Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 56 (2015) 357 – 362

Procedia Computer Science

The 10th International Conference on Future Networks and Communications (FNC 2015)

Building a simulation-in-the-loop sensor data testbed for cloud-enabled pervasive applications

Antti Iivari*, Jussi Ronkainen

VTT Technical Research Centre of Finland, Kaitoväylä 1, Oulu FI-90571, Finland

Abstract

When experimenting with pervasive systems consisting of numerous intercommunicating machines and sensors, it quickly becomes desirable to mix simulated virtual nodes and real world machines. This will enable the implementation of a test-bed where a large number of simulated nodes, instead of a potentially very expensive set of real prototype devices, are interacting with a smaller number of physical devices through real world physical network hardware. Such setups are referred to as "simulation-in-the-loop" systems, where it is possible to send and receive simulator generated packets on real world devices. This approach will also inherently ensure a certain degree of validity and realism of the simulation models as packets generated within the simulator must travel through a real network and be successfully received by a physical machine which, ideally, has no way of knowing if the packet originated from a simulated node or a real one. In this paper, we are presenting the design principles for an interconnected test-bed system for trialing the data handling capabilities of cloud-enabled sensor systems.

Peer-review under responsibility of the Conference Program Chairs

Keywords: sensors; iot, m2m; cloud; networks; big data; testbed; pervasive; simulations

1. Introduction

The implications of the recent advances in technologies related to pervasive sensor networks and the Internet of Things (IoT) are not only limited to the embedded technologies, wireless communication protocols and small devices that are becoming more and more ubiquitous in various aspects of our everyday lives and businesses.

* Corresponding author. Tel.: +358 40 833 5480; fax: +358 20 722 7001
E-mail address: antti.iivari@vtt.fi

Peer-review under responsibility of the Conference Program Chairs
doi:10.1016/j.procs.2015.07.219

Indeed, vast amounts of data are being generated each second by myriads of interconnected communicating devices and sensors already today, while an increasing number of small embedded devices of various kinds are being brought forth by the rise of the current IoT sensor revolution. To fully realize the potential benefits of these sensor systems, one key aspect will be harnessing the value of the information contained in the rapid, constantly incoming streams of machine-generated data. In order to meet these challenges and this vision to become a reality, the various kinds of sensor data coming in from the pervasive sensor applications must be managed with tools and processes that are arising from the fields of big data and cloud computing. This processing challenge of sensor data streams has been referred to as the "Analytics of Things"[1]. Scalable NoSQL databases[2], distributed data processing engines and efficient tools for handling the huge number of streaming events and messages are but some examples of the necessary technologies for managing this ever growing onslaught of machine-generated big data. However, even before one can get to the point of actually being able to run complex analytics or machine learning algorithms on this data, there are many challenges that must be overcome to get the raw machine data marshaled and packaged in usable formats. There is still a wealth of untapped potential and technical challenges to be overcome in managing and processing machine-generated big data[3] on such a scale.

When studying these complex software intensive systems consisting of numerous intercommunicating sensors and machines it quickly becomes desirable to mix simulated virtual nodes with real world devices. This will enable the design and implementation of a testbed where a large number of simulated nodes, in place of a potentially very expensive set of hundreds of real prototype devices, are communicating and interacting with physical network hardware and a much smaller number of physical real world sensor devices. Such setups are referred to as "simulation-in-the-loop" systems[4], where it is possible to send and receive simulator generated packets on real world network devices. This approach will also inherently ensure a certain degree of validity and realism pertaining to the simulation models being used due to the fact that the traffic generated within the simulator must travel through a real network. In this paper, we present the design for an interconnected testbed system to model the traffic and data generated by pervasive sensor systems and the Internet of Things for the purpose of trialing and experimenting with the data processing capabilities of data sinks such as different cloud-enabled back-end solutions.

2. Sensor data formats and protocols for cloud-enabled IoT applications

In order to extract value and insights from pervasive sensors and data streaming in from the Internet of Things (IoT), we must be able to collect and transfer any incoming data from those small devices or, in some cases, gateways to be stored and processed in in the cloud, typically with tools and technologies such as those based on Apache Hadoop[5]. While human generated big data is already being analysed and exploited with remarkable success by businesses such as Twitter, Facebook, LinkedIn[6,7] to name a few, harnessing the untapped potential of machine-generated big data is still a field largely left untouched[3,8,9].

