Serverless Management of IoT Sensing Systems for Fog Computing Framework

Suvajit Ksarkar, Rajeev Wankar, Satish Srirama, Nagender Kumar

Serverless Management of IoT Sensing Systems for Fog Computing Framework

Suvajit KSarkar*, Rajeev Wankar*, Satish Srirama* and Nagender Kumar *

Abstract— In this paper, we propose a framework for the management of the Internet of Things (IoT) devices in a smart building to model services based on the serverless computing paradigm. The deployment of an IoT compatible serverless paradigm consists of a hierarchical structural design across the edge, fog, and cloud computing layers. The fog/edge nodes collect the data generated from various sensors, process the data in the intermediate nodes, and then forward certain data to a cloud for future analysis. The framework consists of a heterogeneous IoT network. We proposed a data distribution algorithm in the framework to make sure management, maintenance and availability of heterogeneous IoT network in the serverless computing paradigm are effective and efficient. The experiments conducted are validated at the developed fog and edge gateways using API mechanism. The response times for an application doing the computation at fog level and at the cloud level are compared. The experimentation shows that latency is less for the fog model as compared to the data sent to the cloud model.

Keywords—Wireless Sensor Network, Smart Building, Fog Computing, Serverless Computing, Sensing Systems

I. INTRODUCTION

Smart building technology is an emerging technology that has the provision of automating and managing various services in a building. Conducive computing architecture is very much required for the present connected world of things. It allows sensor data exchange with low latency [1], reliable operation, and overcomes the connectivity issues with the cloud. The combination of IoT, Artificial Intelligence, and 5G communication technologies allows the applications to perform effectively with the computing architecture. The "Fog computing" architectural model provides the facilities as mentioned above to perform the computations and storage from the cloud closer to the devices based on the service and data requirements [2].

The IoT application research involves the things/devices to the globally interconnected network [3]. The smart sensing devices in the IoT capture large amounts of data that have to be transmitted, stored, processed, analyzed and act accordingly within a specific period time to provide a real-time outcome [4]. Cloud architectural model for the IoT applications is latency intensive.

The Fog-based solutions move the data processing closer to the network edge, which allows for faster response times and increased energy efficiency [5]. Instead of continually moving data to the cloud for computing operations, which accounts for the energy costs, data can be processed and mined on fog/edge devices closer to the user [6]. For cases involving health monitoring, low latency driven by edge and fog solutions allows for emergency medical help to arrive promptly on time. Due to a large amount of data traditionally sent to cloud services, privacy and security remain a crucial issue, especially in cases where a patient's medical data could be hacked [7].

*The authors are with the School of Computer and Information Sciences, University of Hyderabad, Telangana, India;
+The author is with the University of Tartu, Estonia.

In existing fog deployments, an increased number of fog nodes contribute to lower latency in data transfer [8]. Various fog/edge mining techniques can also contribute to lowering the amount of time spent transferring data to the cloud [9]. Fog/Edge can have lightweight computations on close continuous, for example, the sensor data aggregation and handling of stream data. This innovation is an essential empowering influence for the future improvement of cutting edge administrations, for example, traffic checking and arranging through the mix of road sensors information (e.g., vehicle following, air quality estimations) and meteorological data.

In spite of the fact that fog gateways (hubs) offer an obliged registering limit contrasted with their cloud partners, regardless they have capacities to process information in close ongoing to give confined administration to users, limiting the communication necessities with the cloud, or guaranteeing application strength and avoiding blackouts between the Fog and Cloud layers. With circulated Fog-based computational capacities, applications require a better capacity than adjust to the consistent changes that happen inside the elements of a cutting edge city: the best way to give the required adaptability will be using progressed Artificial Intelligence (AI) procedures that assist frameworks to learn and demonstrate the conduct of data in close continuous.

When using the cloud-only solutions, the data retrieval times are too high for a real-time scenario, such as fall detection or stroke mitigation, that require high response times from medical professionals [10]. Frequently sending information to the cloud for computation accounts for higher power consumption and costs associated, even more so today, when the amount of data generated by sensors is huge [11]. In a typical cloud service, it was proved that high latency and low sustained performance as compared to distributed computing architecture with several computing nodes at different geographical locations is better than cloud-based solutions [12]. Cloud-based solutions also do not offer the user a low-cost mobile environment, which is required for many of the patient monitoring scenarios.

A few advances applicable to the development of the IoT have risen in the most recent years, including 5G communication and fog processing [12]. The blend of these innovations opens another scope of potential applications with regards to Smart Cities. There is a quick development in the number of activities wanting to convey new administrations to citizens, in view of the sending of countless hubs close to the edge, in the lanes of present-day urban communities, overcoming any issues among gadgets and cloud-based administrations. The present research work identifies a needed shift from centralized cloud architectures to distributed fog and edge-based ones that better meet the needs of a smart building system with an excess of data as compared to legacy systems. This work also proposes a device abstraction mechanism in the form of the Application Program Interface (API). It supports

heterogeneity, smooth integration of new sensing devices, and portability of the system to a new environment.

The fog computing model uses distributed computing approaches to establish smart and connected areas. Fog computing ability is to support compute-intensive applications. It enables real-time decision-making; deploy low latency-sensitive operations to perform very effectively. Consideration of a decentralized architecture for the device to connect to the network shall reduce the bandwidth restrictions and create room for the continued addition of more connected devices with the help of fog computing.

The present smart home/buildings are utilizing the IoT theme for better efficiency in energy consumption, improved inhabitant experience, and lower operational expenses. They require distributed fog/edge arrangements since they contain a large number of sensors estimating different structural working parameters [13], such as ambient temperature, humidity, monitoring inhabitants, energy use, smart card readers, parking spot inhabitance, fire, smoke, security, elevators, and air quality. This utilization case shows how fog/edge hubs at the room level, floor level, building level, and cloud level can be progressively architected for effective and efficient real-time processing, empowering many new applications.

II. DEPLOYMENT MODELS FOR IoT APPLICATIONS

This section describes different computation models currently followed by application developers for various IoT tasks:

A. ***The Cloud computing*** model provides access to a shared pool of resources through the internet [14][15]. The resources may be network, storage, and computation [16]. The deployment of IoT applications using the cloud in comparison to traditional data-center is cost-effective.

B. Edge computing for an IoT based application is to carry out the computation at the same network level as that of data generation. Since the processing is done near the source, latency for data transfer is less. An application that requires quick response time uses edge computing [17][18].

