International Journal of Electrical and Computer Engineering (IJECE)
Vol. 10, No. 3, Juni 2020, pp. 2755~2762
ISSN: 2088-8708, DOI: 10.11591/ijece.v10i3.pp2755-2762

Simulation, modelling and packet sniffing facilities for IoT: A systematic analysis

Bimal Patel, Parth Shah
Department of Information Technology, CSPIT, CHARUSAT, India

Article Info

Article history:

Received Jun 13, 2019
Revised Nov 28, 2019
Accepted Dec 9, 2019

Keywords:

Design characteristics
Internet of things (IoT)
IPv6
Packet sniffer
Quality of service
Simulation tools

ABSTRACT

Man and Machine in terms of heterogeneous devices and sensors collaborate giving birth to the Internet of Things, Internet of future. Within a short span of time 30billions intelligent devices in form of smart applications will get connected making it difficult to test and debug in terms of time and cost. Simulators play vital role in verifying application and providing security before actually deploying it in real environment. Due to constraint environment in terms of memory, computation, and energy this review paper under a single umbrella will throw insight on comprehensive and in-depth analysis keeping in mind various barriers, critical design characteristics along with the comparison of candidate simulator and packet sniffing tool. Post simulated analysis play vital role in deciding behavior of data and helping research community to satisfy quality of service parameters. This review makes it feasible to make an appropriate choice for simulators and network analyzer tool easy fulfilling needs and making IoT a reality.

Corresponding Author:

Bimal H Patel,
Department of Information Technology,
CSPIT, CHARUSAT,
Changa, 388421-India
Email: bimalpatel.it@charusat.ac.in

1. INTRODUCTION

Ever growing networks of physical object that tends to interconnect real world with digital concept in form of smartness is gaining momentum. This gives emergence to the term Internet of Things (IoT) proposed by Kevin Aston. According to Gartner report Internet of things installed base will be populated by 30 billion smart devices [1]. Any thing communication is now widespread to Internet of Peple, Internet of Content and Internet of Services with the help of IPv6 addressing. IoT enabled device will provide smart application to industry in form of Industrial IoT, agriculture, smart home, healthcare, logistics etc. Wireless sensor networks, actuators and embedded system with microcontroller and chips acting as integral part in designing smart and intelligent devices.

As we already know change is only thing which is permanent in this world, value of industrial internet in terms of different sectors will be huge due to sizing the opportunity. At a point when new concept start evolving in researcher's mind actually it is a journey which leads to result and experiments. Novel ideas often require simulation tools to develop into fully grown solutions deployable in target environment. The intricate networks obtained by IoT devices are difficult to design and modelled. Smart device connectivity is influenced by various factors like geological area, cooperation, scale, network capabilities, communication and interference. It is evident that simulation tools often tends to make system less complicated, faster and able to test quality of service parameters before deploying it in real world.

As we already know IoT as connectivity act as an enabler but the real value lies in large chunks of data generated. Packet analyzer tools not only converts data into meaningful information but also helps to achieve wisdom level. Network monitoring tools helps to fix issues faster, stay ahead of outages, identify

security threats, manage growing and changing networks.Collected data helps to identify congested links and by predictive analysis it may improve quality of service parameters.

Motivation and highlights, To best of my knowledge this will be first research paper that provides comprehensive details analysis of simulators used along with its usage in terms of research paper publication for Internet of Things environment. Along with simulator under single umbrella it discusses packet analyzer/sniffing/network diagnostic tools which play vital role in evaluating IPv6 parameter effecting quality of service. Key highlights of this review are as follows:

- Considering various challenges for simulating IoT environment.
- Provides design characteristics necessary for consideration of simulators.
- Candidate simulators discussed and compared based on state of art efforts.
- Choosing right simulators based on merits and demerits along with comparison of additional features.
- Widely used packet analyzer which enhance traffic evaluation for IPv6 network is discussed.
- Comparison of network analyzer tools based on core and additional features in form of overall table.

