IoT Testing-as-a-Service: A New Dimension of Automation

Babur Hayat Malik[1], Myda Khalid[2], Maliha Maryam[3], M. Nauman Ali[4], Sheraz Yousaf[5], Mudassar Mehmood[6], Hammad Saleem[7]
Department of CS & IT
University of Lahore, Chenab Campus, Gujrat Pakistan

***Abstract*—Internet of Things (IoT) systems has become a global trend enhancing the capabilities smart computing era involving a variety of distributed end-devices and multi- scalable applications. The collaborative nature of IoT systems connected through the Internet increases the heterogeneity of coming data streams that need to be processed for correct decision making in a real-time environment. The processing of huge data streams for remotely distributed IoT systems create loops for data breaches and open new challenges for security and scalability of system testing. Thus, the testing of IoT systems is becoming the necessity, requires automated testing framework due to the amount of IoT devices and processing of data events is prone to error by traditional software testing. An automated IoT testing service based framework is purposed in this paper, to test the distributed IoT systems by reducing cost and scalability issues of software testing. The infrastructure of IoT systems demands a large number of platforms be developed which requires systematic testing approach. Therefore, the purposed automated IoT testing as a service model performs distributed interoperability testing, oneM2M based conformance testing, security testing of distributed systems and validating semantics/syntactic testing of IoT devices in a systematic approach. Lastly, to provide more strength to the work we discussed and analyze existing IoT testing models to evaluate our proposed model.**

Keywords*—Testing automation; IoT; interoperability testing; conformance testing; security testing; semantic testing

I. Introduction

The connectivity of people, objects and devices termed as Internet of Things (IoT). IoT uses sensors devices let physical objects and virtual world to conjoin their environment via the Internet [1]. The connectivity of virtual and physical over the Internet opens up new dimensions such as smart cities, smart homes, etc. the growth in IoT services and devices give rise to humongous data streams and a single device will create four times in its five-year duration and this amount will lead to more than 600 Zeta-byte data and number of IoT devices increase to 29 billion in the year 2020 [2]. IoT environment provides an opportunity for high-scalable end-devices with constrained computing and storage along with cloud integration to maintain latency sensitive systems in IoT [3].

Due to highly complex distributed computing structure, lack of communication frameworks and multiple protocols developing these type of systems is a tedious task because such complexity in the structure of IoT systems is vulnerable to unauthorized access and external attacks [4], [5]. Usage test-case and test suites for ensuring the security and interoperability of IoT system is challenging but testing IoT systems with the integration of diverse technologies and handling big data streams makes it more difficult [6]. Therefore, there is a need to implement a testing framework to ensure both conformance and interoperability along with the security of IoT systems [4]. Although, a number of researches on exploring the security of IoT has been done in the past. Only rare searches highlight the security, correctness, and completeness of both hardware and software attached remote IoT applications [7]. Mostly, the IoT system (software and hardware) has been overlooked from past years.

In this paper, we have presented a model to automate IoT testing in a real-world scenario. Firstly, we go through the previous work done on IoT interoperability and conformance testing approaches to strategies potential IoT framework for testing. Our proposed work is a model-based approach to testing as a service which is interconnected to the Internet of things (IoT) systems. A distributed cloud service based approach over the network is being adapted to facilitate the IoT system with automated testing as a service. Basically, we are extending a testing service model and using a holistic approach to debug network-related features and perform efficient testing of remote IoT systems. This model analyzes technicalities of testing IoT devices and present solutions by incorporating interoperability, conformance testing along with the security validation of IoT devices and also performing semantic and syntactic testing.

This article is organized as Section II. Background work and present the overview of automated testing. In Section III, various testing methodologies in IoT are discussed. Section IV describes our purposed model of automated IoT testing as-a-service then Section V discusses the strengths of this model. Lastly, Section VI concludes the article.

II. Background

The model-based testing is studied thoroughly in literature [8], [9]. Whereas, mostly IoT connected approaches are premeditated for mobile applications. Though, other related work emphasizes more on the liability of IoT based systems testing. Authors in [10] developed an approach named as IoT testing as a service for creating automatic test cases with the use of several patterns. Cloud consumers and cloud providers are also provided with a testing service known as Testing as a Service (TaaS) [11]. Work done on TaaS is more related to web services and cloud computing. Zech et al. in [12]

presented an approach for creating test cases with help of risk analysis of cloud computing for ensuring security.

The work in [13], described standardized interoperability testing implementation and other affecting issues like cost, scalability, and coordination. These are few issues which arise during the IoT products development and methodologies being applied such as testing methods used by the telecommunication industry aren't flexible enough to command the IoT systems. Few of them aren't able to interact with high-level protocols because of their small sizes. Interoperability issue occurs at the semantic level because the streams of data that passes on are needed to be checked at semantic and syntactic levels for the purpose of data correction if any of data is a flaw. In the past, IoT testing was used for handling this issue of interoperability from the creation and execution of test cases and testbeds for real-world IoT devices.