C. ***Fog computing*** is an important model in the deployment of distributed applications that are latency aware [19]. Fog cluster consists of fog nodes arranged in hierarchical order in multiple layers. Each layer of a fog cluster can be designed to solve a particular problem. Load balancing among the nodes of a fog cluster can be done [20]. Also, data can be offloaded to the powerful cloud servers for further processing.

In an edge-computing architecture, data operations, such as classification or compression, are completed at the edge of the network. These edge nodes are often small servers that allow for the fast processing of data that mobile devices can often not achieve. Edge or fog nodes can be a multitude of devices, deployed at different distances between the Cloud and edge user device, depending on the operating range. The commercially available products such as Raspberry Pi [21] [22], Arduino [23] [24], and Field-Programmable Gate Array (FPGA) [25] platforms serve as fog/edge gateways. IoT applications are the stimulus to events. An event such as sensed data from the temperature sensor can trigger the HVAC system [26]. Based on the location of data processing [27], we have explained different models that are used currently for deploying IoT applications.

- **IoT + edge model**

Data generated from the IoT sensors/devices are processed in some near edge devices. The edge devices take action based on the processing. The model is limited by the low computation power of the edge devices.

- **IoT+cloud model**

In this model, data from the sensors are directly transmitted to the cloud for processing and analysis. High-performance computing units support the cloud infrastructure. Hence, this model can support heavy applications. The drawback of this model is high latency cost since the cloud nodes are geographically far from the IoT sensors.

- **IoT + fog + cloud model**

Data flow from sensors to fog nodes. Fog nodes are capable of taking any action or can filter data and send it to the cloud for further analysis. Hence, this model has a high-performance computing power of the cloud as well as can respond quickly to an emergency using the low latency fog infrastructure.

D. Software service models offered by Cloud/Fog computing

We can architect the cloud as well as fog infrastructure to provide various services such as Infrastructure as a Service (IaaS), Platform as a Service (PaaS), Software as a Service (SaaS) [28][29]. Many applications use these services for computation, storage, management and control purposes.

IaaS: Using virtualization techniques IaaS provider offers virtual machines to the users on top of the underlying fog/cloud infrastructure [30][31]. Users have complete control over the operating systems, packages to be installed, and networking.

PaaS: Offers an environment for running user tasks. The user can design the application in any programming language. The user also has control over the libraries used. The environment is a container which is a process running on any host physical/virtual machine.

SaaS: It is a software service deployed on a fog or cloud cluster. The user has very little control over the system components.

A few urban areas around the globe are associated with new schemes towards Smart Cities. The frameworks in the cities such as Nice, France, use Connected Boulevard [32] which has been developed to advance several aspects of the city including traffic, road lighting, waste transfer, and environmental conditions. Likewise, in Santander, Spain, the venture SmartSantander [33], centers around the European office for research and experimentation of models, innovations, and applications for brilliant urban communities, yet without concentrating on Fog computation. Further, different urban communities like Songdo (South Korea), Masdar City (Abu Dhabi, UAE), Paredes (Portugal), Manchester (UK), Boston (US), Tianjin (China) and Singapore, declared smart city-related tasks [34]. Even though different approaches portray on every city, a flexible and secure investigation between the

fog/edge and server computations are interesting issues, rotating around reasonable and moderate methods for the framework [34].

Many fog applications depend on the real-time data stream process. Yang et al. [35] present general models and designs for Fog information streaming and examine the basic properties of the most widely recognized applications. An outline about device-to-device correspondence on the fog can likewise be found in the article Bao et al., concentrating on the physical plane of such devices. Similarly, Gang Xu et al. indicated how availability issues are imperative in this field, explicitly in remote correspondences, applying a calculation to organize which of the accessible information in a given field with interconnected sensors is send to a versatile bearer and how to course it when associations between information supplier and the portable transporter are discontinuous and short in time.

The most popular case requiring low latency is elderly monitoring in homes. Rasika et al. [38] and Shalom et al. [39] proposed systems that collect patient's sensors data on current body status and transmit to a Personal Digital Assistant (PDA) or mobile phone, which does local processing and alerts family or emergency services in case of a fall detected or a deviation from healthy heart rate or blood pressure. Communication between a device and fog node is done with short-range communication protocols, such as IEEE 802.15.1 or 802.15.4 [40]. Often a sensor node will be connected to additional computing devices or cloud services using a wireless 802.11protocol [41]. Many applications similar to [42] utilize IEEE 802.15.1 or Bluetooth as a protocol for communication between a medical device and a smartphone, where computation is done at the device level. Once a small amount of computing is finished on the smart device, data is transferred to a doctor or an additional server via mobile communication services such as 4G or 5G. These are popular solutions due to low cost and simple programming. Other research uses a graphics processing unit in cases where pictures are the data input to be computed. Other popular nodes are Telos Mote [43] and Intel Edison [44], especially for cases involving ambient sensing. Telos is a collection of sensing devices developed by UC Berkeley for wireless sensor network (WSN) research that utilizes WPAN/IEEE 802.15.4. Intel Edison, is similar to the Telos mote, except it is compatible with IEEE 802.11 and IEEE 802.15.1.

The basic features of edge/fog computing gateways (hubs):-

1) In the industry, we have wireless communication protocols such as Zwave, Zigbee, WiFi, and Bluetooth. Hence, the edge/fog computing system recognizes the necessity of incorporating wireless compatibility of several communication technologies.
2) The edge/fog gateway needs to collect and process the data locally to reduce latency.
3) Access the data remotely for the distribution of data so that the load balancing mechanism can be incorporated.
4) The edge/fog gateway should have the capability to store the data for local computations.
5) The edge/fog gateway should have the mechanism for data security and be able to provide data on request.
6) Gateways should support services/applications for various IoT protocols as such REST, MQTT and COAP.

III. SYSTEM SETUP

The system setup for the smart building application consists of the hardware components, computing devices, communication modules, and software units.

A. *HARDWARE COMPONENT DETAILS:*

The IoT framework usually consists of sensors based on the specific application. The general hardware devices are Arduino, Raspberry Pi, nodeMCU [45], XBee [46]. These devices are also edge devices but can also be used as fog nodes.

Sensing Systems:

Sensors to monitor room environment like DHT [47] and gas sensors [48] can be an important inclusion in a Smart home application. They have a low cost and easy to maintain. Sensors do not have any computational power. Sensors, along with a computational device, can be abstracted as a sensing system. Sensors can be further categorized into environmental and smoke sensors.

