balancing. As a usual method of polling, we use HTTP communication protocol for the sensors communicating rarely or which has complex standard data such as images, video frames etc. At the grass root level, we prefer HTTP for single directional request-response suitable use case.

We generate multiple stubs, to generate data. Each stub has its own unique ID with random standardized pre-processed payload based on JSON, connected with MQTT communication protocol, which has a model of publish-subscribe. Each Stub will publish or subscribe based on the requirement, the middle man, i.e. MQTT broker such as Mosquito, Mosca or Aedes will forward to the designated recipient upon authentication and acceptability of topics.

JSON is language independent. A concept of arrays or objects encapsulated in a string format which support multiple data types such as Boolean, String, Integer and float as against XML, which doesn't preserve primitive data types. JSON as a string can be converted from any to any language, if the same conversion state machine is employed. Thus, JSON can be operated in any language making it most suitable for Internet of things applications where a heterogeneous language resides across different servers leading to easy parsing and translation.

During the implementation using python, we use inbuilt libraries such as JSON to convert from language specific data structure to string and vice versa using json.dumps() and json.loads() respectively, Paho python MQTT Client for MQTT message call-back mechanisms for asynchronous scalable approach.

It is necessary to use data storage that stores large amount of data and low latency and utilize low bandwidth. Here, we have used Mongo dB, NoSQL database which is document oriented, and free from joins and normalization.

Scheme of MongodB

Data in MongodB has a flexible database schema with tables named as collection, where each collection containing varied types of documents formatted in JSON documents. Since there is no strict schema structure other than documentation model, any dynamic data addition is possible at runtime enabling it to be perfect contender for storage medium related to Internet of things.

Suppose, we are designing schema for sensors which has n fields, after configuration we add one more sensor having n+ 1 fields, then MySQL Schema based approach would pose architectural issues due to static nature of table at the base level. But, MongodB is capable of handling this by masking

off the non-existent values without raising errors. The JSON format for sensor data is as shown

Sen={"id":uniq,"type":"sensor","subtype":
"temp","Lmodel": optional","values": random }

NoSQL is data driven and SQL is query driven.

V. RESULTS AND DISCUSSION

The objective is to build an IOT platform with multiple protocols for real-time analysis of heterogeneous data streams. The web portal consists of the Home Page and Device Page, through which user can sign-in, sign-up and open pages to enrol resources to the system respectively.

5.1 HTTP Performance Evaluation :

Algorithm :
HTTP Stub Generator: (Input Q)
Start
Initialize data for demodataset of m KB
Choose Random Sleep Range T (For Network Simulation)
For n in Q Requests
Send HTTP Request
Sleep for T Sec

The above Stub is being threaded for N Instances to simulate the clients pinging the same server at the same time. A Global Count is being tracked for time vs request. One client is continuously sending around 100 requests, such m clients sending request simultaneously to the same server. Number of request which the server able to take with the certain period is the time.

Figure 5.1 show that for 100 request it takes 0.95 unit of seconds and for 16,000 request it takes around 118.46 unit of seconds and on further analysis and interpolation data we get the graph as shown below.

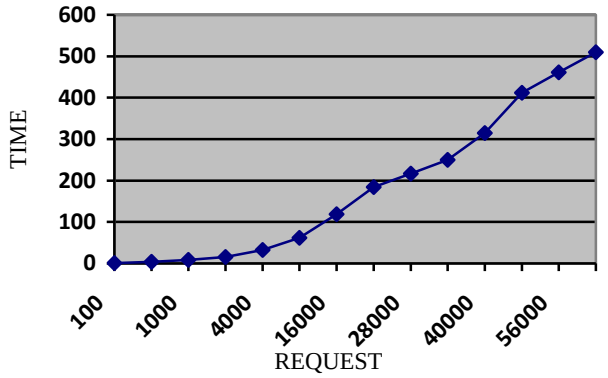

Figure 5.1. HTTP Requests Performance

5.2 MQTT Broker Performance Evaluation

Mosquitto is an open source message broker that implements MQTT protocols. It is suitable for internet of things messaging such as low power sensors and mobile phones, microcontrollers like Arduino.

Algorithm
MQTT Stub Generator: (Input Q)

Start
Initialize data for demodataset of m KB
Choose Random Sleep Range T (For Network Simulation)
For n in Q Requests
Publish to Topic.

Sleep for T Sec

The above Stub is being threaded for N Instances to simulate the clients pinging the same server at the same time. A Global Count is being tracked for time vs request.
Figure 5.2 show the time taken by MQTT to respond requests.

