Journal of Engineering Science and Technology
Vol. 13, No. 8 (2018) 2504 - 2519

A REFERENCE MODEL FOR TESTING INTERNET OF THINGS BASED APPLICATIONS

PRAMOD MATHEW JACOB*, PRASANNA MANI

School of Information Technology, Vellore Institute of Technology, Vellore Campus, Tamil Nadu, India
*Corresponding Author: pramod3mj@gmail.com

Abstract

The entire world is moving towards the era of smart technology. Internet of Things (IoT) is the key factor behind this revolutionary change. A simple IoT system comprises a device or thing with embedded sensors, which are connected to an application through the internet. The thing or device can be remotely controlled and monitored from anywhere in the world through the internet. Due to this feature, it has been applied in various domains like home automation, Patient health monitoring, agricultural sector, environment surveillance and much more. Due to the increased usage of IoT based systems, there arises a need for validating and verifying the devices in all aspects. Though different IoT developers follow their own methodologies to test their IoT device, there is a need for a generic test model for validating IoT applications. We propose a test model for testing an IoT system by integrating the existing test models for software, hardware and communication protocols. We evaluated our model by executing a case study for Intrusion Detection System (IDS) using IoT. Experimental evaluation proves that this model can be used as a reference model for IoT developers for evaluating the performance and capabilities of an IoT system.

Keywords: Internet of things (IoT), Intrusion detection system, IoT testing, Software testing, System testing.

1. Introduction

IoT is a term, which was quite unfamiliar to ordinary people in old days. However, in this era of smart technology and smarter systems, IoT became much popular. IoT is an emerging field, which can play a vital role in almost every industry and disciplines including agriculture, health sector, home automation, aviation and transport, defence, military applications and much more.

IoT is the internetworking of physical devices, vehicles (also referred to as "connected devices" and "smart devices"), buildings, and other items-embedded with electronics, software, sensors, actuators, and network connectivity that enable these objects to collect and exchange data [1, 2]. IoT can be used to make the objects or things smarter by remotely sensing or controlling it. IoT comprises things that have unique identities and are connected to the internet [3]. IoT describes a system where items in the physical world, and sensors within or attached to these items, are connected to the internet via wireless and wired internet connections. These sensors use various types of local area connections such as Radio Frequency Identification (RFID), Near Field Communication (NFC), Wireless Fidelity (Wi-Fi), Bluetooth, and ZigBee. Sensors can also have wide area connectivity [4] such as Global System for Global System for Mobile Communication (GSM), General Packet Radio Service (GPRS), Third Generation (3G) services and Long Term Evolution (LTE).

The typical IoT architecture is illustrated in Fig. 1. It is a networked set of devices and software embedded components. The embedded system can be a mini processor or boards, which have some processing capability. For example, Arduino, Raspberry Pi, Intel Galileo, etc. Internet Protocol Version 6 (IPv6) is used for IP addressing though it can address up to millions of devices. There will be some application domain where the central things or sensors are deployed for monitoring and controlling. It will sense the relevant information and will pass towards some Cyber-Physical System (CPS), which performs the computation and coordination. The connected components are communicated using wireless technologies. Based on the instructions or feedback, actuators initiate the necessary actions to be carried out in the scenario.

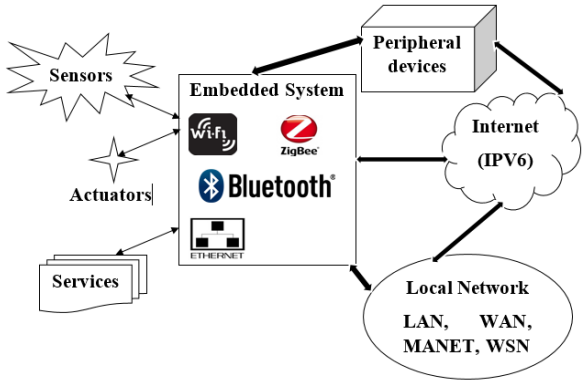

Fig. 1. Typical architecture of Internet of Things (IoT).

A typical IoT possesses the following characteristics:

- Connectivity: All the devices, sensors and equipment should be connected to each other through a secure network.
- Things: Things' in IoT perspective can be a wide variety of devices such as biochip transponders on pet animals or farm animals, heart monitoring implants, electric clams in coastal waters, automobiles with built-in sensors, DNA (Deoxyribonucleic Acid) analysis devices for environmental/food/pathogen monitoring or field operation devices that assist firefighters in search and rescue operations [4]. Things can be either actuators, sensors or even gateway providers. These devices or things collect necessary data using existing technologies and then autonomously flow the data between other devices.
- Data: Data is the adhesive agent in the IoT, which, initiates action and intelligence.
- Communication: Every component or things are communicating with each other for data transfer or for initiating system monitoring and controlling. Communications are usually initiated by using wireless protocols.
- Intelligence: Sensors usually makes the IoT devices intelligent with some processing element for data analytics and pre-processing.
- Action: It can be either manual action or action based upon debates regarding phenomena (for instance in climate change decisions) and automation, usually initiated by actuators.
- Ecosystem: The environment or domain of the IoT.