Paper Organization, The rest of paper is organized as follow: Section II described major concern and challenges for IoT simulation environment. This is followed by state of art survey related to widely available simulators by comparing main features along with merits and demerits for IoT in Section III. In Section IV commonly used sniffing tools for IPv6 network along with key highlights is discussed. In Section V conclusion is presented which will give clear idea of candidate simulator and network analyzer under varying IoT environment.

2. MAJOR CHARACTERISTICS FOR CONSIDERING IOT SIMULATION TOOLS

Simulation tool play vital role in imitating behavior of real world without building it. It helps to find unexpected error by analyzing various quality of service parameters before actually deploying it. In this section we try to consider various desirable characteristics for considering simulation tool in context of IoT. Finally we try to compare it with mostly used IoT simulation tools.

2.1. Heterogenity

By 2025 75billions smart diversified devices will get interconnected, revolutionizing Internet of Things in an unthinkable way. To analyze the behavior of these smart things through simulator before actually deploying it heterogeneity plays major role. Simulators should be able to cope with heterogeneity in terms of devices, technologies, environment and application. Simulators should allow a network to disseminate information across heterogeneous environment by applying different technologies and giving realistic results [2-3].

2.2. Scalability

IPv6 provide scalable solutions in form of addressability due to which IoT is able to expand vastly. Simulators for IoT need to take care of vertical scalability i.e. scaling up processing power, storage and network interface and also horizontal scalability i.e. adding more machines/servers, adding more nodes. While providing scalability its need to take care that performance (in terms of quality of service parameters) does not get degrade [2-3].

2.3. GUI/debugging/trace support

Simulators of IoT can be either Command line interface (CLI) or Graphical user interface (GUI) driven. GUI driven simulator facility should provide debugging support i.e. identify, watch, detech and resolve unexpected behavior and visualization support in terms of speed (play, pause etc.) and plotting of graphs to actually compare results for quick review. Simulators should have capability to inspect modules, monitor queue types and have post processing trace support [2-3].

2.4. Scripting language

Scripting language plays vital role in controlling simulation while performing distinct experiments. Scripting language should be interactive in nature i.e. user don't have to read manuals and explore, easy to learn and accessible to people of all genre not only programmers. Scripting language should support powerful data structure and provide not only standard library support but also extensive third party library support. Scripting language should be able to run on scarce resources i.e. low power and low memory which is core characteristics of IoT. Since IoT produce larger chunks of data scripting language should be able to process output data precisely and in quick manner [2-3].

2.5. Reusability/availability

Generally simulation are useful to analyze, compare and test existing techniques with novel or proposed approach in realistic and controlled scenarios. Key aspect to decide simulator is whether it includes implementation of full IoT protocol stack along with sensor network protocols. Other factor should be whether user are able to build/add or change new functionality easily or not. Simulator should also provide various testbed implementation and propagation model to full extent. To predict accurate results user need to perform repeat experiments under varying parameters, it is essential that simulator produce reliable results in repeat scenarios [2-3].

2.6. Extensibility

Simulator should be active in terms of development, debugging support and upgradation in terms of versioning. Simulators should provide detailed support/technical document, have valid website with current and previous versions details and comparison. Along with different versions there should be active blogs, tutorials and video session available for simulators to ease the learning and usability part [3]. Along with core characteristics mentioned above simulators should also focus on some non-functional characteristics like portability [3] (in terms of programming code and operation system), cross level simulation support [3] and real time capabilities.

3. STATE OF ART EFFORTS

It is always difficult to evaluate performance by deploying it in real-time environ-ment directly due to factors like time, cost space and efforts. Simulators model real world environment by predicting result in terms of quality of service parameters. In this section different popular and widely used open source/ commercial simulation tools for IoT is explored in detail.