The term machine-generated data has been defined by Curt Monash[10] as follows: "data that was produced entirely by machines or data that is more about observing humans than recording their choices". To get some perception on the importance and scope of this kind of data, it has been estimated by IDC[3] that the amount of machine-generated data will increase to 42 percent of all data by 2020. Furthermore, according to Gartner[11] Internet of Things, excluding PCs, tablets and smartphones, will grow to 26 billion units of installed devices in 2020 thus representing an almost 30-fold increase from the 0.9 billion of 2009. In any case, it is clear that as the number of interconnected devices and sensors embedded into our everyday lives keeps increasing, so does the total amount of machine data being generated every moment.

To make this huge amount of data available to be used and processed by cloud-enabled applications, one must first understand the data and protocols behind it. Therefore, the major data formats, communication protocols and characteristics of sensor data and data communications in the internet of things must be studied. In this work, the focus will be kept on the data streaming in from sensors, IoT, pervasive environments or communication gateways[12]. Technical details on how and if the sensors communicate among themselves, with other nearby sensors, limited area networks or personal area networks via peer-to-peer technologies[13], mobile Ad hoc networks[14,15], wireless sensor networks[16], or other short-range low power wireless protocols[17] are left out of scope in this research.

Sensor systems and the IoT are still a complex labyrinthine medley of competing protocols, platforms, data formats and technologies[18,19]. In order to future-proof the design of the sensor data testbed being built, it is important

to focus on and support the most promising and widely accepted sensor data protocols. Currently, the most essential sensor application and data protocols[20,21] can be narrowed down to the following three:

- Message Queuing Telemetry Transport: MQTT [22] is a lightweight publish/subscribe based message protocol especially well-suited for running on limited computational power and lean network connectivity. IBM and systems provider Eurotech were the first to develop MQTT before contributing the protocol to OASIS for standardisation. The protocol is already widely used the field within a wide variety of embedded applications.
- Constrained Application Protocol: CoAP[23] aims to be a generic web protocol for the special requirements of constrained sensor environments while easily integrating with HTTP and existing web technologies with a very low overhead. The CoAP protocol is designed specifically for machine-to-machine (M2M) applications such as smart energy and smart building- and energy automation. It also offers features desirable for sensor applications such as built-in discovery, multicast support, and asynchronous message exchange functionality.
- Hypertext Transfer Protocol: HTTP[24], the tried and true RESTful HTTP over TCP[25], very familiar to us all from the web service world, is particularly attractive due to the universal availability and compatibility of the legacy HTTP-stack. The RESTful HTTP approach has found success in several smaller scale application scenarios and is also employed in some M2M-platforms[26,27].

There are many others[19], of course, but these three major protocols will be the main focus in this work. Technologies related to the Representational State Transfer (REST) software architectural style, such as CoAP and HTTP, should be given special attention, as RESTful architectures are a very common approach employed in many contemporary IoT-platforms[20,28]. Furthermore, employing RESTful APIs will also give the advantage of easy integration with other web services[29].

When studying the data formats being widely used for sensor data in pervasive systems, we seem to find that, again, a handful of major ones can be easily identified to be the most essential ones. The data formats which are most commonly employed in most of the current existing sensor platform solutions[20] are the following three:

- JSON[30] (JavaScript Object Notation) lightweight data-interchange format for storing and exchanging data. It is easy to read and write by both humans and machines. While JSON is a text-based format and language independent, it is a subset of the JavaScript programming language.
- XML[31]:Extensible Markup Language (XML) is essentially a set of rules for encoding documents in a format which is readable for both man and machine. XML also acts as a basis for certain messaging protocols, such as XMPP[32] and BitXML.
- CSV[33]: Basic comma separated values are by far the simplest and most rudimentary of commonly used sensor data formats. It has, however, found success in many places due to its inherent simplicity and easy applicability.