Testing is a quality control activity, which involves defect detection and bug fixation. Testing is part of the System Development Life Cycle (SDLC) process to validate and verify the working of a developed product or application [5]. Testing can be performed at different stages of the development process depending upon the methodology and tools being used and usually begins after the requirement confirmation phase. The initial phase is at unit level where it mainly focuses on testing individual components, devices, and communication layers. When IoT components are interconnected, we perform testing to find the interfacing errors. The ultimate purpose of system testing is to satisfy the stakeholders and to ensure the quality of an application.

IoT system testing can also be mentioned as the process of performing validation and verification to an IoT based system or an application that meets the business-oriented and technical-oriented functional requirements that guided in its design phase and development. Validating and Verifying (V&V) [6] is the process of ensuring that a software or system meets the requirements mentioned in System Requirement Specification (SRS) document and that it fulfill its intended functionality. It can be considered as a methodology to ensure product quality. The terms can be defined as follows:

- Verification: It is the process of ensuring whether the developed product is built in the right manner.
- Validation: It is the process of ensuring whether the developed product is what they expected to be.

We propose a hybrid model by integrating the testing models of different hardware, software and communication protocols.

This paper is organized into five sections as follows: Section 2 reviews the related works in the evaluation of numerous IoT based testing techniques. Section

3 illustrates the necessity of an IoT test model and provides an overview of our proposed test model. Section 4 discusses the different phases of testing carried out in the various IoT system levels and finally we summarize our conclusions in section 5. We have also discussed the future works that can be implemented in near future in conclusion section.

2. Related Works

We have performed a literature survey to identify what are all testing methodologies used in IoT based systems so far. There are only very few works done in this area though IoT is an emerging area in Information Technology.

Marinissen et al. [7] discussed the existing IoT trends and the challenges involved in testing IoT based applications. They derived the testing challenges from different viewpoints. Design viewpoint consists of test challenges regarding sensors, ZigBee or Wi-Fi radio and power management. They also discussed the need for IoT manufacturing test as well as the necessity of security testing in IoT based applications. Their findings claim that developing a quality assured IoT product with minimal cost is the ultimate challenge for IoT developers.

Reetz et al. [8] proposed a new approach for testing IoT based application built on a code insertion methodology. It is derived using the semantic description of IoT based service. Their proposed architecture framework consists of test design engine and test execution engine as well as a sandbox environment, which has the capability of simulating the networking aspects of the system. Code insertion is done manually. However, this methodology lacks the validation of the hardware and physical things of an IoT based system.

Leal et al. [9] proposed an IoT test model for testing RFID based technologies and IoT used in Brazilian intelligent road transport systems. They designed a testbed and framework for evaluating the performance and functionalities of RFID based systems used in smart road transport, but this methodology addresses only the test cases in their domain.

Rosenkranz et al. [10] proposed an IoT testing framework, which supports continuous integration techniques for hardware and software. The proposed model uses test clusters so that anyone can deploy test platforms to this system. This distributed testing strategy allows shared access to individual IoT based systems or for fully edged IoT testbeds. It generates test cases for different operating platforms. They claim that their architecture can be used for network interoperability testing using forwarding communication from one IoT system under testing to another with location transparency.

From our survey, we came to derive that there is no generic IoT testing model available for testing IoT based applications. Most of the existing systems fail to test the integrated IoT system. We integrate the best features of previous works and importing some IoT test methodologies used by various industry people to assure the performance and efficiency of IoT systems.

3. Proposed Model

IoT differs from other web applications and embedded system applications due to the following characteristics [11].

- Integrated system with hardware, software, sensors, connectors, and gateways.
- Live data stream analytics with complex event processing.
- Support for data volume and velocity.
- Visualization of big data
- Cloud services and computing.
- Distinct interoperable communication protocols

Due to the above-said characteristics, it is not an easy task for the IoT architects to evaluate the performance and efficiency of the IoT based systems. The numerous challenges in IoT testing are listed below.