3.1. Cooja

Cooja which is created by Adam Dunkels is flexible, extensible, discrete-event based and cross-level simulator included as a part of Instant Contiki which concentrate mainly on wireless sensor network behavior [4]. Cooja is written in Java Language and uses Java native interface (JNI) calls for running and interaction of all Contiki application programs. Cooja supports three types of nodes simulated at three different abstraction levels. The highest level of abstraction is called network or application level where user can focus on implementing application related to networking, routing protocol and distributed concepts using Java nodes. Developed, deployable and debugging code is executed at operating system level with the help of native nodes. Final level of abstraction is machine-code instruction level which focus on emulating CPU and other hardware components behavior using MSPSim emulator [5] with the help of emulated ESP nodes and microcontroller. Cooja supports set of standards like TR 1100 and TI CC2420 while protocols like RPL [6] and stack (uIPv4 and uIPv6) is also supported. For simulation purpose cooja supports four different propagation model.i.e. UDGM (Unit disk graph medium) constant loss, UDGM distance loss, DGRM (Directed graph radio medium) and MRM (Multipath ray tracer medium) [4-5].

3.2. Tossim

Tossim is open source, discrete event-driven simulator included as a part of TinyOS operating system possessing specially designed requirements like scalability, fedility (i.e. network behavior), completeness and bridging (between test implementation and real world scenarios) [7]. Currently Tossim provides bit level network simulation to MicaZ motes only which can be one of the disadvantage. Simulation code is written in “nesC” [7], a dialect of “C” language which provides advantages like safety of code through component level design, parameterialized interface for optimization and flexible concurrency model. Issues related to energy/power consumption is resolved by Powertossim [8] which is further extension of Tossim. Powertossim helps to analyze energy consumption state of each node along with CPU profiling support. Visualization (i.e. visualize, monitor, control and debug) environment is provided by java based GUI tool called TinyViz.

3.3. Avrora

Avrora is an open source, flexible, portable (in terms of language and operation system), and cycle accurate discrete-event simulator tool specially designed for AVR micro-controllers supporting Mica2 and MicaZ platforms developed as a research project by “UCLA (University of Califonia, Los Angeles)” compiler group [9]. Simulation code is written in java language and it operates on instruction level with one thread per node it tends to achieve higher speed (20 times faster) and same level of accuracy with higher no of simulating nodes (up to 10000 nodes simultaneously). Avrora provides monitoring and profiling utilities

tool which helps to study program behavior by setting breakpoints and timeout. Control flow graph tool that provides graphical representation of instruction flow and energy analyzer tool which examines power consumption along with battery life. Avrora lacks integrated GUI and everything needs to be performed manually which can be one of the disadvantage [9].

3.4. NS3

NS3 is free, open-source, modular discrete-event driven network simulator licensed under "GNU GPLv2 license" with an objective to do research and development for communication networks which eventually is main motive of IoT [10]. The core part of simulation is written in C++ while optional bindings is provided with the help of Python scripting interface. Visualization and animation environment for network scenarios is provided by Netanim [11] with the help of XML trace files generated during simulation. Another live visualization tool provided by NS3 is PyViz [12] (Python based network visualization tool) specially designed for debugging purpose. NS3 provides fully configurable integration tool called Bake [13] for automatic handling of reproducible dependencies build from other projects and untrusted/unrelated parties.

3.5. Netsim

Netsim is commercial, robust, versatile and flexible discrete-event driven network simulator developed by "Tetcos, in association with Indian Institute of science" in June 2002 [14]. Commercially Netsim available in three versions [15]. 1) Academic for educational purpose 2) Standard for research and development purpose and 3) Professional for defense and enterprise solutions based on requirement. Wide range of network technologies like cognitive radio, 802.11, IPv4 stack and latest technologies like IoT, MANET, WSN, Enhanced TCP versions is also supported by Netsim. Netsim provides packet animator to visualize data, control and error packet flow (in terms of different colors) in both wired and wireless networks. Packet and event trace files can be easily exported to excel and pivot tables to examine performance metrics in terms of various quality of service parameters. Netsim also provides facility of interfacing other external tools like MATLAB, SUMO and wireshark with itself [14].