System security concepts are described in this paper [14], along with language-based technique which describes the process of dealing with BOF. It also describes the data structures and techniques which are used for code and memory analysis (Control Flow Graph (CFG) and Dependence Graph (DG)), (Points-To and TFA), and Memory Safety tools (Address Sanitizer and SAFE Code). Buffer overflow contains a large quantity of data to let go of the upper bound of the buffer which means overwriting of data on another. It usually occurs in the form of heap or stack. Dependence Graph (DG) is used for modeling of data and instructions reliance in the program. Tainted flow analysis is another technique for poking paths of information that moves through inputs to sensitive operations.

In [15], the RM model is being used for knowing the IoT domain along with other models for finding out IoT concepts and constraints. It also works as a base of RA. RM particularly consists of a first level IoT concepts description known as Domain Model. It also consists of an Information Model that deeply describes the processing of IoT information. RA offers key Functional Groups (FG) which is required by IoT architecture through its functional view. FG explains applications functionalities which are made on the peak of IoT infrastructure. It also offers IoT –aware demonstration which is accomplished during process execution.

A. Overview of Internet of Things (IoT) Automation Testing

Test automation in IoT is used for execution management and compares actual result with the predicted one. It also helps out in enhancing the speed of unit testing, API testing, and GUI testing. Though it also executes regression tests and its extremely economical and with the technology shift industry depends for testers with automated skills [16]. Apart from the regression test, compatibility tests are also run by it which improves productivity level and make sure that customer is provided with quality software. Although it enhances the efficiency of tests few drawbacks also exist over here. These drawbacks involve unable to enhance test potency and identifying errors. Therefore, one major drawback involves scripts automation. Mohd Ehmer Khan discussed such software testing tools which are used for testing software like performance, reliability, etc. As these, all are categorized according to their main working and in different types [17]. Researcher Manjit Kaur did a comparison between different automation tools just like QTP, TC [18]. The tools QTP Pro and QA are compared on the basis of various characteristics of cost, time and scripts creation, etc. QTP is basically more efficient in regard to those applications which requires more security whereas test complete efficiently work for those applications which require less security. Author Harpreet Kaur presented a comparative analysis of different tools like Selenium, etc. and identified their performance on the basis of cost, application support, etc. QTP [19] is compared with selenium and TC and deeply analyzed and compared according to each possible factor and considered best among all [20].

III. Testing Methodologies in Internet of Things

A. Interoperability Testing in Internet of Things (IoT)

For the assurance of network interoperability testing standard bodies of communication (ETSI and Bluetooth SIG) benchmarks some rules and processes which includes plug-test events and conformance testing. Plug-test events involve a meeting of organizations who implement technologies, each party test and check their systems against others. Such events cost high overdue of organizations and also attended by IoT communities and research centers, it also requires one developer and tester which is economically not preferable without sponsors for open-source communities [10], [21]. To test the network interoperability in IoT systems, an external IoT system (both software and hardware) is integrated by the third party service providers with the minimum code already written to apply initial functions to set the system in a stable state for interoperability testing that can be handled by the abstraction layer like resetting device or configuring network. Test cases to test the interoperability of the system are presented at the top of the abstraction layer. The challenged faced in executing this method requires both third party and already integrated communication systems to be present in the similar location which can be controlled by implementing transparent network bridges in distributed test system scenario [10]. The communication systems involving System Under Test (SUTs) need to be connected with the end point of the bridges in order to transfer communication to and from the third party systems placed in a different locality. In addition to this, to create a network bridge both the wireless transceivers and endpoint systems must be using the same distributed messaging service (e.g. IEEE 802.15.4).

1) Interoperability testing models of IoT: Various types of configuration testing are modeled to deal with the diverseness of IoT test and deployment systems controlled by testbed alliance are discussed as [13]:

- *Simple Conformance Testing:* Appropriate for testing the conformance of only one IUT at a time. It can only check the functionality of the IoT devices.
- *Simple Conformance and Interoperability Testing:* This model is suitable for both conformance and interoperability testing of only single new IUT with a standard testbed.

- *Simple Conformance and Compound Interoperability Testing:* This mode is appropriate for conformance and interoperability testing of a new IUT with a number of testbeds in the system when reference implementation is unclear.
- *Compound Conformance Testing:* This model performs conformance and interoperability testing to analyze the cooperative and collaborative behavior of several IUTs without testbeds.
- *Compound Conformance and Compound Interoperability Testing:* This model performs conformance and interoperability tests on several IUTs numbers of testbeds.

B. Conformance Testing in Internet of Things (IoT)

Conformance testing measures and ensures the implementation of a specific standard to the required level. Generally, the conformance testing model comprises of two parts, one carries out the Implementation Under Test (IUT), is System Under Test (SUT) and other is Means of Testing (MOT) which includes coordination, logging and reporting activities handled by a minimum of one tester depends on IUT's architecture and interface [13].