- Dynamic environment: We cannot follow the test principles of software applications working in a defined environment though it deals with the integrated working of different sensors and devices controlled by some application software.
- Complexity: Though the IoT system operates on multiple devices and communication protocols, there is a large set of use-cases, which make the testing activity complex.
- Scalability: It is not an easy task to create a scalable test environment since most IoT based systems are dynamic and there will be a need to expand the system boundaries and capabilities at any time [12].
- Reliability: Due to the inter-processing and intercommunicating devices, it is not feasible to accurately measure the reliability of an IoT system.
- Security: Subsystems and components may be managed by third parties, which may lead to privacy and security issues.
- Hardware quality and accuracy: Though the IoT system uses much hardware, the quality and efficiency of each component we use should be thoroughly passed the quality assurance. Accuracy plays a vital role in most IoT systems.

Incorporating the above-mentioned challenges we propose a derived IoT test model as illustrated in Fig. 2. The proposed test model comprises of four levels of testing. Each level performs a set of verification and validation procedures to ensure the functionality and capabilities of IoT based system.

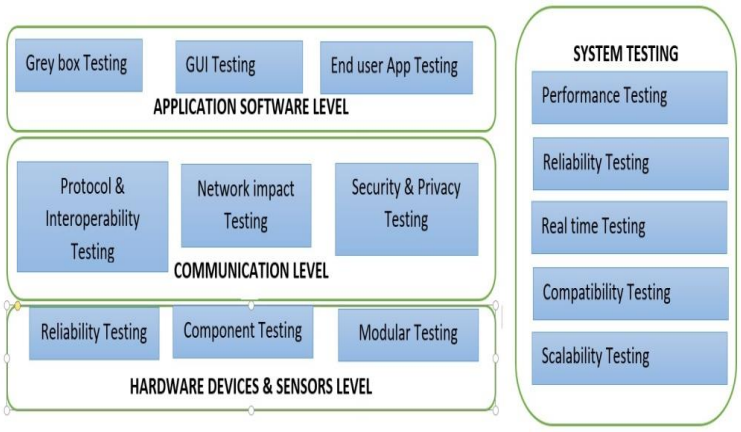

Fig. 2. Proposed IoT testing model.

3.1. Hardware devices and sensors level

This level is considered the physical level of the IoT system. Physical level of IoT consists of distinct components like sensors, monitoring devices like surveillance camera and a central processor (Raspberry Pi [13], Arduino [14], and Intel Galileo [15]). IoT based system may or may not use sensors based on the application domain. Sensors generally used include a temperature sensor, proximity sensor, pressure sensor, PIR (Passive Infrared) motion sensor, humidity sensor, etc. The developers should ensure that the component used at the physical level is working perfectly. The set of tests to be carried out at this level include component testing, modular testing, and reliability testing.

3.1.1. Component testing

The different class of components used in IoT system may be manufactured in external industry or plants. The developer should ensure that each component is doing its functionalities accurately and precisely. Precision testing, continuity testing, functional testing, etc., should be done in this level for each component used in IoT system.

3.1.2. Modular testing

Different components and devices are connected to design the physical part of the IoT system. Each developed physical module is individually tested to ensure its functionality.

3.1.3. Reliability testing

Reliability describes the ability of a system or component to perform its intended functionality under predefined conditions for a specified period of time. It is also defined as the probability that, which, the system is available to perform its tasks. Reliability values range between 0 and 1. Reliability of a hardware component can be either periodically monitored or randomly assessed. The hardware component failure rate usually follows a bathtub curve as shown in Fig. 3.

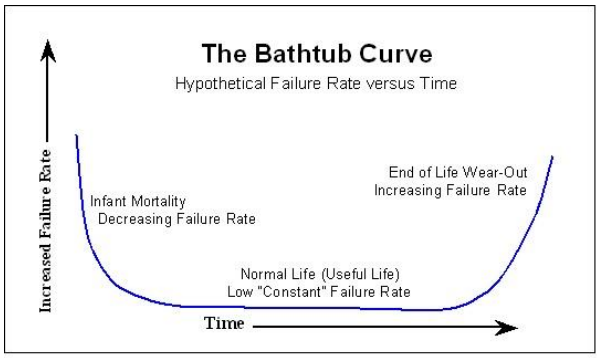

Fig. 3. Hardware reliability.

3.2. Communication level

The discrete devices are communicated with a cloud service using a mobile or web application. The IoT communication model can be a request-response model, the

publish-subscribe model, push-pull model or an exclusive pair model [3]. The various communication protocols generally used in a typical IoT system [3] is shown in Fig. 4. It consists of four layers: the link layer, network or internet layer, transport layer and application layer. The testing activities to be performed in IoT communication level is discussed below.