3.6. Glomosim/qualnet

GloMoSim is scalable, layered, extensible and parallel discrete event simulator developed at "Parallel Computing lab, UCLA" written in C language with support of Parsec simulation language [16-17]. GloMoSim 2.0 is the final version available and after that in 2000 Scalable technologies came up with derivate product called Qualnet [18]. Qualnet is commercial, layered, providing real time support of communication networks and possessing high fidelity property (by running on cluster, multi-core and multiprocessing system). Its again discrete event driven simulator written in C++ with a support of Parsec [19] simulation language. It provides support to various hardware, software and other third party visualization tools. Qualnet consist of five components to carry out various tasks. 1) Qualnet Architect which is design and visualization tool works in design mode (for setting up network connections/models, forming subnets and providing mobility and other functional parameters to nodes) and Visualize mode (for analyzing network performance). 2) Qualnet Analyzer which is graphical tool for forming graphs using quality of service parameters and can be easily exportable in excel spreadsheets. 3) Qualnet Packet tracer which is another graphical tool to represent trace files (in form of text files in XML format) in pictorial/visualization format. 4) Qualnet editor and 5) Command line interface to interact with simulator [18]. Detailed comparison along with its advantages/disadvantages and other miscellaneous characteristics is described in Table 1 [3, 20] while the common characteristics (type, scale, language support) is shown in Table 2 [21-22]. Finally we compare widely used simulation tools and required design characteristics in Figure 1 [23].

Table 1. Comparison in terms of merits/demerits

Simulators Emulators	Operating System	Latest Version	Merit	Demerit
Tossim (Part of Tiny OS)	Linux	2.1.2	- Bit level simulation	- Difficult to evaluate heterogeneous applications - Supports only single hardware platform i.e. MicaZ
Cooja/MSP Sim (Part of Contiki OS)	Linux	3.0	- Cross level Simulator	- Difficult to execute time dependent simulation
Avrora	Linux	Beta 1.7.106	- Provides much larger scalability and higher accuracy (up to 10000 nodes with accuracy)	- Lacks integrated GUI - Does not model clock drift

Table 1. Comparison in terms of merits/demerits (*continue*)

Simulators Emulators	Operating System	Latest Version	Merit	Demerit
NS-3	FreeBSDLinux/ Windows (Through Cygwin)	NS-3.28	- Highly Modular - Code can be used in real implementation	- Lack of credibility - Python bindings doesn't work with Cygwin
Qualnet	Windows, Centos, Linux	Qualnet 7.4	- Real time communication networks support - Convenient and easy to use GUI	- Commercial
Netsim	Windows	Netsim V11	- Best in class GUI and visualization support - Result Dashboard and Inbuilt graphing	- Freely not available - Only single process and single event queue

Table 2. Comparison of core features

Simulators Emulators	Simulator Type	Language Supported	License	Mobility	Scalability (>100 nodes)
Tossim (Part of Tiny OS)	Discrete-Event	NesC, Python, C++	Open Source	Yes	1000 nodes
Cooja, MSPSim (Part of Contiki OS)	Discrete-Event	Java	Open Source	Yes	200-500 nodes
Avrora	Discrete-Event	Java	Open Source	Yes	10000 nodes at a time with equivalent accuracy
NS-3	Discrete-Event	C++, Python	Open Source	Yes	Up to 400 nodes for good result
Qualnet	Discrete-Event	C++, Parsec	Commercial	Yes	Support 500-20,000 nodes
Netsim	Event-Trace	C, Java	Commercial	Yes	Academic (100), Standard (500), Pro version (100,000)