Considering the communication is persistent and stateless, thus expectation of more request processor is higher on experimental, it is seen that 100 requests of unit data evaluated in 0.005 seconds and 10,24,000 requests were evaluated in 65.2 seconds.

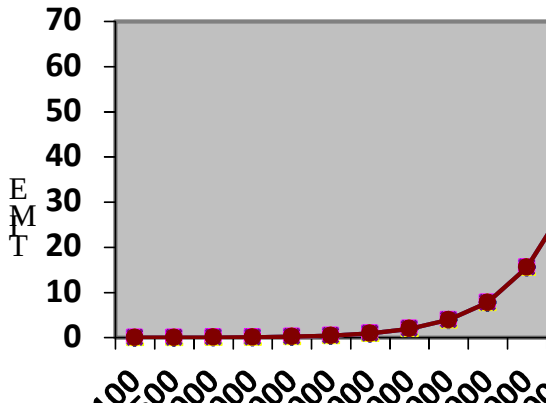

Figure 5.2. MQTT Request Performance

5.3 Comparison of MQTT and HTTP

Figure 5.3 shows that with HTTP 64000 request are evaluated in 509.54 seconds unit of time where as with MQTT it evaluated in 3.34 seconds unit of time. Both incorporate mongodb as a persistent storage.

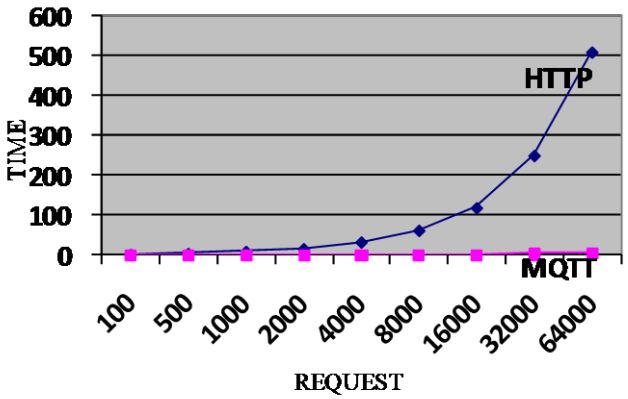

Figure 5.3. Comparison of HTTP and MQTT Performance

In MQTT Time taken is constant for around 64000 units of request, where as in HTTP, increase in time is parabolic with respect to request due to its state levnest and TCP handshaking.

The performance of HTTP and MQTT is dependent on the system configuration. It varies from system to system.
We did our work on i7 processor with CPU @ 2.40GHz,120GB SSD,8GB RAM.

VI. Conclusion and Future Work

After exploring various aspects of Internet of things and data generated by things in varied forms, generalizing into a common format with variable structure was understood as a significant factor for channelizing data units into respective data stores, for meaningful manipulations. JSON and XML are the most popular data interchange formats used throughout. After our analysis on the requirement, certain data formats need to be preserved as at the source, which was possible only using JSON considering its platform independent data structure format along with its capability to preserve primitive datatypes. During the communication phase, considering longevity and frequency of data packet transmission, we considered HTTP and MQTT due to its multimedia and efficient textual data transmission. After Analysis with real-life test as discussed, MQTT is best suitable for connections which are kept for longer time with shorter data burst without multimedia where as HTTP plays a major role in API or longer frequency stateless communication.

As a future work, we are planning to incorporate various statistical data analysis on the data store to calculate mean median and standard deviation of values and to make a supervised learning of calculation of thresholds in case of sensors to longer run for better understanding of deployment strategies. .

Acknowledgment

Authors are grateful to Prof.G. Hemantha kumar, Chairperson, Manasagangothri, Mysuru for constant encouragement.

References

[1] J. Mineraud, O. Mazhelis, X. Su, And S. Tarkoma, A Gap Analysis of Internet-ofThings Platforms. Computer Communications, 89-90, 2016, 5-16. http://dx.doi.org/10.1016/j.comcom.2016.03.015

[2] K. Ashton, That "Internet of Things" Thing: In the Real World Things Matter More than Ideas. RFID Journal.2009 http://www.rfidjournal.com/articles/view?4986

[3] A. Mahizhnan, Smart Cities: The Singapore Case. Cities, 16,1999,13-18.http://lkyspp.nus.edu.sg/wp-content/uploads/2013/04/pa_Arun_Smart-Cities-The-Singapo re-Case_99.pdf http://dx.doi.org/10.1016/S0264-2751(98)00050-X