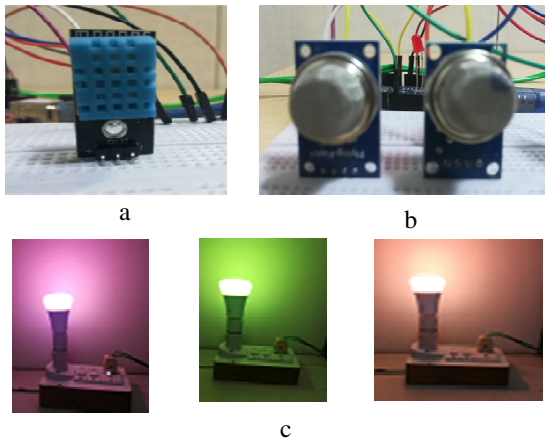


Fig.1 a.DHT11, b. Gas Sensors and c. Philips Hue Lights used in the experimental setup

Environmental Sensors: DHT11 Digital Humidity and Temperature (DHT) sensor outputs calibrated digital signal as temperature and humidity data in the output pin. A DHT11 sensor consists of three pins VCC, ground, and data pin, which is shown in Fig.1(a) The calibrated data can be collected in an Arduino and processed further to get the value of temperature and humidity.

Gas Sensors: These sensors are used to test the quality of air to detect some gases it is designed for. Fig.1 (b) shows various smoke sensors used in our work. MQ2 sensor detects the quantity of carbon monoxide (CO) gas in the air. It requires the power of 5V and gives the output as an analog signal. MQ2 sensor is capable of detecting gases like CH_4, H_2, LPG, smoke, propane, and CO. It has both analog and digital output pin. MQ135 sensor can be used in a smart home or office to detect the proportion of different gases like CO2, Alcohol, NH3, benzene, and smoke.

Philips Hue Lights is a smart light that has a varying range of colors and is used to control the ambiance of a room. The color at any point of time can be changed remotely from other devices using the REST layer provided by openHAB. Fig.1(c) shows Philips hue light used in our experiment with varied colors at different points of time.

B. Computational Devices

The following section describes the computing devices required in the present IoT environment. We have used these devices to act as an edge or fog node.

1) Arduino: Arduino is an open-source platform for creating hardware and software solutions for critical digital problems. The physical board or microcontroller is capable of computation or processing of data it senses from the attached sensors. It is a low cost as well as supports multiple platforms like Windows, Linux, and MAC. The software Arduino IDE helps to write the logic. It is based on the C++ language.

2) Raspberry Pi: Similar to Arduino, Raspberry Pi is also a computational device with more capability. The microcontroller has ARM-based processor, memory, storage cards, a wireless communication module, GPIO pins. It supports Debian based operating system known as Raspbian. Sensor output can be connected with the GPIO pins to get the data at the Raspberry Pi. Raspberry Pi 3 Model B+ was used in our experimentation.

C. Communicational Devices

These devices are used to forward the data to other computational devices for data processing.

1) NodeMCU: Similar to Arduino, NodeMCU is also an open-source project providing open-source solutions for both hardware and software. It supports the Wi-Fi protocol for communication.

2) XBee: These devices are low-cost radio modules designed only to route the data from an XBee connector to an XBee receiver. The XBee connectors also pass the data to the next level for processing. The XBee devices are connected in the form of mesh topology and follow the Zigbee protocol for communication.

D. Open Source Software Tools

The software tools like openHAB and OpenFaaS can be integrated with the IoT devices to have customized management of an application.

i). openHAB: The open Home Automation Bus (openHAB) is a platform for home automation applications. It provides the ability to connect a large number of devices and systems [49]. openHAB communicates electronically with devices in the smart home environment and performs user-defined actions. Using open-HAB control panel or openHAB REST API, various parameters of hue lights like color, brightness, and saturation can be controlled to create a smart home environment. Fig.2 shows an instance of the configuration panel in openHAB, which provides information about various connected things in our experimental setup.

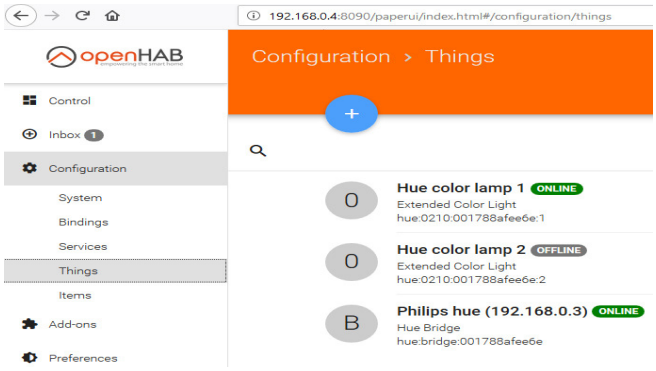


Fig. 2 An instance of an openHAB configuration panel

ii). OpenFaaS: It is an open-source framework used to achieve Function as a Service on the top of a cluster of computational devices [50]. OpenFaaS uses the service of other software tools like docker and docker swarm. It installs a docker engine on top of the underlying operating system. Docker engine is responsible for the creation of containers that is a lightweight process running to provide the desired programming environment. Swarm or Kubernetes Dockers is used to manage docker. Load balancing among the participating nodes is done by calling the services of Prometheus which offers a graphical view of the system workload. The Graphical User Interface (GUI) of openFaaS is shown in Fig.3 and Command Line Interpreter (CLI) to create and deploy micro-services requested by the user of the system. The micro-services become RESTful services with the REST API support of OpenFaaS. Hence, the micro-services can be easily invoked from an application or other micro services.

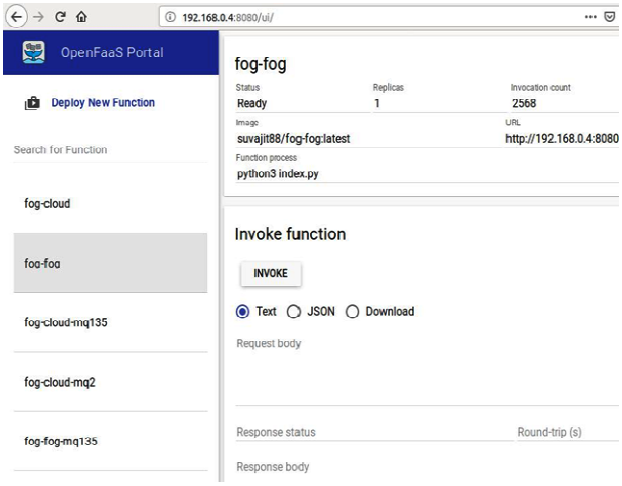


Fig.3 An instance of an OpenFaaS GUI

iii). Software solutions

1) IoT and Cloud Model: This model consists of a device tier, a gateway tier, and a cloud layer. Data from the device tier are pushed to the edge and fog gateways. The corresponding gateway application using the MQTT protocol [51] pushes the data to the public cloud for further processing. The gateway can also use the HTTP protocol to send data to REST API [52] of the public cloud.