1) Architecture of conformance testing: There are various creation elements are used for conformance testing such as Implementation Conformance Statement (ICS), Implementation eXtra Information for Testing (IXIT), Testing Description Language (TDL), Executable Test Suite (ETS), Abstract Test Suite (ATS), and Equipment under Test (EUT) from [22]. Product Functionalities and abilities used for the purpose of checking and provision of interoperability signals are used by ICS. Extra important metadata is provided by IXIT for testing purpose. Test cases are described by formal language known as TDL. Test suites are described by another formal language known as Tree and Tabular Combined Notation version 3 (TTCN-3) and it is done by ETSI and few others like SDOs i.e. 3GPP and oneM2M. Group of test cases which shows test completion and described in a normal language like TTCN-3 is all done by ATS whereas, ETS uses TTCN compiler and its totally irreplaceable. Conformance testing isn't only used for testing of normal behaviors, rather also used for testing of extraordinary behaviors. It also enables the tester to perform broader functional testing. It doesn't completely ensure the interoperability of the system with other systems because the standard test might leave some space for configuration and conformance purposes.

C. Security Testing in Internet of Things (IoT)

The security requirements of IoT system are of extreme importance as functional requirements due to the vulnerable of security functions. However, analytics of IoT systems and security tests summarize factors of IoT systems which resist security issues such as usage of other systems, system security threats and vulnerability, and security function's exploitation. Some Model-Based Testing (MBT) Standards like M2M (Machine to Machine) and its extension oneM2M identified some security vulnerabilities and requirements level that needs to be satisfied before developing the system [23]. Three testing strategies need to be implemented in groups or discretely to check the validation of these requirements. These are as follows:

- *Security Functional testing (compliance with agreed standards/ specification):* It analyzes the system against the required functional specifications in order to ensure that the implementation of security functions is implemented in an approved manner.
- *Vulnerability testing (pattern driven):* It analyzes security attacks and risk-based vulnerabilities. It is based on security patterns used to initialize the security testing, then the targeted test patterns are used to apply appropriate test cases for possible security threats identified.
- *Security robustness testing (behavioral fuzzing):* Measuring invalid messages created by test cases in order to deal unpredicted behavior of the system for the security threat and attacks on large scale IoT systems.

Also, Model-Based Testing (MBT) approaches have used with their shown their benefits and usefulness for security testing of large-scale IoT systems undergoing particular standards defining guidelines and solutions for these security elements of the system [23].

D. Semantic Testing in Internet of Things (IoT)

Testing IoT is a level based approach in which conformance testing and interoperability testing performs protocol level testing, security testing focus on vulnerabilities in a system. While the basic purpose of semantic testing is to test the semantic accuracy of IoT data streams in accordance with the pre-defined standards [13].

The implementation of semantic testing in IoT paradigm is most challenging because of the heterogeneity of IoT devices and semantics testing performs validation in the semantic description at various targeted levels like testing lexical and syntactic validation and then logical and semantic validation. Some reference ontologies have already been defined including oneM2M ontology, W3C-SSN ontology, etc. After defining these ontologies, the next step is the conformance test against these reference ontologies to achieve semantic interoperability. Such diversity in concepts and relations of semantics models could make application of semantic interoperability more complex [24]. Some ongoing research projects like H2020 Fiesta-IoT, provide a unique cloud platform for conducting a test on semantic technologies using semantic IoT testbeds. These cloud platforms give access to semantic data of various testbeds such as smart cities, smart homes, etc., through uniquely identified access points. For analyzing the correctness of semantic and syntactic validation, data regarding particular ontology is selected from the semantic database and used for experiments against defined standard semantic. The database will reject the data in order to keep it clean and accurate if the data does fulfill all the requirements and semantics description reporting all the errors will be provided to ontology developers to model these errors while improvement phase. To complete semantic interoperability of IoT systems, achieving semantic testing is

required. The inclusion of lexical, syntactical, semantic correctness and test feature are crucial for attaining semantic testing [25].

IV. Proposed Testing-as-a-Service Model

We discussed various IoT testing methodologies in the previous section. Both interoperability and conformance are traditional testings in IoT. Also, security testing and semantic testing models are used in IoT which is a major part of IoT testing. Therefore, in this section, we integrate these testing methods used and formulate a model shown in Fig. 1, for IoT testing-as-a-service.