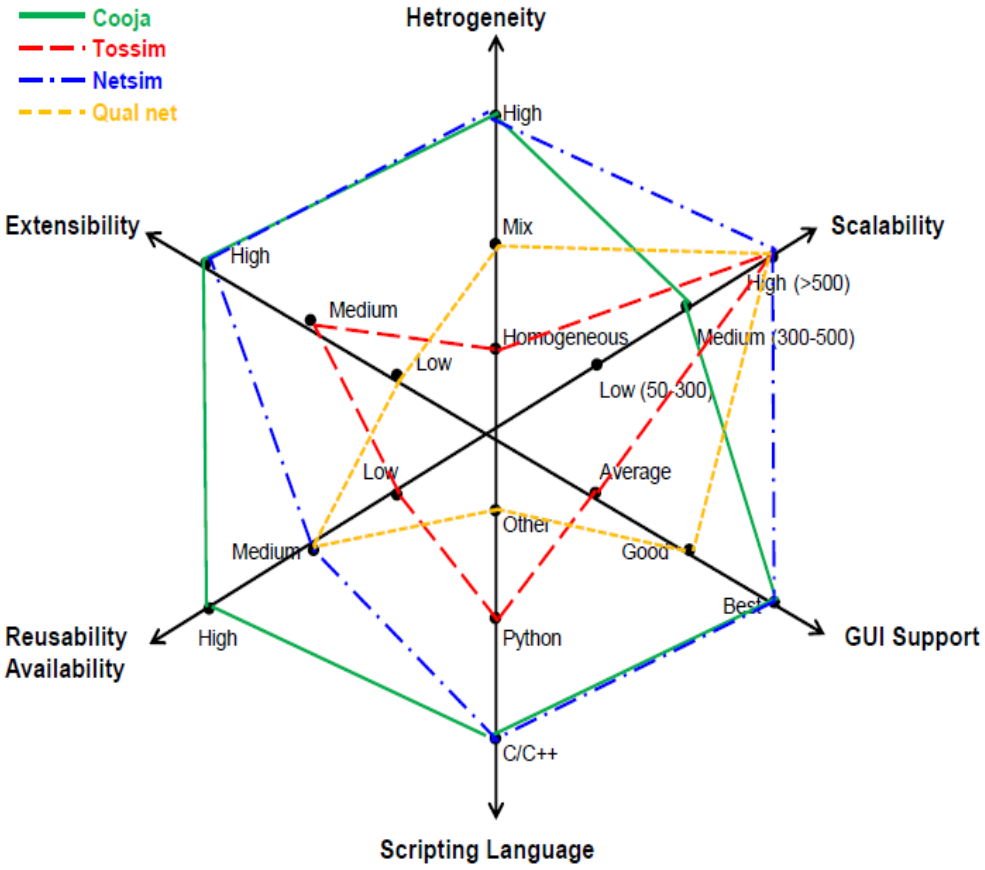

Figure 1. Comparing widely used simulators with design characteristics

4. PACKET ANALYZER TOOLS FOR IPV6 NEWTORKS

In this section widely used packet sniffer and network analyzer tools for IPv6 environment is discussed. Packet sniffer tools helps to collect and analyze data for future prediction of peak traffic and identifying congested links and abnormal behavior in networks so that further remedial action can be taken.

4.1. Wireshark

Wireshark which is formarly known as Ethereal is "widely used, popular, well designed, versatile, free, opensource, easy to use and multiplatform support packet sniffing and network analyzer tool developed by Gerand Combs in 1997"[24].Capturing of data from other tools can also analysed if the file format is in PCAP.It also provide customized reports which can be further exported in plaintext, CSV, XML and postscript.Wireshark can be learned easily due to its large community based support online and there are lot of blogs available to provide valuable information of it [25].