[4] Umer Ahsen,Abdul bais ,A Review on Big Data analysis and the Internet of Things,IEEE ,13th International conference on Mobile Ad Hoc and sensor Systems(MASS) ,pp 325-330,2016

[5] D. Evans,The Internet of Things: How the Next Evolution of the Internet Is Changing Everything, 2011. http://www.cisco.com/web/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf

[6] J. Arias Fernández, et al., IoT-Framework. Product Report,UppsalaUniversity,2013. http://www.it.uu.se/edu/course/homepage/projektDV/ht13/ProductReport.FINAL.pdf

[7] Anusha Bharati, Ritika Thakur, Kavita Mhatre, Protection of Industrial and Residential areas by Wireless Gas Leakage Detector using IoT and WSN, Isroset-Journal (IJSRCSE),Vol.5, Issue.3, pp.62-67,Jun-2017

[8] D. Bhattacharya, and M. Mitra, Analytics on Big Fast Data Using Real Time Stream Data Processing Architecture. EMC Proven Professional Knowledge Sharing, 2013.

[9] S.K. Dash, J.P. Sahoo, S. Mohapatra, and S.P. Pati, Sensor-Cloud: Assimilation of Wireless Sensor Network and the Cloud. In: Meghanathan, N., Chaki, N. and Nagamalai, D., Eds., Advances in Computer Science and Information Technology. Networks and Communications, S pringer, Berlin, 2012, 455-464. http://dx.doi.org/10.1007/978-3-642-27299-8_48

[10] A. Alamri, W.S. Ansari, M.M. Hassan, M.S. Hossain, A. Alelaiwi, and M.A. Hossain,) A Survey on Sensor-Cloud: Architecture, Applications, and Approaches. International Journal of Distributed Sensor Networks, 9, 2013, Article ID: 917923. http://dx.doi.org/10.1155/2013/917923

[11] A.S. Rozik, M.A. Tolba, El-Dosuky, Design and Implementation of the Sense Egypt Platform for Real-time Analysis of IOT Data Streams.Advances in Internet of Things,2016,6,65-91. http://www.scrip.org/journal/ait

[12] D. Bhattacharya, and M. Mitra, Analytics on Big Fast Data Using Real Time Stream Data Processing Architecture. EMC Proven Professional Knowledge Sharing, 2013.

[13] D.L. Hall, and J. Llinas, An Introduction to Multisensor Data Fusion. Proceedings of the IEEE, 85, 1997, 6-23. http://dx.doi.org/10.1109/5.554205

[14] F. Ganz, P. Barnaghi, and F. Carrez, Information Abstraction for Heterogeneous Real World Internet Data. IEEE Sensors Journal, 13, 2013, 3793-3805. http://dx.doi.org/10.1109/JSEN.2013.2271562

[15] M. Eid, R. Liscano, and A. El Saddik, (A Universal Ontology for Sensor Networks Data. IEEE International Conference on Computational Intelligence for Measurement Systems and Applications, Ostuni,27-29 June 2007, 59-62. http://dx.doi.org/10.1109/cimsa.2007.4362539

[16]P.Manickam,V.MuthuGaneshan,M.Girija ,Comprehenssive approach in Studying the behaviour of Contiki RPL Protocol in Diverse Data Transmission Ranges,International Journal on Scientific Research in Network Security and Communication.

Authors Profile

Mrs Sumanashree Y S pursued Bachelor's, Master's degree in Science and M.Tech in Computer Science from University of Mysore. M.Phil from Periyar University. She is pursuing Ph.D. and working as H.O.D and Assistant Professor in Post-Graduate Department of Computer Science, JSS College Of Arts,Commerce and Science,Mysuru.Her main research work focuses on IOT, Computer Network, Cloud Computing, Big Data Analytics, Data Mining,She has 18 years of teaching experience and 5 years of Research Experience.

Dr.Suresha pursued Bachelor of Science from University of Mysore, Mysore in 1987 and Master of Science from University of Mysore in year 1989. and M.Tech in Computer Science from IIT Kharagpur. M.Phil from DAVV,Indore.He Pursued Phd from IISc,Bengaluru in year 2007.He is currently working as Professor in Department of Studies in Computer Science, Manasagangothri, University of Mysore, Mysuru.

He has published more than 60 research papers in reputed international, national journals and conferences. His main research work focuses on World Wide Web, Computer Network and Security, Cloud Computing,Adhoc network,Big Data Analytics, Data Mining, IoT,mining web users and e-governance. He has 27 years of teaching experience and 15 years of Research Experience. 5 students have completed phd under his guidance and 6 students are prsuing PhD under his guidance.