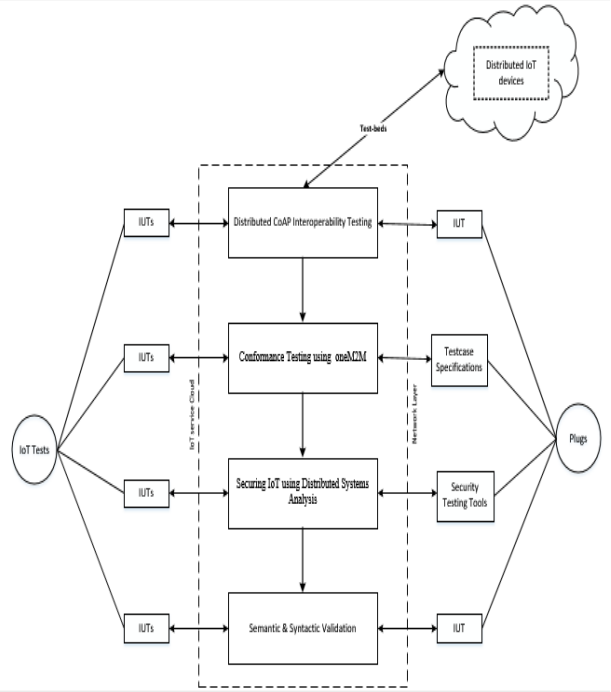

Fig 1. Proposed IoT testing-as-a-service Model.

A. Distributed Interoperability Testing for Remote Devices

The application of Interoperability testing on our model is based on a remotely distributed test system architect for automating both interoperability and conformance testing. Previously discussed network interoperability testing applies a suite of test cases but here we presented the extension of distributed interoperability testing with using distributed test plugs which will help developers and third party service providers with quick test case response from different locations. ETSI designed a Constrained Application Protocol known as CoAP for such plug-tests [26]. CoAP specifies a group of test requirements for interoperability testing and each requirement defines CoAP properties, after finalizing these requirements a test case is derived for each of them. From details of these test cases expected system behavior of CoAP protocol is analyzed [27]. Therefore passive testing methodology is appropriate for such resource constrained and operational architecture of IoT, which does not allow overheads in networks. Furthermore, to test the implementation of passive testing a message (Pass, Inconclusive or Fail) is released if a packet is captured by packet sniffer between client and server shown in Fig. 2 as:

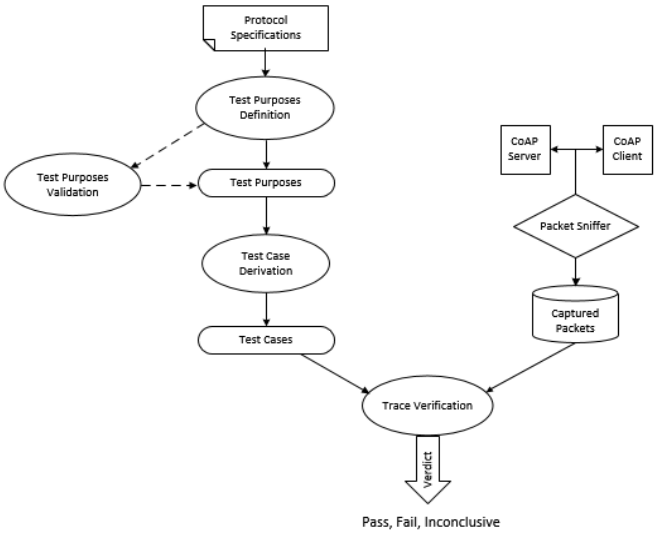

Fig 2. Architecture of CoAP Interoperability Testing.

Distributed CoAP test plugs involve two different configurations. The basic system includes TS (Test System) and SUT involving two IUTs as CoAP server and client shown in Fig. 3. However, using passive testing technique might cause capturing packets by sniffers while exchanging packets between IUTs shown in Fig. 4. Thus, distributed CoAP interoperability testing in such environment uses a UDP gateway in between CoAP server and CoAP client to replicate a lossy medium.

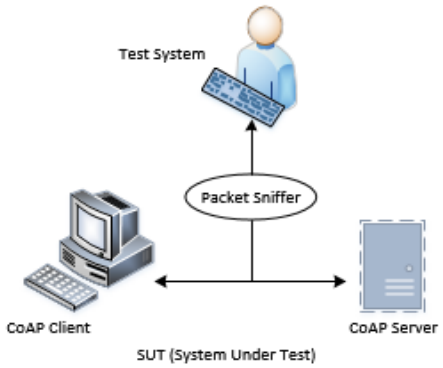

Fig 3. Basic CoAP Testing Model.

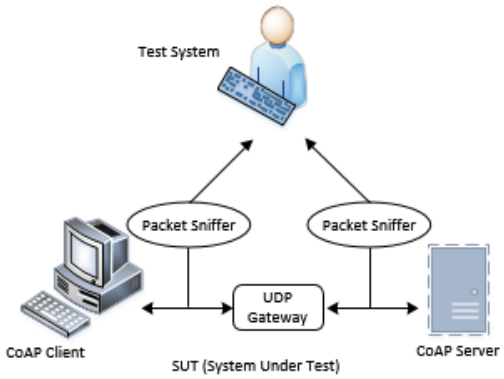

Fig 4. CoAP Passive Testing Model.