4.2. Foren6

It is "free,open source passive sniffing tool for 6lowpan networks published under GPL licence maintained by Laurent Deru and Sebastial Dawans from CETIC's Embedded & Communication Systems team".Foren6 provides multiple sniffers to capture traffic and reconstruct a visual and textual representation of network information to support realworld IoT based application.It provides rewind/replay facility for packet capture scenarios and provides navigation through different overlays to identify problems.In future mobile sniffer will be available with the help of android port to analyze 6lowpan network on tablet [26].

4.3. TCPDump

It is "open source, free and oldest command line snipping tool developed by Van Jacobson and team at Lawrence Berkeley National laboratory in 1990".It comes with pre-installed support on Unix and on windows in form of Windump.Output/Log files are difficult to understand by user except basic information like timestamp and packet header information.Due to its limited IPv6 support currently its preferable for TCP/IP packets only [27]. Detailed comparsion along with key highlights is shown in Table 3.

Table 3. Comparison of main features

Sniffing Tools	Compatible OS	User Interface	Cost/License	Space	Highlights
Wireshark	Windows Linux Solaris macOS FreeBSD NetBSD	GUI CLI	Free Opensource	~15MB	Offline and live capturing, Customize report facilities, Triggering and alerts
TCPDUMP	Windows Linux Solaris macOS FreeBSD NetBSD	CLI	Free	~1 MB	Less intrutive, Not fully complaint for IPv6 format,
Foren6	Linux macOS X	GUI	Free	-	Passive diagnosis tool, Best GUI and display of visual information of network state

Considering the database of last 2 years research paper published and idea considered from [28] above Figure 2 clearly shows that cooja is most popular simulation tool used along with Contiki operating system and covers almost 60-65% of research work done in field of IoT while tossim along with powertossim covers around 15% with the help of TinyOS.Netsim is commercial simulation tool mostly used by educational institution.Other network simulators like avrora, j-sim and Omnet++ even used for wireless sensor networks which is part of IoT roughly covers 15-20% usage. Figure 3 shows usage of network monitoring tools in terms of research paper published and market share covered and clearly it shows that wireshark is leader in this category and covers roughly around 60-70%.TCPDump covers roughly 15-20% market share followed by Foren6 which is best analysis tool for RPL protocol.All network diagnostics tools require PCAP file generated individually for input purpose to analyze quality of service parameters.