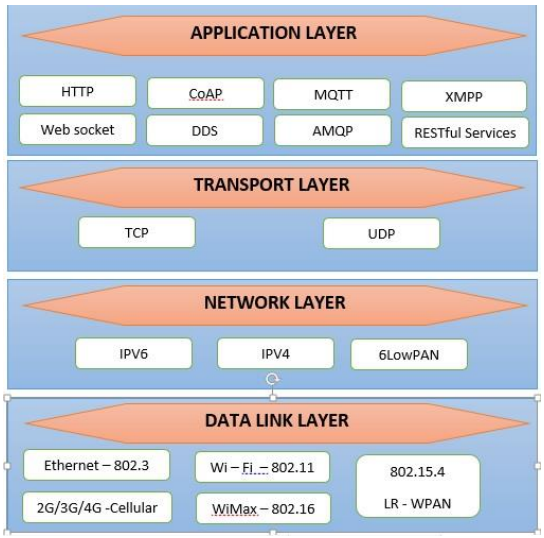

Fig. 4. Protocols in IoT system.

3.2.1. Protocol and interoperability testing

Though the system uses a wide range of protocols, it is the responsibility of the developer to ensure that every devices and component use interoperable communication protocols. This testing is compulsory though we use devices that operate in Body Area Network (BAN) to Wide Area Network (WAN). Protocol analyser and simulator can be used for this purpose. Protocol analyser ensures decoding is properly executed along with call and session analysis. The simulator simulates numerous entities and elements of the network. Usually, protocol and interoperability testing are performed using DUT (Device Under Test) to other devices like routers and switches by configuring protocol inside it. Conformance testing [16], network feature testing and negative testing can also be performed at this level.

3.2.2. Security and privacy testing

Though it uses different interoperable communication protocols in a distributed environment, the IoT based system is much vulnerable to security threats. Privacy of data is also to be preserved in some application domains like the military, health monitoring, etc. In this scenario, it is the onus of the developer to ensure the security aspects of data encryption and decryption, data privacy and protection, device identity authentication and trust among different cloud and mobile services.

3.2.3. Network impact testing

It is performed for analysing the quantitative and qualitative performance of a deployed IoT based application in real time network constraints. This methodology

includes testing IoT based system with a combination of different topologies, network area, network size and other environmental conditions.

3.3. Application level

Most of the IoT based system consists of a mobile application or a web-based application to remotely monitor and control the devices or things. These applications may run on distinct IoT Operating System platforms like Windows 10 IoT Core, Raspbian, NOOBS, Snappy Ubuntu Core, PINET, RISC OS, etc. But developers always give priority to developing Android based applications than other alternatives so that the IoT system can be managed using end user's mobile phones. Application software level performs the generic software testing principles as discussed below.

3.3.1. Grey box testing

It is the hybrid form of software testing, which, integrates the best features of both black box testing and white box testing. It evaluates both the behavioral and functional features of the system. It ensures that the developed IoT system is performing its intended functionality and is built according to the standard guidelines. Different sorts of integration testing methodologies can also be done at this level to detect the interface errors. Regression testing procedures can be followed in case of continuous integration of modules.

3.3.2. GUI based testing

Graphical User Interface (GUI) has much to perform in an IoT based application, though the end user uses some smart devices to remotely control objects and things. The GUI should be designed in such a manner that it should be user-friendly and unambiguous. Standard notations and icons should be followed while designing the user interface. It can be evaluated by providing a beta version to the end user so that end-user feedback can be collected. Based on user feedback the GUI design can be improvised and can make it more acceptable.

3.3.3. End-user app testing

After developing the application in its full version and is when integrated with the IoT based system, end-user application testing can be performed. This can be considered as a sort of system testing. It can be included in alpha, beta and acceptance testing. Alpha testing is usually performed by the developers to ensure the system functionalities. Beta testing is done by a random set of end users to ensure that the software application is behaving as user expectations. Acceptance testing or end-user testing is performed by the target user to determine whether to accept or reject this IoT based system.

3.4. System level

System level testing mainly focuses on non-functional requirements and performance capabilities of an IoT based system. Performance capabilities are like scalability, reliability, availability, portability, etc. IoT system testing includes the following test procedures.

3.4.1. Performance testing

IoT performance testing mainly relies on network-based factors like latency, bandwidth, packet loss, handling concurrent users, etc. Test cases should be generated for typical and nontypical use cases to handle the exceptional use cases. It can be supplemented by peak load testing.

3.4.2. Reliability testing

As we already discussed the reliability testing at hardware devices and sensors level, this reliability testing ensures that the entire IoT based system works precisely and accurately during a given period of time without failure.