Based on the testing of some test requirements for CoAP, working model in both consistent and packet lost scenarios. The distributed CoAP interoperability testing verifies the correctness of client and server interaction involving HTTP method (GET, POST, PUT and DELETE) by analyzing each request/response by the client have correct message code [28]. On the other hand, the server sends a piggy-backed reply upon receiving a request from client such as if the request is confirmed send ACK (acknowledgment), if there is delay in getting a request the server first sends an empty ACK message then upon receiving the request it sends a confirmed response and it will send not-confirmed response for non-confirmable requests. There are some major options selected on the basis of basic transactions such as Token option analyzing any delay in request and response timing, each client request is assigned a token to synchronize the response, URI schemes are used to identify and locate CoAP resources. Here we are using URI query option in which requested resource is allotted and a correct response with accurate message code/type is sent by the server against the client's request [29].

B. Conformance Testing based on oneM2M

M2M testing framework is developed for lab-based conformance testing. In this type of testing developers and vendors have to go to labs for conformance testing purpose. Due to large SMEs (Small to medium-sized enterprises), an appropriate testing method is required whereas individual developers working generates less number of IoT devices and this M2M isn't sufficient for this purpose. Identifying low-cost IoT testing processes isn't any big deal as IoT testing has to manage many communicative standards and protocols [13], [30] creation and coverage of different protocols, logging, etc. Isn't an easy task for SMEs and developers. The automated testing attribute is used in the web-based remote testing framework. It also resolves all the occurring problems during testing whereas the main purpose of this testing is to allocate main conformance logic and provision of APIs. This provision of APIs helps testers in configuration selection according to their needs.

New IoT protocols can be chosen by web-based testing instead of enabling unknown third parties to include their protocols to the core system. For initiative test case communication triggering of the device is necessary by M2M. For helping various network protocols M2M ensures flexibility with the usage of network protocol through the UpperTester performs the previous action. UpperTester is a software which is used for converting test pointer to a message that is perceivable by IUT. IUT's ability decides the implementation of UpperTester either inwards or outwards of IUT.

Testing configuration information is required by a tester who is about to test IoT device such that selection of test cases, protocols, and devices that performs web interface. An actuated message consisting of test cases and configuration data is sent by test system to UpperTester on basis of inputs entered. On the basis of the provided guidelines, one M2M action is performed by the test system as tester passes on the message to the test system. M2M function consists of creating, retrieve, update, delete and notification. One of these M2M function is guided to the IoT device by UpperTester [13].

An agreement is required between UpperTester and IoT device for doing test procedures mentioned above. The specific operation is applied by IoT device on the basis of mentioned test cases in actuated messages when UpperTester provides test case data. After verification of conforming standard messages, test system develops findings of IoT device's conformance testing [31].

The second step of the testing model is the provision of support for managing communicative variables and also automated assistance for developing conformance tests verdicts. IoT uses different kinds of protocols as their integration is quite necessary for a testing framework. An automated IoT testing feature is being developed by us for usability and test distribution. That feature in the framework is described as follows:

1) Protocol adapter: Ascendable testing is done by various protocols of various domains and it is done by IoT devices. IoT devices need scalable testing using various protocols for different domains of application. Normal data integrity is done by TCP (Transmission Control Protocol) and request-response time is also dependent on TCP. Whereas, publish-subscribe is only used when real-time communication is required by the environment. Scalability by external code is based on IoT TaaS.

2) Automated device testing: Different applied transactions in the target device are actuated for testing IoT devices. Required action performed by developers is done by using stimulus. IoT TaaS defined various transaction and message types. It has helped IoT developers to perform automated testing by simply inserting a code.

C. Securing IoT using Distributed Systems Analysis

Securing Internet of Things (SIoT) model comprises of two parts: application independent and dependent part and executed on Top level of LLVM compiler [32]. The SIoT core is the application independent part for static analysis. The application dependent part is the SIoT instance involves libraries generated by the users for the implementation of the static analysis. The DDG graph is always created for a program which acts as a bridge between SIoT core and instance and works for each instance particularly as shown in Fig. 5.

1) The architecture of the SIoT core: SIoT uses LLVM IR a low-level language to process code files. Formed\ by bytecodes [14] (3-address instructions) use the various size of integers: bit vectors, floating point numbers, arrays, and labels. Using a group of files in this format it creates a DDG graph by undergoing a two-level process of merging and linking. In merging, multiple files are mere into one file reducing the naming conflict of files i.e. several files as the same name. this tool evaluates the network function each bytecode file which is required by merging phase to name bytecode files with the Send and Recv functions [33]. SIoT can identify these functions to add different tags to bytecode files and then merge them into a single file to ease the analysis. Whereas linking uses the recv function for the

creation of SEND and RECEIVE graph of all programs to generate DCFG. Using DCFG, Dist.-Dep graph generates DDG which can detect the vulnerabilities in the data flow.