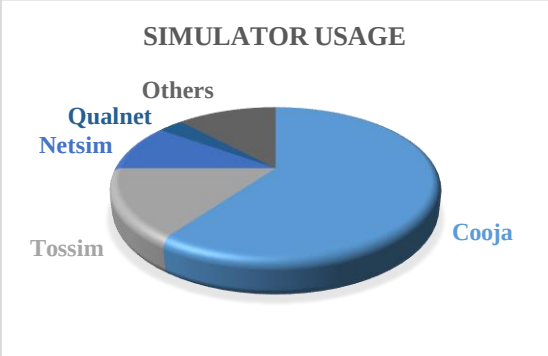

Figure 2. Widely used simulator usage

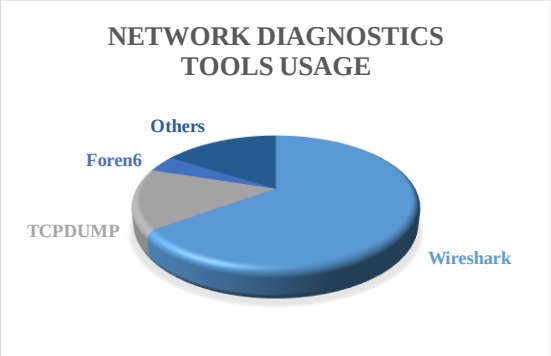

Figure 3. Widely used Diagnostics tools usage

5. CONCLUSION

In this review paper we have started with analyzing various design criteria required for full filling simulation environment for IoT. Widely used free and commercial IoT simulation tools is than presented with core and additional features in detail. Cooja and Tossim used apparently with Contiki and TinyOS is currently most popular and widely used tool due to its best in class GUI and support of post simulated packet analysis features.NS3 is used mostly for MANET and wireless sensor network protocol but lack IoT support.Avrora is specially designed for MICA family.Qualnet and Netsim are commercial simulation tools used for educational as well as research purpose mostly used in schools and colleges.Netsim current version has fully complied IoT protocol stack implementation support.Overall simuation tools can be chosen as per individual need and based on real time support.Packet analysis post simulation play vital role in providing security and satisfying quality of service parameters.Wireshark is most widely used and preferable packet sniffing tool which is easy to learn and also provide filtering and customized report facilty.Foren6 is best tool for analysis of RPL protocol for IoT and also has very good GUI. Only intention to consider details survey is to make easy for research community to choose appropriate simulator ansd network analyzer tool suitable for future demands of IoT.

REFERENCES

[1] "Newsroom," *Gartner*, 2019. [Online]. Available: https://www.gartner.com/newsroom/id/2636073. [Accessed: 12- Jan- 2019].

[2] E. Egea-Lopez, J. Vales-Alonso, A. Martinez-Sala, P. Pavon-Marino and J. García-Haro, "Simulation tools for wireless sensor networks," in *Summer Simulation Multiconference, SPECTS*, pp. 2-9, 2005.

[3] H. Sundani, H. Li, V. Devabhaktuni, M. Alam and P. Bhattacharya, "Wireless sensor network simulators a survey and comparisons," *International Journal of Computer Networks*, vol. 2, no. 5, pp. 249-65, 2011.

[4] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne and T. Voigt, "Cross-level sensor network simulation with cooja," in *31st IEEE conference on Local computer networks*, pp. 641-648, 2014.

[5] J. Eriksson, A. Dunkels, N. Finne, F. Osterlind and T. Voigt, "Mspsim–an extensible simulator for msp430-equipped sensor boards," in *Proceedings of the European Conference on Wireless Sensor Networks (EWSN)*, 2007.

[6] T. Winter, *et al.*, "RPL: IPv6 routing protocol for low-power and lossy networks," Standards Track, 2012.

[7] P. Levis, N. Lee, M. Welsh and D. Culler, "TOSSIM: Accurate and scalable simulation of entire TinyOS applications," in *Proceedings of the 1st international conference on Embedded networked sensor systems*, pp. 126-137, 2003.

[8] V. Shnayder, M. Hempstead, B. Chen and M. Welsh, "Powertossim: Efficient power simulation for tinyos applications," 2014.

[9] B. Titzer, D. Lee and J. Palsberg, "Avrora: Scalable sensor network simulation with precise timing," in *Fourth International Symposium on Information Processing in Sensor Networks*, pp. 477-482, 2005.

[10] "NS3 Tutorial," *Nsnam.org*, 2019. [Online]. Available: https://www.nsnam.org/docs/release/3.29/tutorial/singlehtml/index.html. [Accessed: 12- Feb- 2019].

[11] G. Riley and J. Abraham, "NetAnim," [Online]. Available: http://www.nsnam.org/wiki/index.php/NetAnim. [Accessed: 12- Feb- 2019].

[12] "PyViz – Nsnam," *Nsnam.org*. [Online]. Available: https://www.nsnam.org/wiki/PyViz. [Accessed: 12-Feb-2019].

[13] D. Camara, "Bake: Main Page View," [Online]. Available: http://www-sop.inria.fr/members/Daniel.Camara/bake/. [Accessed: 12- Feb- 2019].

[14] T. Veith, J. Kobza and C. Koelling, "Netsim: Java™-based simulation for the World Wide Web," *Computers & Operations Research*, vol. 26, no. 6, pp. 607-621, 1999. [Online]. Available: 10.1016/s0305-0548(98)00039-2.