3. The ns-3 Simulator for a simulation-in-the-loop approach

Running tests and experimenting with sensor data management scenarios with anything less than hundreds of nodes generating realistic data constantly is obviously non-relevant already when compared to some of the existing M2M and IoT solutions in the field today[16,26,34], but especially so when taking into account the future potential and technological visions[12,15,32,35–38] for the internet of things and pervasive sensor environments. On the other hand, experimenting exclusively with simulators does not provide results convincing and verifiable enough for scientific purposes. Therefore, a tool enabling the mixing and intercommunication of real and simulated networks and devices is required. The ns-3[39] is an open source project launched in 2006 for a discrete-event network simulator which has been developed by an open community primarily for the purposes research and educational use. The most common reasons to employ the ns-3 in research work are to perform experiments that are unfeasible to implement with real hardware and to study system behaviour in a controlled, reproducible environment. The ns-3 tool provides a set of models for how various packet data networks work and perform in addition to providing a simulation engine for users of the simulator to run their experiments. Also, compared to the legacy ns-2 simulator already familiar to many a researcher in the field of computer networks, ns-3 is an entirely new simulator that is not in any way backwards compatible with ns-2[40]. The ns-3 simulator has built-in support for simulations consisting of both IP and non-IP

based networks and includes a comprehensive library of models for a number of technologies such as the internet stack, ad hoc routing protocols, wireless radios, energy consumption, and mobility just to name a few. The key component and functionality provided by the ns-3 simulator for the purposes of the testbed work described in this paper is the ns-3 EMU net device, which essentially gives us the ability to inject traffic from the simulator[4] and the virtual nodes therein to a real world network with real devices and vice versa. One such scenario is shown in figure 1, where virtual ns-3 nodes communicate with each other by driving real testbed hardware. The ns-3 simulation platform is deployed as an integral part of the sensor data processing testbed proposed in this paper.

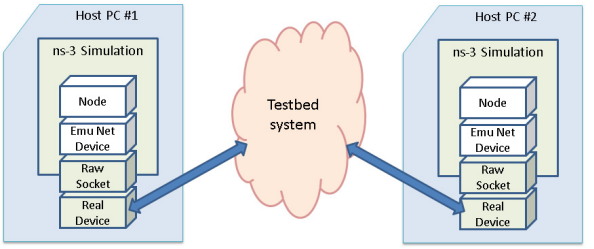

Figure 1. Simulated ns-3 applications attached to a testbed with real hardware via the emu netdevice.

For the testbed discussed in this paper, an ns-3 application is written on top of the EMU net device, which generates traffic towards the real network using the protocol(s) for sensor data as discussed in the previous section.

4. Design and structure of the sensor data testbed

The main aim is to set up a testbed for generating realistic sensor data, real or emulated, from a combination of physical hardware sensor nodes and virtual simulated nodes in a simulation-in-the-loop environment connected to the real testbed network. Several embedded platforms, such as the Raspberry Pi and Arduino, were chosen as additional parts of the testbed system based on their proven flexibility and capabilities of running real world IoT applications and sensor protocols[41]. The first version of the testbed contains the following components:

- Raspberry Pi Model B: Arm-based minicomputers with 512MB of Ram.
- STMicroelectronics NUCLEO-F401RE: a development board based on the STM32 microcontroller.
- Arduino UNO R3: Small microcontroller boards based on the ATmega328.
- Android mobile devices: A set of smartphones and tablets based on the Android platform.
- A Linux -desktop PC: An Intel i3-based multicore computer running the Ns3-simulation platform
- A distributed server: A set of Linux-based PC's to act as distributed data sinks for the generated sensor data.
- Networking hardware: Switches, adapters and routers required for interconnecting the components

A diagram for the design of the sensor data testbed overviewing the main components is shown in figure 2.

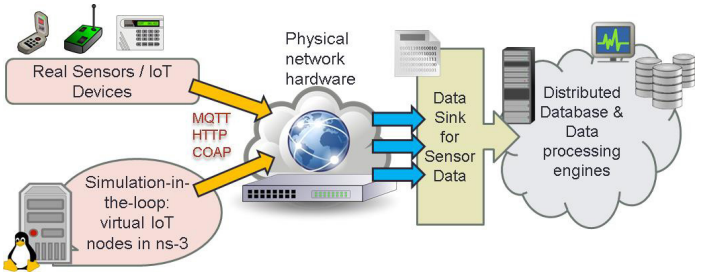

Figure 2. An overview and the main components of the sensor data testbed.

In order for the testbed to emulate the data and traffic generated by realistic IoT sensor environments, a hybrid approach is required where simulated virtual IoT nodes are injecting data traffic into a real network in combination with a significantly smaller amount of real hardware sensor nodes, which are also transmitting data towards a back-end. The back-end is essentially all the tools and technologies for handling this machine-generated data, making up the data processing pipeline for whatever IoT or sensor application case under study. Further discussion on the details concerning the back-end is left as the subject matter for another paper, but typically such a pipeline would include components for data ingestion, stream processing, data storage, processing engines and visualization.