2) Buffer overflow instance: For detecting the liabilities of BOF attacks we analyze memory and input dependencies. If data is functioned to unreliable input, then we highlight the vulnerability of the array. An unreliable input may be accessed by some malicious user with the sensors, serial ports or by the file system to use the memory access. Through DDG graph we can analyze the flow of information and detect the vulnerabilities in the program by providing DDG and input values to DistVulArrays. The LLVM pass checks various paths among memory access and unreliable inputs in the DDG. Once the analysis of program completed DistVulArrays gives these possible outputs such as true-positive rate and potential false-positive rate and number of malicious paths in the program in graph [14].

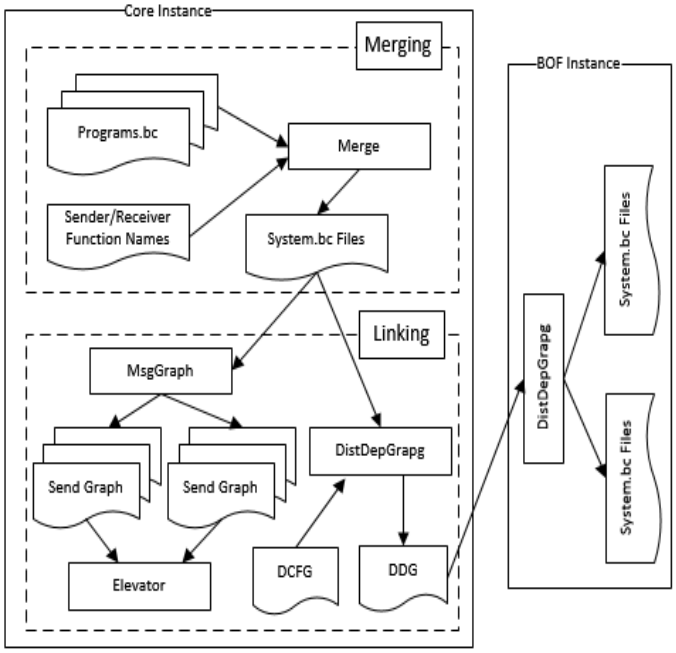

Fig 5. Architecture of SIoT.

D. Validating Semantics Testing

F-Interop project is being created for the implementation of semantic interoperability in conformance testing. F-Interop helps test systems and SUT which are placed in far areas by providing a cloud-based platform. This platform has enabled developers to work from their residences instead of moving from place to place, in this way more applicable tests are being generated in a better way in regard to time and cost. High-level testing premises are discussed in upcoming paragraphs which are applied within EU H2020 F-Interop project 11.

There are various scripts of semantic conformance test described as follows: There is fundamental interaction between the tester and SUT (System Under Test). SUT sends semantic data which is then checked by the tester whereas at end of conformance testing tester provides a report regarding completion of semantic data according to ontology acknowledgment. And if any issue occurs in that semantic data that issue is mentioned in the report. Semantic conformance testing chart is discussed in Fig. 6.

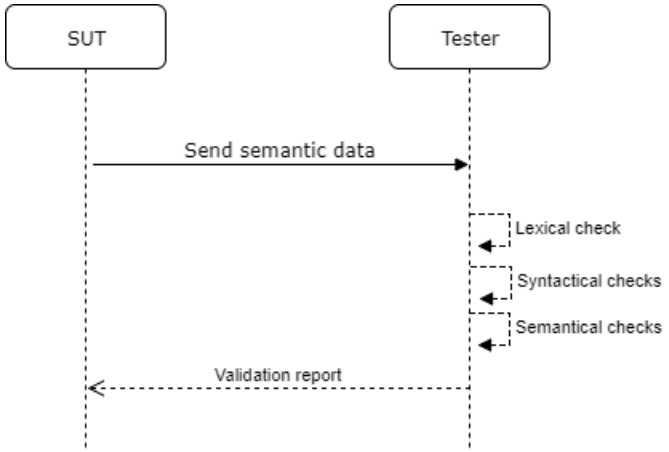

Fig 6. Basic Workflow of Semantic Testing.

There are two test scenarios of semantic interoperability. One carries SUT whereas a halfway tester is required for completion of the test in the second scenario. Here is technology agnostic that tells they aren't obliged of testing any specific platform or semantic attribute such as identifying if the semantic descriptor is generated in a proper way that it consists of M2M system's semantic data. The main purpose behind all this generation of such tests is applicable for each type of semantic data that obeys some specifications [34]. This enables test integration by applying specific standard along with all types of test which is limited to standard.