3.4.3. Real-time testing

The CPU testing and virtualized testing will not capture all the errors. Though IoT systems have dynamic behavior, there is a chance of occurring a new genre of errors while working in real life situations. So before deployment, the developers should perform testing in the real-time environment with realistic test data and system conditions. This is very important in case of critical IoT based applications used in health care and remote monitoring applications used in sensitive regions or places where a human cannot directly intervene and control.

3.4.4. Compatability testing

The developed IoT system should have the capability to operate on all standard work conditions. Though the IoT system consists of distinct devices, which, use diverse software and hardware platforms, there arises a need for performing portability testing.

3.4.5. Scalability testing

Due to the dynamic behavior of IoT systems, any time the range or region of IoT application can be expanded to more devices or things. In this case, the entire IoT system should have the ability to expand without any failure. This has to be ensured by performing scalability test procedures supplemented by load testing, volume testing, penetration testing, etc.

Our proposed model hence deals with all the dimensions of an IoT based system by evaluating hardware, software, communication and system levels.

4. Results and Discussion

Our proposed model is evaluated by choosing an IoT based system for intrusion detection in a particular room or area as discussed below. Detailed architecture and design report are available on our blog [17] titled "Intrusion Detection System using IoT".

Security systems play a significant role in our modern society. It is needed in most environments including home, offices, industries, manufacturing plants, power plants, restricted areas or war fields and much more. It is not economical to deploy a security person for simply monitoring all the nooks and corners of an area. We implemented an IoT based surveillance system using the camera to remotely monitor a particular area or region. We use PIR motion sensors and surveillance cameras

integrated with a Raspberry Pi module for this purpose. Whenever an intruder enters into a particular region, the motion sensor detects the motion and it communicates with the Raspberry Pi. The Raspberry Pi module sends a signal to the integrated camera, take pictures of that region, and is notified the administrator with an alarm light or text message. The system architecture is shown in Fig. 5.

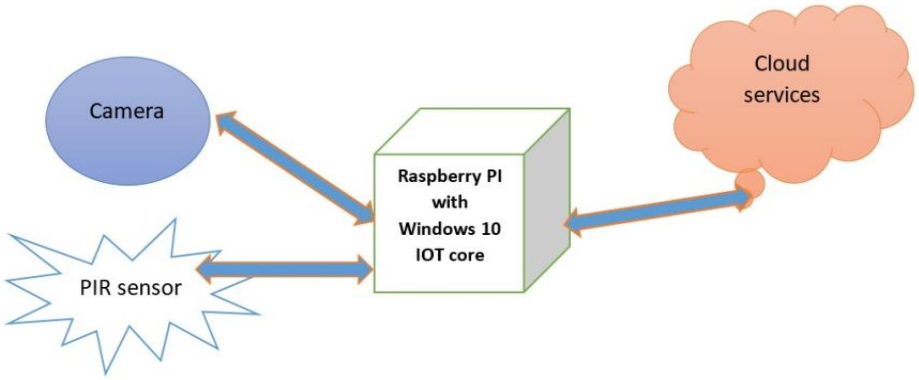

Fig. 5. Architecture of intrusion detection system.

The Intrusion Detection System consists of the following four modules:

- IoT based surveillance module*:* This module consists of PIR motion sensor connected to a Raspberry Pi to detect motions. Whenever a motion is sensed, Raspberry Pi communicates with the web camera installed in the region to take the snapshots.
- Communication module: This module initiates the communication between distinct devices like camera, sensors and storage systems. It uses IEEE 802.11 Wi-Fi standard.
- Storage module: This module stores the images taken by the camera. It can be either system storage or even a cloud storage service.
- User application module: This module provides an Android or mobile app installed on the user system. Whenever an intrusion is detected, snapshots will be received to this application and it will notify the user to initiate some necessary actions.

The numerous modules in this system are to be evaluated for its capabilities and functionalities. All four modules in the Intrusion Detection System can be mapped to any one of our proposed model testing levels as shown in Table 1.

Table 1. Applicable test levels for intrusion detection system.

Intrusion Detection System module	IoT testing model levels
IoT based surveillance module	Hardware devices and sensors level testing
Communication module	Communication level testing
Storage module	Application software level testing
User application module	Application software level testing
Integrated IoT system	System testing levels

System testing level can be done by integrating all four individual models to form the IoT based Intrusion Detection System. From this evaluation, it can be claimed that our IoT test model is applicable to any generic IoT based system for verifying and validating its functionalities. Our IoT test model reduces the burden over test engineer to choose, which, all testing is to be applied in which, levels of IoT. Though we followed a layered test model, it is possible to perform testing in different IoT modules using different test levels concurrently. This may lead to time-consuming in testing activity and hence improve productivity.