2) IoT, Fog and Cloud computing: This model is similar to the previous (Edge computing) model except that the edge gateway pushes the data to some near to the device fog node. The fog nodes filter the data at this layer before sending it to the cloud. Hence, latency can be reduced if this model is followed.

3) Serverless/Distributed deviceless Model: Serverless computing is running user functions on runtime in a container and gets back results. This architecture helps in cost optimization as compared to IAAS where virtual machines have to run continuously to keep the application up. Multiple fog devices can be used to create a distributed system having provision for load balancing. The proposed setup can be configured to run the applications as given in [53-58] for effective quality of network service parameters.

IV. Experimental Setup and Results

Hardware Setup

We have set up the physical layer consisting of edge devices like Raspberry Pi, Arduino, and some sensors like MQ7, MQ2, and DHT11. Our experimental setup also contains XBee devices that follow the Zigbee protocol. It also consists of openhab environment. The sensors are connected to the Arduino pins. The data flows from the Arduino to the Raspberry Pi through the serial cable. The XBee unit is also connected with the Raspberry Pi by cable. The Raspberry Pi is the edge gateway that collects the sensor data. We have a cluster of Raspberry Pi, which is a fog cluster. The fog cluster has been set up using Openfaas. Hence, the microcontroller where Openfaas gateway application is running acts as the fog gateway.

Hardware Abstraction

Device abstraction is the method of hiding the complex working knowledge of devices while reflecting only the functionality it provides. We have designed and developed an API that does the abstraction by creating classes for each device. Multiple devices of the same class can be included in the system in the form of objects having a unique id. In this work, we have created separate classes for environmental and smoke sensors. The objects of the proposed classes are capable of calling the set of functions which are depicted in Fig.4.

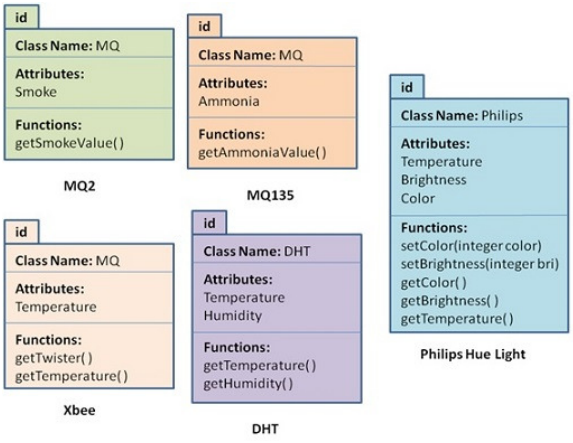


Fig.4 Hardware Device Abstraction

Distributed Deviceless Computation Model

We have proposed a distributed serverless computation model. For that, we have used an open source orchestration platform OpenFaaS to create a fog cluster. We can create functions using the OpenFaaS CLI tool or through the gateway application provided by OpenFaaS. The functions deployed are capable of doing a predefined set of tasks on a container when invoked. The invocation of a function can also be done using the REST API of the function. We can send the sensor data as arguments in the REST based get/post call. Fig.5 shows a fog cluster having multiples of Raspberry Pi units in the cluster model. The function defined uses the infrastructure to run in an appropriate node to provide the service it is intended. There is a provision to offload some computation to the cloud by calling the particular responsible function.

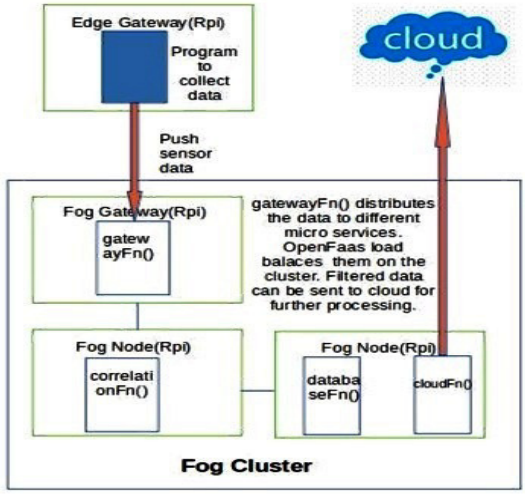


Fig.5 Distributed deviceless Platform

A. Data Aggregation

For data collection, we connected multiple sensors to a raspberry pi which acts as an edge gateway. We created a software service in the form of API to monitor the sensors. The API has abstracted the sensors to reflect only the functionality defined. Algorithm.1 collects the data stream from the input buffer and returns the data.

```
Algorithm 1 collectData(edgeGatewayIP)
Result: Data read or failed to
read while True do
    if check buffer then
        data = read stream; if data
            then Display data;
        else
            Display "failed to read
        data"; end
    else
        wait
    end
    return data
end
```

B. Data Distribution

We have created several micro-services, each one responsible for its own set of tasks. Using the load balancing facility of OpenFaaS we have distributed the services evenly on the fog cluster.

Algorithm2 is used to send data to cloud for further processing.

```
Algorithm 2 sendCloud(data,chId,key)
Result: Http response(200,404,500 etc)
while data do
    requests.post(data,keys)
end
```

storeDataDb() procedure is used to store the data in a database. The micro-service may segregate the data or it can receive the segregated data from the below layer.

```
Algorithm 3 storeDataDb(data,tableId)
Result: Http response(200,404,500 etc)
while data do
    insert data into table tableId
end
```

SYSTEM ARCHITECTURE:

The proposed system architecture is described in Fig.6 An application "edge API" is running on the edge gateway to gather the data from the sensors. The middleware is responsible for the management of the fog cluster and the edge gateway. The fog cluster is created with openFaas. Data from the edge gateway can be pushed to the fog gateway using the REST API provided by openfaas.