[15] D. McGrath, D. Hill, A. Hunt, M. Ryan and T. Smith, "NetSim: a distributed network simulation to support cyber exercises," in *Huntsville Simulation Conference, Huntsville, AL*, 2004.
[16] X. Zeng, R. Bagrodia and M. Gerla, "GloMoSim," *ACM SIGSIM Simulation Digest*, vol. 28, no. 1, pp. 154-161, 1998. [Online]. Available: 10.1145/278009.278027.
[17] L. Bajaj, M. Takai, R. Ahuja, K. Tang, R. Bagrodia and M. Gerla, "Glomosim: A scalable network simulation environment," in *UCLA computer science department technical report*, pp. 213, 1999.
[18] "QualNet," *En.wikipedia.org*, 2019. [Online]. Available: https://en.wikipedia.org/wiki/QualNet. [Accessed: 12-Mar-2019].
[19] R. Bagrodia *et al*., "Parsec: a parallel simulation environment for complex systems," *Computer*, vol. 31, no. 10, pp. 77-85, 1998. [Online]. Available: 10.1109/2.722293.
[20] I. Minakov, R. Passerone, A. Rizzardi and S. Sicari, "A Comparative Study of Recent Wireless Sensor Network Simulators," *ACM Transactions on Sensor Networks*, vol. 12, no. 3, pp. 1-39, 2016. [Online] Available: 10.1145/2903144.
[21] M. Kabir, S. Islam, M. Hossain and S. Hossain, "Detail comparison of network simulators," *International Journal of Scientific & Engineering Research*, vol. 5, no. 10, pp. 203-18, 2014.
[22] M. Chernyshev, Z. Baig, O. Bello and S. Zeadally, "Internet of Things (IoT): Research, Simulators, and Testbeds," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1637-1647, 2018.
[23] A. Tonneau, N. Mitton and J. Vandaele, "A survey on (mobile) wireless sensor network experimentation testbeds," in *International Conference on Distributed Computing in Sensor Systems (DCOSS)*, pp. 263-268, 2014.
[24] A. Orebaugh, G. Ramirez and J. Beale, "Wireshark & Ethereal network protocol analyzer toolkit," 2006.
[25] L. Chappell and G. Combs, "Wireshark network analysis: the official Wireshark certified network analyst study guide," in *Protocol Analysis Institute, Chappell University*, 2010.
[26] S. Dawans and L. Deru, "Foren 6 a RPL/6LoWPAN Diagnosis Tool," in *EWSN 2014: Posters and Demos*, pp. 45, 2014.
[27] V. Jacobson, C. Leres and S. McCanne, "The tcpdump manual page," Lawrence Berkeley Laboratory, Berkeley, CA, pp. 143, 1989.
[28] B. Pourghebleh and V. Hayyolalam, "A comprehensive and systematic review of the load balancing mechanisms in the Internet of Things," *Cluster Computing*, 2019. [Online]. Available: 10.1007/s10586-019-02950-0.

BIOGRAPHIES OF AUTHORS

Bimal Patel received B.E. in Information Technology from Charotar Institute of Technology Changa, Gujarat University, India, in 2000. He received M.Tech in Information Technology from prestigious L.D. College of Engineering, Ahmedabad, Gujarat Technological University in 2013 with gold medal. He has guided more than 3 students at master level and published more than 7 papers in Journal and International Conferences. His research interest include MANET, Wireless Sensor Networks and Internet of Things. Currently a Ph.D. student at Charotar University of Science and Technology, Changa, India.

Parth Shah has obtained Ph.D. in the area of Security in Cloud Computing from CHARUSAT, Changa, Gujarat. He has more than 14 years of teaching experience. Currently, he is working as Associate Professor at Department of Information & Technology, CHARUSAT, Changa, Gujarat. His research interest includes High-Performance Computer Architecture and Information Security. He has guided more than 30 ME/M Tech dissertation. He has published more than 40 papers in Journal and conference proceedings. Currently, he is head of IT department in CSPIT, CHARUSAT. He has received grants of 134900.00 from GUJCOST, MHRD, and CSI.