Semantic interoperability is considered as data interpretation from the system. In this premise, every portion generates a piece of semantic data processing which are as follows: semantic data and semantic query. Results of semantic processing parts are then compared if they are equal or not. Their equality shows their mutual understanding. SUT1 and SUT2 should have similar semantic queries for executing tests. At last, if a similar query is executed from similar data of both SUT1 and SUT2 it results that both SUT1 and SUT2 have the same level of understanding of data.

Interoperability performed at the data level. As discussed before interoperability is given on the basis of ontology. Hence, our purpose is to identify semantic data which is used on the basis of ontology used. Ontology is a combination of vocabulary and the relationship between vocabularies. In this test, two SUT's data submitted is checked if they have the same vocabulary which is discussed in the same ontology. If they share similar vocabulary, then it is implemented at a semantic data level because they are workable.

Transmitted semantic data (D1 and D2) produced from SUT1 and SUT2 have verified their conformance as it's a condition of test. Tester recovers D1 and D2 vocabulary and verifies if they share a similar vocabulary. If similar vocabulary is shared, then D1 and D2 are totally practical [35].

V. Discussion

The integration of cloud and Internet of Things platforms provides various services such as platform as a service, infrastructure as a service and software as a service. The distributed nature of IoT devices also requires such kind of testing service for analyzing and reconfiguring IoT application during development. In this papers, we purposed ad extension of plug-test with existing IoT testing methodologies, the framework of automation testing as a service has four phases: interoperability testing is performed using CoAP protocol to verifies the correctness of client and server interaction and analyze the request/response of target message type [28]. Conformance testing based on oneM2M use test plugs to test system specifications using test case on IUTs. Validating semantic testing used different ontologies to validate the semantic/syntactic correctness of the particular document. Furthermore, the addition of security testing in the model identifies the vulnerabilities in the system and provides a solution to increase system reliability. Therefore, this framework could allow developers to easily implement automation testing as a service to enhance correctness, reliability, and interoperability of IoT application being developed.

VI. Conclusion

The testing model presented in this paper is a service-based approach of IoT system testing which enables automated testing for distributed IoT systems by providing constraints on cost, scalability, and complexity of IoT applications. Firstly, we analyze automation testing in IoT and then, in accordance with this we presented an insight into existing methodologies of IoT testing with its design and implementations. Furthermore, we extended an existing testing concept and introduced a novelty framework to generalize testing-service in remote IoT systems. Automation IoT testing as a service model architecture incorporating four IoT testing methodologies: distributed interoperability testing, conformance testing based on oneM2M, security testing distributed systems and semantic/syntactic testing in a systematic approach. Our model creates a distributed plug-test to enables network interoperability testing without delaying data transfer from one SUT to another irrespective of location constraints. As future work, we will extend this work in order to design automation testing suites to enables the development team to analyze and enhance the security of IoT devices.

References

[1] L. Ling, M. Loper, Y. Ozkaya, A. Yasar, and E. Yigitoglu, “Machine to machine trust in the IoT era,” in 18th International Workshop on Trust in Agent Societies, Singapore, 2016.

[2] “Cisco global cloud index: Forecast and methodology, 20152020,” Tech. Rep., 2016.

[3] E. Yigitoglu, M. Mohamed, L. Liu, and H. Ludwig, “Foggy: A Framework for Continuous Automated IoT Application Deployment in Fog Computing,” 6th International Conference on AI & Mobile Services, IEEE, 2017.

[4] I. Schieferdecker, S. Kretzschmann, A. Rennoch and M. Wagner, “IoT-Testware- an Eclipse Project,” International Conference on Software Quality, Reliability and Security, IEEE, 2017.

[5] M. Ammar, G. Russello and B. Crispo, “Internet of Things: A survey on the security of IoT frameworks,” Journal of Information Security and Applications, vol. 38, pp. 8-27, 2018.

[6] M. Leotta, F. Ricca, D. Clerissi, D. Ancona, G. Delzanno, M. Ribaudo and L. Franceschini, “Towards an Acceptance Testing Approach for Internet of Things Systems,” ICWE, Springer, 2018.

[7] Z. B. Celik, P. McDaniel and G. Tan, “Soteria: Automated IoT Safety and Security Analysis,” USENIX Annual Technical Conference, 2018.

[8] M. Utting, B. Legeard, F. Bouquet, E. Fourneret, F. Peureux and A. Vernotte, “Chapter 2 - recent advances in model-based testing,” Advances in Computers 101, pp. 53–120, 2016.

[9] M. Utting, A. Pretschner and B. Legeard, “A taxonomy of model-based testing approaches,” STVR vol. 22, issue 5, pp. 297–312, 2012.

[10] P. Rosenkranz, M. Wahlisch, E. Baccelli and L. Ortmann, “A Distributed Test System Architecture for Open-source IoT Software,” Proceedings of Workshop on IoT challenges in Mobile and Industrial Systems, pp.43-48, 2015.

[11] L. M. Riungu, O. Taipale and K. Smolander “Research issues for software testing in the cloud,” 2nd International Conference CloudCom, IEEE, pp. 557–564, 2010.