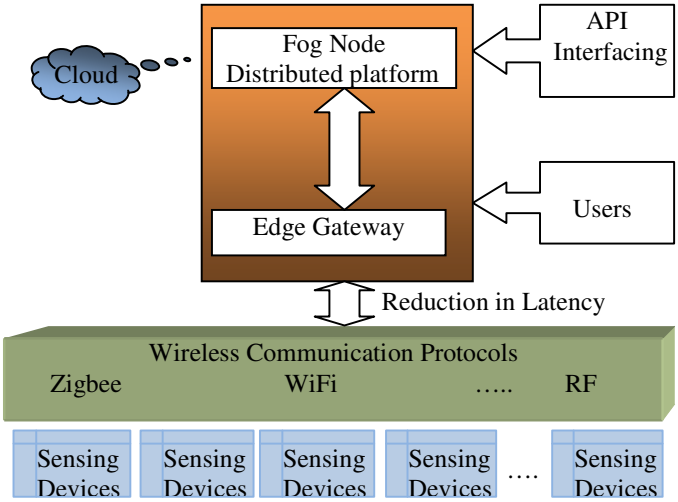


Fig.6 Overall System Architecture

Data at the fog layer can be processed using the underlying fog Infrastructure. We have designed the fog cluster to provide Platform as a Service (PaaS) to the users or application developer. The developer can make decisions whether the processing is to be done at the fog level or send to the cloud for deeper analysis.

A. Edge Gateway Configuration

The edge gateway is a raspberry pi connected with different sensors using different protocols. We are running a data aggregation program to collect the sensor data. Both Raspberry Pi and Arduino are used as edge nodes in our lab setup. We have configured a Raspberry Pi as the gateway. An Arduino is connected to it which collects the data from smoke and DHT sensors. The combination of different sensors with the edge gateway can be found in Fig.7

B. Fog Cluster Configuration

We have created a cluster of four Raspberry Pi devices to act as a fog infrastructure. All the machines are on the same level of the network. We installed OpenFaaS on a cluster of micro-controllers. The configuration details of the fog nodes which are Raspberry Pi in our setup are given in Table-I.

C. ThingSpeak Cloud

ThingSpeak is an IoT based platform for visualizing and analyzing data generated from IoT sensors. We have used the free version of ThingSpeak cloud to upload our sensor data. DHT sensor data uploaded to ThingSpeak channel can be visualized as shown in Fig.8.

Table.I Configuration of the existing computing devices

Parameters	Arduino Uno	Raspberry Pi B+
Processor	ATMega328P	Broadcom BCM2835 SoC based ARM11 76JZF
Operating Voltage	5V	5V
System Memory	2kB	512MB
Flash Memory	32kB	-
EEPROM	1kB	-
Communication Supported	IEEE 802.11 b/g/n, IEEE802.15.4, 433RF, BLE4.0, Ethernet, Serial	IEEE 802.11 b/g/n, IEEE802.15.4, 433RF, BLE4.0, Ethernet, Serial
I/O Connectivity	SPI, 12C, UART, GPIO	SPI, DSI, UART,SDIO, CSI.GPIO