4.1. Testing summary of Intrusion Detection System

The following set of procedures are followed in testing the Intrusion Detection System in the various levels as mentioned in Table 1.

4.1.1. Testing the IoT based surveillance module

In this level, we performed the component verification testing for PIR motion sensor and PI camera. PIR motion sensor testing circuit [18] is shown in Fig. 6. Wire up the circuit in a breadboard and insert batteries as shown in Fig. 6. The PIR sensor requires some time to be stabilised. During this time, the LED may blink up. After the LED got off, it move to front of the motion sensor. If the PIR is working, it will light up the LED.

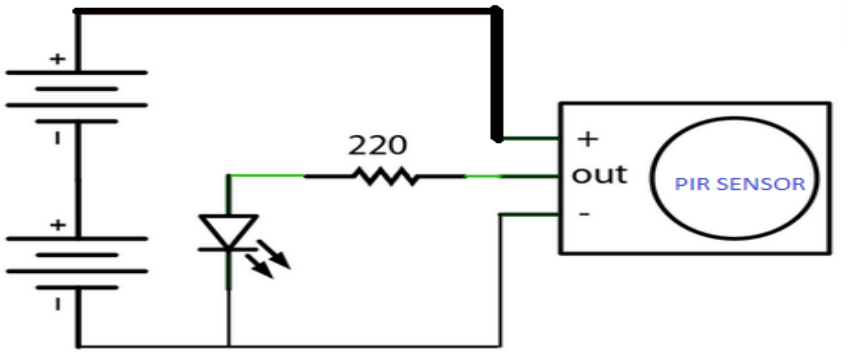

Fig. 6. Circuit for testing PIR sensor.

For component verification of PI camera module, connect the PI camera on Raspberry Pi board. Type the following command in the command line interface of the system.

raspistill–o testshot.jpg

If PI camera is working right, the camera mode will enable in the screen for a moment and the captured image can be found out in the system file list.

4.1.2. Testing the communication module

In this project, when the PIR sensor senses a motion, an image of the surveillance area should be captured by the PI camera and is sent to the registered email ID. This connection is established using the Simple Mail Transfer Protocol (SMTP). Python smtplib module is imported into our script file. The communication can be verified

by sending test data from source to destination. Detailed source script is available in our code file included in the project report [17].

4.1.3. Testing the storage module

The storage module used in this project is the micro SD card inserted in the Raspberry Pi board. The accessibility of the storage module can be verified from the command line interface by using the following command.

sudo fdisk -1

This command will display all the disk partitions in the system. We can verify whether our intended storage module is listed or not. In case if, the server is the storage module, send a ping request to ensure its availability.

4.1.4. Testing the user application module

The user application module can be tested using various testing procedures like, black box testing, white box testing, etc. We tested our module using path testing procedures included in white box testing. The Pseudo code for Intrusion Detection System is provided below and its equivalent control flow graph is shown in Fig. 7.

Pseudo code for Intrusion Detection System

MONITOR_AREA( )

1. Initialize
   PIR_sensor= "LOW"//Setting PIR pin voltage as low
   LED_PIN= "LOW"
2. Run PIR_sensor
3. If (PIR_sensor== "HIGH")//PIR pin returns high voltage when a motion is sensed
   Set LED_PIN= "HIGH"
   Image= CAPTURE_IMAGE( )
   SEND_DATA(Image)
   Endif
4. Else MONITOR_AREA( )

CAPTURE_IMAGE( )

1. Initiate camera and take snapshot
2. Save image in jpeg format
3. Return image

SEND_DATA(image)

1. Attach image as data in email
2. Set the destination address as = "destination_mailid".
3. Attach the text message "AREA IS UNSAFE".
4. Send the mail.

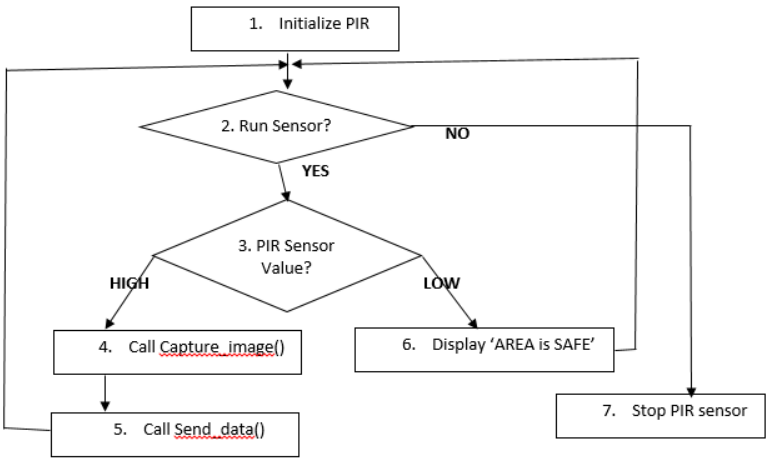

Fig. 7. Control flow graph (CFG) for the intrusion system.