[12] P. Zech, M. Felderer and R. Breu, “Towards a model-based security testing approach of cloud computing environments,” 6th International Conference SERE-C, 2012.

[13] H. Kim, A. Ahmad, J. Hwang, H. Baqa, F. Gall, M. A. R. Ortega and J. Song, “ IoT-TaaS: Towards a Prospective IoT Testing Framework,” in IEEE Access, vol. 6, 2018.

[14] F. A. Teixeira, F. M. Q. Pereira, H. Wong, J. M. S. Nogueira and L. B. Oliveira, “SIoT: Securing Internet of Things through Distributed System Analysis,” Future Generation Computer Systems, vol. 92, 2019.

[15] S. De, F. Carrez, E. Reetz, R. Tonjes and W. Wong, “Test-Enabled Architecture for IoT Service Creation and Provisioning,” Springer, 2013.

[16] K. Saravanan and E. P. C. Prasad, “Open Source Software Test Automation Tools: A Competitive Necessity,” International Journal of Management and Development, vol. 3, issue 6, pp. 103-110, 2016.

[17] M. E. Khan, “Different forms of Software testing techniques for finding errors,” International Journal of Computer Science Issues, vol. 7, issue 3, pp. 11-16, 2010.

[18] M. Kaur, and R. Kumari, “Comparative study of automated testing tools: Test complete and quick test pro,” International Journal of Computer Applications, vol. 24, issue 1, pp. 1-7, 2011.

[19] Mercury Quick Test Professional tutorial, version 8.0. Mercury Interactive Corporation, Documentation, 2004.

[20] Automated Testing: Process, Automated Testing Tutorials: What is Process, Benefits and Tools Selection. Guru99. Retrieved (2015), from http://www.guru99.com/automation-testing.html.

[21] L. Atzori, A. Iera, and G. Morabito, “The Internet of Things: A survey,” Computer Networks, vol. 54, 2010.

[22] S. Moseley, S. Randall, and A. Wiles, ``Experience within ETSI of the combined roles of conformance testing and interoperability testing," in Proc. 3rd Conf. Standardization Innovation Information Technology, pp. 177-189, 2003.

[23] A. Abbas, G. Baldini, P Cousin, S. N. Matheu, A. Skarmeta, E. Fourneret and B. Legeard, “Large Scale IoT Security Testing, Benchmarking and Certification,” Cognitive Hyperconnected Digital Transformation, Chapter: 7, 2017, pp.189-220.

[24] M. Bermudez-Edo, T. Elsaleh, I. P. Barnaghi, and K. Taylor, “A lightweight Semantic model for the Internet of Things," in Proc. Int. IEEE Conf. Ubiquitous Intell. Comput., Adv., Trusted Comput., Scalable Comput. Commun., Cloud Big Data Comput., Internet People, Smart World Congr, 2016.

[25] Linked Data_Connect Distributed Data Across theWeb. (Online), 2017 Available: http://linkeddata.org/.

[26] C. Lerche, K. Hartke, M. and Kovatsch, “Industry adoption of the internet of things: A constrained application protocol survey,” In Proceedings of the 7th International Workshop on Service Oriented Architectures in Converging Networked Environments, 2012.

[27] D. Lee, A. N. Netravali, K. K. Sabnani, B. Sugla, and A. John, “Passive testing and applications to network management,” International Conference on Network Protocols, IEEE, pp. 113-122, 1997.

[28] A. Ahmad, F. Bouquet, E. Fourneret, F. L. Gall and B. Legeard, "Model-Based Testing as a Service for IoT Platforms," ISoLA, 2016.

[29] N. Chen, C. Viho, A. Baire, X. Huang and J. Zha, "Ensuring Interoperability for Internet of Things: Experience with CoAP Protocol Testing," Journal for Control, Measurement, Electronics, Computing and Communications, vol. 54, issue 4, 2017.

[30] B. Ahlgren, M. Hidell, and E. C. H. Ngai, "Internet of Things for smart cities: Interoperability and open data," IEEE Internet Computer, vol. 20, issue 6, pp. 52-56, 2016.

[31] Functional Architecture, document oneM2M TS-0001-V2.10.0, 2016.

[32] C. Lattner and V. S. Adve, "LLVM: A compilation framework for lifelong program analysis & transformation," In: CGO, IEEE, 2004.

[33] C. Cowan, F. Wagle, C. Pu, S. Beattie and J. Walpole, "Buffer overflows: attacks and defenses for the vulnerability of the decade," DISCEX, DARPA, 2000.

[34] "Web of Tthings (WoT) thing description specification." https://github.com/_w3c/wot-thing-description.

[35] S. K. Datta, C. Bonnet, H. Baqa, M. Zhao and F. L Gall, "Approach for Semantic Interoperability Testing in Internet of Things," GIoTS, 2018.