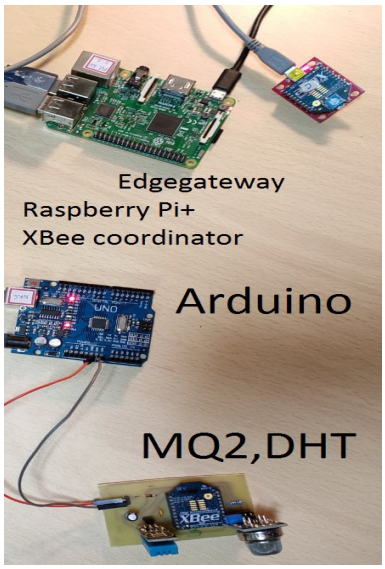


Fig.7 Edge gateway connected with different sensors

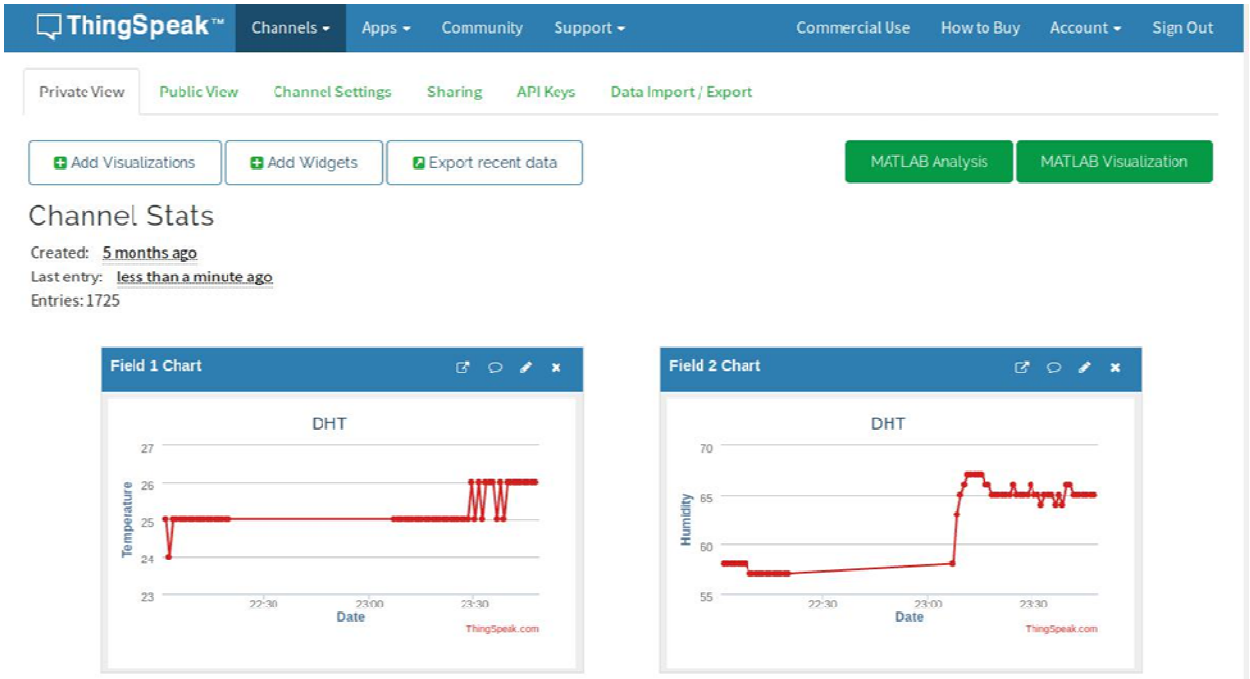


Fig.8 ThingSpeak cloud visualization of sensor data

A. Result of fog + cloud Model

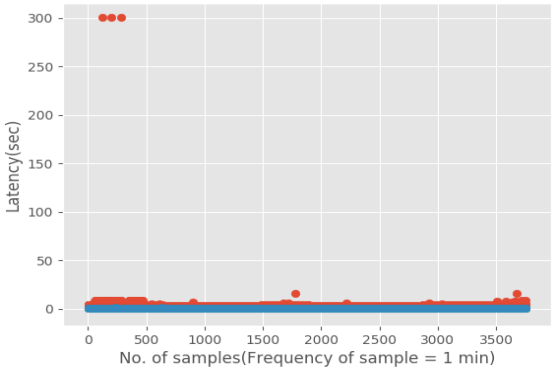
The combination of fog and cloud model is used when we have to do very heavy computations which are not possible in less powerful fog devices. The computation is offloaded to a more powerful cloud infrastructure. We have computed the turnaround time (latency) for sending data from edge devices, where the data is generated to fog gateway and then to cloud. We have used ThingSpeak for our experimentation. The combined data of different sensors are separated either at the edge gateway or at a fog node.

The turnaround time for each sensor to reach the cloud is plotted in Fig.9 a,b,c The orange points in the graph are the time required using the cloud model. An orange dot in Fig.9 a,b,c shows the distribution of time taken to upload all the sensor data to their corresponding ThingSpeak cloud channel. The separation of data is done at the fog level.

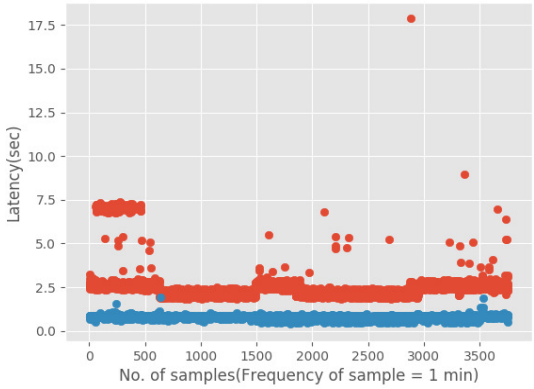
B. Result of fog+fog Model

The fog cluster model is used in applications that required frequent data access and quick response time. The Openfaas distribute the data at the device and send to the fog cluster for further processing. In another experiment, the data is segregated and then processed at the fog level. Fig.9 a,b,c shows the time taken for the sensor data received and processed at the fog node. Time taken by the fog model is represented by the blue points in all the graphs. It is clear that the turnaround time hence latency for the cloud model is higher than the fog model. In the fog model, the data can be processed locally with minimum data transfer hence better result. But the fog architecture is limited by the less powerful micro-controller used in our experimentation as fog nodes. The average latency delay between the cloud model and fog model is 2 sec for the two sensing systems deployed in the building.

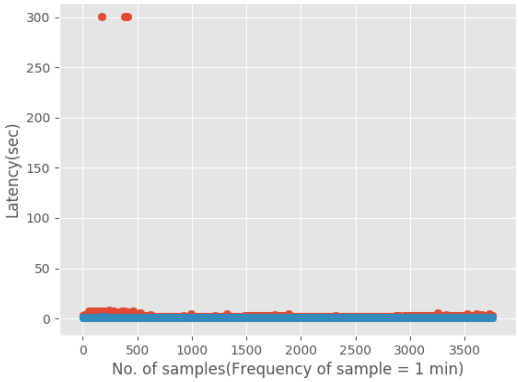
Similarly, the data related to Philips Hue Lights such as Light intensity, Light temperatures and Color values too have reduction in the response time as compared to cloud model.



(a)



(b)



(c)

Fig.9 a) DHT11 sensor data latency comparison between cloud and fog node
b) MQ135 sensor data latency comparison between cloud and fog node
c) MQ2 sensor data latency comparison between cloud and fog node

An orange dot indicates the latency of the sensor data received at the cloud model, and a blue dot indicates the latency of the sensor data received at the fog node.

V. CONCLUSION AND FUTURE WORK

From this study, we can conclude the following points:

- The serverless mechanism will be most appropriate in the event that you aren't worried about the issues related to cloud lock-in.
- The arrangement of serverless computing with the fog framework under the IoT theme does not require the transmission of data from the device to the cloud. Hence, expenses can be reduced considerably.
- The developed framework using open source IoT arrangements is effective as compared to a cloud-centric system where the application demand swift actions based on real-time data.
- The solution does not require frequent upload of data to the cloud. Data can be filtered before sending it to the cloud.
- The serverless model does not charge for idle time as compared to the cloud model.
- The internal system administration process is trivial compared to the cloud agnostic model.
- It is adaptable and deficiency tolerant structure
- It lessens the improvement and arrangement costs and time periods.
- It gives us flexibility in using openHAB and customer unit software.
- By distributing information across a fog instead of concentrating important information in one part of the network, enhanced privacy can be achieved.

In this work, devices are abstracted to give a functional view to the user. There is a provision for device authentication by registering them on to the system and only the authenticated devices are allowed to perform any action. This will protect the system from Denial of Service (DoS) attack as well as data security can be achieved. The future scope of this work is to build a system more robust, secure and incorporate a few predictive learning models to extract some useful information from sensor data.

REFERENCES

[1] Gopika Premsankar, Mario Di Francesco, and Tarik Taleb. "Edge Computing for the Internet of Things: A Case Study", IEEE Internet of Things Journal 5 (2018), pp. 1275–1284.

[2] Ashkan Yousefpour, Genya Ishigaki, and Jason P. Jue. "Fog Computing: Towards Minimizing Delay in the Internet of Things". IEEE International Conference on Edge Computing (EDGE) (2017).

[3] Ebraheim Alsaadi and Abdallah Tubaishat. "Internet of Things: Features, Challenges, and Vulnerabilities", International Journal of Advanced Computer Science and Information Technology (IJACSIT) 4 (2015).

[4] Faycal Bensaali , Xiaojun Zhai , Abbes Amira , and Lu Liu. "Guest Editorial Special Issue on Real-Time Data Processing for Internet of Things".IEEE Internet of Things Journal 5 (2018), pp. 3487– 3490.