5. Conclusions

In this paper, we have presented the initial design, setup and components for an interconnected simulation-in-the-loop data testbed to trial and experiment with pervasive sensor applications and management of large amounts of realistic machine-generated sensor data. The Internet-of-Things with its huge numbers of intercommunicating everyday objects and modern pervasive sensor environments is generating huge amounts of machine data, for which cloud-enabled back-end solutions capable of distributed big data processing and scalable data storage will be required. Such data management pipelines or sensor back-ends must be trialed and validated before deployment, preferably using realistic data and packet traffic patterns. But therein lies the very challenge; how do we generate realistic sensor data streaming in via state-of-the-art sensor applications and IoT communication protocols without relying exclusively on unconvincing methods of simulators and software tools or the unfeasible alternative of purchasing and setting up hundreds of potentially expensive hardware nodes? The simulation-in-the-loop based approach for the data testbed proposed in this paper was designed to meet this challenge by combining simulated virtual nodes that intercommunicate with real IoT sensors, devices and servers. This enables the implementation of a testbed setup where a large number of simulated nodes, instead of a potentially very expensive set of hundreds of real prototype devices, are interacting with a smaller number of physical devices through real world physical network hardware using the most relevant IoT message protocols and data formats.

While the initial design for the data testbed is complete and has been presented and the first phase of setting up the test environment has been successfully carried out with promising results, there are still many steps and matters to be improved or enhanced which are left for future work phases. Furthermore, the deployments of the testbed presented in this paper against specific data processing backend solutions, and any related test results thereof, are the subject matter for another prospective paper. Some issues that will be considered for future phases of the work are as follows: the addition of more diverse set of hardware sensor nodes, more powerful network hardware (e.g. switches, routers), running the simulation part in parallel distributed mode to generate even more voluminous amounts of data and the added support of more IoT communication protocols and sensor data formats.

Acknowledgements

Much of the research presented in this paper has been conducted under DIGILE's Need for Speed (N4S)[42] - program funded by Tekes, to which organizations the authors of this work wish to express gratitude.

References

1. What is the Internet of Things (IoT) n.d. http://www.sas.com/en_us/insights/big-data/internet-of-things.html (accessed April 8, 2015).
2. Redmond E, Wilson JR, Carter J. Seven Databases in Seven Weeks: A Guide to Modern Databases and the NoSQL Movement. Pragmatic Bookshelf; 2012.
3. Big Data: Rise of the Machines. Bits Blog n.d. http://bits.blogs.nytimes.com/2012/12/31/big-data-rise-of-the-machines/ (accessed April 8, 2015).
4. Ivanic N, Rivera B, Adamson B. Mobile Ad Hoc Network emulation environment. Military Communications Conference, 2009 MILCOM 2009 IEEE 2009:1–6. doi:10.1109/MILCOM.2009.5379781.
5. Dumbill E. What is Apache Hadoop? - O'Reilly Radar n.d. http://radar.oreilly.com/2012/02/what-is-apache-hadoop.html (accessed March 20, 2015).
6. Fan W, Bifet A. Mining Big Data: Current Status, and Forecast to the Future. SIGKDD Explor Newsl 2013;14:1–5. doi:10.1145/2481244.2481246.
7. Russell MA. Mining the Social Web: Data Mining Facebook, Twitter, LinkedIn, Google+, GitHub, and More. O'Reilly Media, Inc.; 2013.