The number of independent test paths is referred to as the cyclomatic complexity of a graph.

Cyclomatic complexity, *V(G)*=Number of closed regions in the graph +1 (1)

The independent paths in our CFG=2+1=3.

The three independent test paths are displayed below:

Path 1: 1–2–3–4–5–2-7.

Path 2: 1–2–3–6–2-7.

Path 3: 1–2-7.

We derived test cases for evaluating each independent test path and thereby ensured the path coverage for our system.

4.1.5. Testing the integrated IoT system

The entire system functionalities after both hardware and software integration are performed using the decision table methodology [19]. The decision table that we used for testing the Intrusion detection system is shown in Table 2.

Table 2. Decision table for testing intrusion system.

		RULES			
CONDITIONS	PIR sensor is "HIGH"	*T*	*T*	*F*	*F*
	LED is "ON"	*T*	*F*	*T*	*F*
ACTIONS	Capture the image of the area	*X*	*X*		
	Send the image to registered email	*X*			
	Restart monitoring	*X*	*X*	*X*	

By performing the above-detailed testing procedures, it is clear that our model can be used as a reference model for testing IoT applications. It does not only test the software part but also evaluates the hardware component verification. Though our testing model is a blended version of all good IoT based testing strategies, our IoT test model shows the following features like reduced testing time, support for concurrent

testing and enhanced system reliability. It also provides easiness in choosing a testing level for the test engineer and thereby ensuring the product validation in every possible aspect of testing. The current industrial IoT test models mostly focus on device and component testing whereas our model tests software part also. This characteristic makes our IoT test model different from other models used in industries. This testing model can be applied in various application areas like smart farming [20], weather forecasting systems, smart cities [3], home automation, etc. It can be also useful in research level and industry-based projects like Modeling of the photovoltaic panel by using Proteus [21], low-cost virtual instrumentation of PV panel characteristics using Excel and Arduino in comparison with traditional instrumentation [22], etc.

5. Conclusion

Though the entire world is moving towards smart technology and IoT based systems, still, there is no generic IoT test model available to validate an IoT based system. We proposed a derived test model from all existing hardware and software-based test models. Our model can be used as a reference test model for any IoT system developer since it addresses all the levels of testing in an IoT based system. The testing level begins right from hardware components and sensor quality check, verifying and validating the interoperability of different communication protocols used in distinct devices, ensuring the functionality and capabilities of the application program and finally the system testing to ensure the overall capabilities of an IoT system. Our module will be a reference model for the test engineers to determine, which, all levels of testing should be done in which, IoT modules. Apart from existing industrial models, our model validates the hardware, software and communication of an IoT system. The experimental analysis proves that our model can be practically used for validating real-time IoT products in an efficient and cost-effective manner. Our future works focus on evaluating the interoperability features of an IoT system using a suitable test bed.

Acknowledgement

We express our sincere thanks to the Vellore Institute of Technology (VIT) and Providence College of Engineering, Chengannur, for their support.

Abbreviations

2G	Second Generation
3G	Third Generation
4G	Fourth Generation
6LoWPAN	IPv6 over Low-Power Wireless Personal Area Network
AMQP	Advanced Message Queuing Protocol
BAN	Body Area Network
CoAP	Constrained Application Protocol
CPS	Cyber Physical System
CPU	Central Processing Unit
DDS	Data Distribution Service
DNA	Deoxyribonucleic Acid
DUT	Device Under Test
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communication

GUI	Graphical User Interface
HTTP	Hyper Text Transfer Protocol
IDS	Intrusion Detection System
IEEE	International Electrical and Electronics Engineers
IoT	Internet of Things
IP	Internet Protocol
IPv6	Internet Protocol Version 6
LED	Light Emitting Diode
LR WPAN	Low Rate Wireless Personal Area Network
LTE	Long Term Evolution
MQTT	Message Queuing Telemetry Transport
NFC	Near Field Communication
NOOBS	New Out Of the Box Software
OS	Operating System
PINET	PI Networks
PIR	PIR
REST	Representational State Transfer
RFID	Radio Frequency Identification
RISC	Reduced Instruction Set Computer
SD	Secure Digital
SDLC	System Development Life Cycle
SMTP	Simple Mail Transfer Protocol
SRS	System Requirement Specification
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
V&V	Verification and Validation
WAN	Wide Area Network
Wi-Fi	Wireless Fidelity
Wi-Max	Worldwide Interoperability for Microwave Access
WPAN	Wireless Personal Area Network
XMPP	Extensible Messaging and Presence Protocol