[5] Aazam M, Zeadally S, and Harrass KA. "IoT Stream Processing and Analytics in the Fog". In: IEEE Communications Magazine (2018), pp. 46–52.

[6] Weisong Shi , Jie Cao, Quan Zhang, Youhuizi Li , and Lanyu Xu. "Edge Computing: Vision and Challenges". IEEE Internet of Things Journal 3 (2016), pp.637–646.

[7] Stephanie B Baker, Wei Xiang, and Ian Atkinson. "Internet of Things for Smart Healthcare: Technologies, Challenges, and Opportunities". IEEE Access 5 (2017), pp. 26521–26544.

[8] Gilsoo Lee, Walid Saad, and Mehdi Bennis. "An Online Optimization Framework for Distributed Fog Network Formation With Minimal Latency",IEEE Transactions on Wireless Communications 18 (2019), pp. 2244– 2258.

[9] Elena I. Gaura, James Brusey, Michael Allen, Ross Wilkins, Dan Goldsmith, and Ramona Rednic. "Edge Mining the Internet of Things". In: IEEE Sensors Journal 13 (2013), pp. 3816– 3825

[10] Ahmet Turan Ozdemir, Cihan Tunc, and Salim Hariri. "Autonomic Fall Detection System". In: IEEE 2nd International Workshops on Foundations and Applications of Self Systems (2017).

[11] Tadapaneni, N. R. (2016). Overview and Opportunities of Edge Computing. Social Science Research Network.

[12] Romana Shahzadi, Ambreen Niaz, Mudassar Ali, Muhammad Naeem, Joel J. P. C. Rodrigues, Farhan Qamar, and Syed Muhammad Anwar. "Three tier fog networks: Enabling IoT/5G for latency sensitive applications". In: IEEE Communications 16 (2019), pp. 1–11.

[13] Nitinder Mohan and Jussi Kangasharju, "Edge-Fog cloud: A distributed cloud for Internet of Things computations", IEEE Cloudification of the Internet of Things (CIoT)(2016).

[14] Cloud Computing Basics. Last accessed 24 Aug 2018. URL: https:/ / www.ibm.com/blogs/cloud-computing/2014/02/04/cloud-computing-basics/.

[15] Cloud Computing. Last accessed 24 Aug 2018. URL: https:/ /www. bartleby.com/essay/Cloud-Computing-Provides-A-Shared-Pool

[16] Tadapaneni, N. R. (2019). Role of Fog Computing in the Internet of Things. International Journal of Scientific Research and Engineering Trends.

[17] Najmul Hassan, Saira Gillani, Ejaz Ahmed, Ibrar Yaqoob, and Muhammad Imran. "The Role of Edge Computing in Internet of Things". In: IEEE Communications Magazine, 56(2018).

[18] Edge Computing, Last accessed 25 Aug 2019. URL: https://en.wikipedia.org/wiki/Edge computing.

[19] Fog Computing. Last accessed 30 Aug 2018. URL: https: //en.wikipedia.org/wiki/Fog computing.

[20] Qiang Fan and Nirwan Ansari. "Towards Workload Balancing in Fog Computing Empowered IoT", IEEE Transactions on Network Science and Engineering (2018).

[21] Raspberry Pi, Last accessed 25 Sep 2018. URL: https: //www.raspberrypi.org/blog/

[22] What is a Raspberry Pi? Last accessed 25 Sep 2018. URL: https://opensource.com/resources/raspberry-pi

[23] Arduino - Tutorials. Last accessed 28 Sep 2018. URL: https://www.arduino.cc/en/Tutorial/HomePage?from=Main.Tutorials

[24] Wei Yu, Fan Liang, Xiaofei He, William Grant Hatcher, Chao Lu, Jie Lin, and Xinyu Yang . "A Survey on the Edge Computing for the Internet of Things". In: IEEE Access 6 (2017), pp. 6900–6919.

[25] Arduino Tutorial. Last accessed 28 Sep 2018. URL: https://www.tutorialspoint.com/arduino/

[26] Field-programmable gate array, Last accessed 20 Sep 2018. URL: https:/ /en.wikipedia.org/wiki/

[26] Wei Song , Ning Feng, Yifei Tian, and Simon Fong. “An IoT-Based Smart Controlling System of Air Conditioner for High Energy Efficiency”, IEEE International Conference on Internet of Things (iThings) (2017).
[27] Antonio Brogi , Stefano Forti, Ahmad Ibrahim, and Luca Rinaldi. “Bonsai in the Fog: An active learning lab with Fog computing” IEEE Third International Conference on Fog and Mobile Edge Computing (FMEC) (2018)
[28] Cloud Service Models. Last accessed 28 Oct 2018. URL: https://www.ibm.com/cloud/learn/iaas-paas-saas.
[29] Cloud Computing, Last accessed 20 Oct 2018. URL: https://en.wikipedia.org/wiki/Cloud computing.
[30] Virtualization Technology. Last accessed 18 Dec 2018. URL: https:/ /www.vmware.com/in/solutions/virtualization.html.
[31] Virtualization in Cloud Computing. Last accessed 18 Dec 2018. URL:https://www.sam-solutions.com/blog/virtualization-techniques-in-cloud-computing/.
[32] Connected Boulevard – It’s what makes Nice, France a Smart City. https://blog.iiconsortium.org/2014/09/connected-boulevard-its-whatmakes-nicefrance-a-smart-city.html. Last accessed 15 Feb 2019.
[33] Luis Sanchez, Veronica Gutierrez, José Antonio Galache, Pablo Sotres, Juan Ramon Santana, Javier Casanueva, and Luis Munoz. “SmartSantander: Experimentation and service provision in the smart city”. In: 16th Inter-national Symposium on Wireless Personal Multimedia Communications (WPMC) (2013).
[34] Cities Get Smarter- MIT Technology Review. Last accessed 25 Jan 2019. URL: https://www.technologyreview.com/business-report/ cities-get- smarter/
[35] Shusen Yang. “IoT Stream Processing and Analytics in the Fog”. In: IEEE Communications Magazine 55 (2017), pp. 21–27.
[36] Oladayo Bello and Sherali Zeadally. “Intelligent Device-to-Device Communication in the Internet of Things”, IEEE Systems Journal 10 (2016), pp. 1172– 1182.
[37] Gang Xu, Edith C.-H. Ngai, and Jiangchuan Liu. “Ubiquitous Transmission of Multimedia Sensor Data in Internet of Things”, IEEE Internet of Things Journal 5 (2018), pp. 403–414.
[38] Rasika S. Ransing and Manita Rajput. “Smart home for elderly care, based on Wireless Sensor Network”. In: IEEE International Conference on Nascent Technologies in the Engineering Field (ICNTE) (2015).
[39] Shalom Greene, Himanshu Thapliyal, and David Carpenter. “IoT-Based Fall Detection for Smart Home Environments”. In: IEEE International Symposium on Nanoelectronic and Information Systems (iNIS) (2016).
[40] IEEE 802.15.4. Last accessed 20 Jan 2 019. URL: https: //en.wikipedia.org/wiki/IEEE 802.15.4
[41] IEEE 802.11. Last accessed 24 Mar 2019. URL: http: //www.ieee802.org/11/.
[42] Dominik Kobylarz and Jacek Danda. “A Common Inter-face for Bluetooth-based Health Monitoring Devices”. In: 29th Southern Biomedical Engineering Conference (2013).
[43] Telosb Sensors. Last accessed 20 Jan 2019. URL: https: //telosbsensors.wordpress.com/.
[44] Intel Edison. Last accessed 20 Jan 2019. URL: https: //en.wikipedia.org/wiki/Intel Edison.
[45] NodeMcu. Last accessed 28 Jan 2019. URL: https:// www.nodemcu.com/index en.html.
[46] Digi XBee Ecosystem. Last accessed 28 Jan 2019. URL: https://www.digi.com/xbee
[47] DHT11 and DHT22 Sensors. Last accessed 25 Feb 2019. URL: https://howtomechatronics.com/tutorials/arduino/dht11-dht22-sensors - temperature - and - humidity-tutorial-using-arduino/
[48] Gas Sensor Basics. Last accessed 25 Feb 2019. URL: https: / / www. dnatechindia.com/GAS-Sensor-Basics.html.
[49] OpenHAB. https / /www.openhab.org Last accessed 21 Jun 2019.
[50] Introduction to OpenFaaS. https://docs.openfaas.com/ Last accessed 10 Apr 2019.
[51] MQTT. Last accessed 20 Feb 2019. URL: http://mqtt.org/
[52] REST API Tutorial. https://www.restapitutorial.com/ Last accessed 21 Feb 2019.
[53] Sandeep Pirbhulal, Heye Zhang, Subhas Chandra Mukhopadhyay, Wanqing Wu and Yuan-Ting Zhang. “Heart-Beats Based Biometric Random Binary Sequences Generation to Secure Wireless Body Sensor Networks” IEEE Transactions on Biomedical Engineering, 65 (12), 2751-2759 http://ieeexplore.ieee.org/document/8314739, 2018.
[54] Wanqing Wu, Heye Zhang, Sandeep Pirbhulal, Subhas Chandra Mukhopadhyay and Yuan-Ting Zhang, Assessment of Biofeedback Training for Emotion Management Through Wearable Textile Physiological Monitoring System, IEEE SENSORS JOURNAL, VOL. 15, NO. 12, DECEMBER 2015, pp. 7087-7095.
[55] G. M. Mendez, M.A.M. Yunus and S. C. Mukhopadhyay, A WiFi based Smart Wireless Sensor Network for Monitoring an Agricultural Environment, Proceedings of IEEE I2MTC 2012 conference, IEEE Catalog number CFP12MT-CDR, ISBN 978-1-4577-1771-0, May 13-16, 2012, Graz, Austria, pp. 2640-2645.
[56] G. M. Mendez, M.A.M. Yunus and S. C. Mukhopadhyay, A WiFi based Smart Wireless Sensor Network for Monitoring an Agricultural Environment, Proceedings of IEEE I2MTC 2012 conference, IEEE Catalog number CFP12MT-CDR, ISBN 978-1-4577-1771-0, May 13-16, 2012, Graz, Austria, pp. 2640-2645.
[57] N.K.Suryadevara, M.T.Quazi and S.C.Mukhopadhyay, Intelligent Sensing Systems for measuring Wellness Indices of the Daily Activities for the Elderly, proceedings of the 2012 Eighth International Conference on Intelligent Environments, Mexico, June 1-3, 2012, pp. 347-350.
[58] Suryadevara N.K., Negi A., Rudraraju S.R., A Smart Home Assistive Living Framework Using Fog Computing for Audio and Lighting Stimulation. In: Satapathy S., Raju K., Shyamala K., Krishna D., Favorskaya M. (eds) Advances in Decision Sciences, Image Processing, Security and Computer Vision. ICETE 2019. Learning and Analytics in Intelligent Systems, vol 3. Springer, Cham.
[59] B.B Prahlada Rao, Paval Saluia, Neetu Sharma, Ankit Mittal, and Shivay Veer Sharma . “Cloud computing for Internet of Things and sensing based applications”. In: IEEE Sixth International Conference on Sensing Technology (ICST) (2012).
[60] Liu, Allan and Yu, Ting, Overview of Cloud Storage And Architecture (2018). International Journal of Scientific & Technology Research

Suvajit Sarkar has recently completed his M.Tech from The University of Hyderabad, India. This research is a part of his M.Tech thesis. His research interests are in the areas of Cloud Computing, Serverless Computing, Fog Computing and IoT.

Rajeev Wankar is working as a Professor in the School of Computer and Information Sciences (SCIS) at University of Hyderabad (UoH). He earned Ph.D. in Computer Science from the School of Computer Sciences, Devi Ahilya University Indore. In 1998, the German Academic Exchange Service (DAAD) awarded him "Sandwich Model" fellowship. His research interests are in the areas of Cloud Computing and Grid Computing.

Satish Narayana Srirama is a Research Professor and the head of the Mobile & Cloud Lab at the Institute of Computer Science, University of Tartu, Estonia and a Visiting Professor at University of Hyderabad, India. His current research focuses on cloud computing, mobile web services, mobile cloud, Internet of Things, fog computing, migrating scientific computing and enterprise applications to the cloud and large scale data analytics on the cloud. He is IEEE Senior Member, an Editor of Wiley Software: Practice and Experience journal and a program committee member of several international conferences and workshops.

Nagender Kumar Suryadevara received the Ph.D. degree from the School of Engineering and Advanced Technology, Massey University, New Zealand, in 2014. He has authored/co-authored one book and over 40 papers in different international journals/conferences and book chapters. His research interests are in the domains of wireless sensor network, Internet of Things, and Time-Series data mining.