References

1. Brown, E. (2016). Who needs the internet of things? Retrieved October 23, 2016, from https://www.linux.com/news/who-needs-internet-things.
2. Wikipedia. (2016). Internet of things. Retrieved February 14, 2017, from https://en.wikipedia.org/wiki/Internet_of_things#cite_note-Linux_21OSP-2.
3. Bahga, A.; and Madisetti, V. (2015). *Internet of things: A hands-on approach*. Telangana, India: Universities Press.
4. Erlich, Y. (2015). A vision for ubiquitous sequencing. *Genome Research,* 25(10), 1411-1416.
5. Jacob, P.M.; and Mani, P. (2016). A comparative analysis on black box testing strategies. *Proceedings of International Conference on Information Science*. Kochi, India, 1-6.
6. Pressman, R.S. (1982). *Software engineering-A practitioner's approach*, (7th ed.). New York: McGraw-Hill.
7. Marinissen, E.J.; Zorian, Y.; Konijnenburg, M.; Huang, C-T.; Hsieh, P-H.; Cockburn, P.; Delvaux, J.; Rozic, V.; Yang, B.; Singelee, D.; Verbauwhede,

I.; Mayor, C.; van Rijsinge, R.; and Reyes, C. (2016). IoT: Source of test challenges. *Proceedings of 21st IEEE European Test Symposium (ETS)*. Amsterdam, Netherlands, 1-10.

8. Reetz, E.S.; Kuemper, D.; Moessner, K.; and Toenjes, R. (2013). How to test iot-based services before deploying them into real world. *Proceedings of the 19th European Wireless Conference*. Guildford, United Kingdom, 1-6.
9. Leal, A.G.; Santiago, A.; Miyake, M.Y.; Noda, M.K.; Pereira, M.J.; and Avanço, L. (2014). Integrated environment for testing IoT and RFID technologies applied on intelligent transportation system in Brazilian scenarios. *Proceedings of the IEEE Conference on Brazil RFID*. Sao Paulo, Brazil, 22-24.
10. Rosenkranz, P.; Wählisch, M.; Baccelli, E.; and Ortmann, L. (2015). A distributed test system architecture for open-source IoT software. *Proceedings of the Workshop on IoT Challenges in Mobile and Industrial Systems*. New York, United States of America, 43-48.
11. Infosys. (2016). Testing IoT applications-A perspective. Retrieved March 5, 2017, from https://www.infosys.com/IT-services/validation-solutions/Documents/testing-iot-applications.pdf.
12. Jacob, P.M.; and Mani, P. (2018). *Software architecture pattern selection model for internet of things based systems*. United Kingdom: IET Software.
13. Raspberry Pi. (2017). Raspberry pi. Retrieved on March 5, 2017, from https://www.raspberrypi.org/.
14. Arduino. (2016). Arduino boards. Retrieved on March 5, 2017, from https://www.arduino.cc/.
15. Intel Software Developer Zone. (2017). Intel galileo board documentation. Retrieved on March 5, 2017, from https://software.intel.com/en-us/iot/hardware/galileo/documentation.
16. Guru 99. (2015). Protocol testing. Retrieved on March 5, 2017, from http://www.guru99.com/protocol-testing.html.
17. Jacob, P.M. (2018). Intrusion detection system using IoT. Retrieved on January 2, 2018, from https://iottesting.blogspot.com.
18. Adafruit. (2016). Testing a PIR. Retrieved on December 5, 2017, from https://learn.adafruit.com/pir-passive-infrared-proximity-motion-sensor/testing-a-pir.
19. Try QA. (2018). What is decision table in software testing. Retrieved on March 26, 2018, from http://istqbexamcertification.com/what-is-decision-table-in-software-testing/.
20. Jacob, P.M.; Mani, P.; and Sultana, P.H. (2018). A comparative analysis on smart farming techniques using internet of things. *Helix*, 8(2). 3294-3302.
21. Motahhir, S.; Chalh, A.; Abdelaziz, E.G.; Sebti, S.; and Derouich, A. (2017). Modeling of photovoltaic panel by using proteus. *Journal of Engineering Science and Technology (JESTEC)*, 10(2). 8-13.
22. El Hammoumi, A.; Motahhir, S.; Chalh, A.; El Ghzizal, A.; and Derouich, A. (2018). Low-cost virtual instrumentation of PV panel characteristics using excel and arduino in comparison with traditional instrumentation. *Renewables: Wind, Water and Solar*, 5, 1-16.