8. Big Data—Still A Lot of Untapped Opportunity Amidst All the Noise — Entrepreneurship Blog n.d. http://beacon.wharton.upenn.edu/entrepreneurship/2013/02/big-data-still-a-lot-of-untapped-opportunity-amidst-all-the-noise/ (accessed April 8, 2015).
9. The Promise and Pitfalls of Machine-Generated Data. Inside Analysis n.d. http://insideanalysis.com/2012/06/promise-m-g-data/ (accessed April 8, 2015).
10. Examples and definition of machine-generated data n.d. http://www.dbms2.com/2010/12/30/examples-and-definition-of-machine-generated-data/ (accessed March 20, 2015).
11. Gartner Says the Internet of Things Installed Base Will Grow to 26 Billion Units By 2020 n.d. http://www.gartner.com/newsroom/id/2636073 (accessed April 21, 2015).
12. Latvakoski J, Alaya MB, Ganem H, Jubeh B, Iivari A, Leguay J, et al. Towards Horizontal Architecture for Autonomic M2M Service Networks. Future Internet 2014;6:261–301.
13. Rowstron A, Druschel P. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. Middleware 2001, Springer; 2001, p. 329–50.
14. Guizani B, Ayeb B, Koukam A. A new cluster-based link state routing for mobile ad hoc networks. Communications and Information Technology (ICCIT), 2012 International Conference on 2012:196–201. doi:10.1109/ICCITechnol.2012.6285790.
15. Aissa M, Belghith A, Drira K. New strategies and extensions in weighted clustering algorithms for mobile ad hoc networks. Procedia Computer Science 2013;19:297–304.
16. Yick J, Mukherjee B, Ghosal D. Wireless sensor network survey. Computer Networks 2008;52:2292–330.
17. Park C, Rappaport TS. Short-range wireless communications for Next-Generation Networks: UWB, 60 GHz millimeter-wave WPAN, and ZigBee. Wireless Communications, IEEE 2007;14:70–8.
18. Internet of Things Protocols. Postscapes n.d. http://postscapes.com/internet-of-things-protocols (accessed February 10, 2015).
19. Internet of Things requirements and protocols. Embedded Computing Design n.d. http://embedded-computing.com/articles/internet-things-requirements-protocols/ (accessed February 10, 2015).
20. Mineraud J, Mazhelis O, Su X, Tarkoma S. Contemporary Internet of Things platforms. arXiv Preprint arXiv:150107438 2015.
21. MQTT and CoAP, IoT Protocols n.d. http://eclipse.org/community/eclipse_newsletter/2014/february/article2.php (accessed February 10, 2015).
22. MQ Telemetry Transport (MQTT) V3.1 Protocol Specification 2010. http://www.ibm.com/developerworks/library/ws-mqtt/ (accessed March 20, 2015).
23. Shelby Z, Hartke K, Bormann C. The Constrained Application Protocol (CoAP) n.d. https://tools.ietf.org/html/rfc7252 (accessed March 20, 2015).
24. Leach PJ, Berners-Lee T, Mogul JC, Masinter L, Fielding RT, Gettys J. Hypertext Transfer Protocol -- HTTP/1.1 n.d. https://tools.ietf.org/html/rfc2616 (accessed March 20, 2015).
25. Laine M. RESTful Web Services for the Internet of Things. Online] Saatavilla: Http://media Tkk Fi/webservices/personnel/markku_laine/restful_web_services_for _the_internet_of_things Pdf 2012.
26. Latvakoski J, Iivari A, Vitic P, Jubeh B, Alaya MB, Monteil T, et al. A Survey on M2M Service Networks. Computers 2014;3:130–73.
27. Alaya MB, Banouar Y, Monteil T, Chassot C, Drira K. OM2M: Extensible ETSI-compliant M2M Service Platform with Self-configuration Capability. Procedia Computer Science 2014;32:1079–86.
28. Mineraud J, Mazhelis O, Su X, Tarkoma S. A gap analysis of Internet-of-Things platforms. arXiv Preprint arXiv:150201181 2015.
29. Kovatsch M, Mayer S, Ostermaier B. Moving application logic from the firmware to the cloud: Towards the thin server architecture for the internet of things, IEEE; 2012, p. 751–6.
30. <douglas@crockford.com> DC. The application/json Media Type for JavaScript Object Notation (JSON) n.d. https://tools.ietf.org/html/rfc4627 (accessed March 20, 2015).
31. Bray T, Paoli J, Sperberg-McQueen CM, Maler E, Yergeau F. Extensible markup language (XML). World Wide Web Consortium Recommendation REC-Xml-19980210 Http://www w3 org/TR/1998/REC-Xml-19980210 1998:16.
32. Iivari A, Väisänen T., Alaya M., Riipinen T., Monteil T. Harnessing XMPP for machine-to-machine communications & pervasive applications. Journal of Communications Software & Systems 10 (3)
33. <ietf@shaftek.org> YS. Common Format and MIME Type for Comma-Separated Values (CSV) Files n.d. https://tools.ietf.org/html/rfc4180 (accessed April 21, 2015).
34. Atzori L, Iera A, Morabito G. The internet of things: A survey. Computer Networks 2010;54:2787–805.
35. Dhraief A, Belghith A, Drira K, Bouali T, Ghorbali MA. Autonomic management of the HIP-based M2M overlay network. Procedia Computer Science 2013;19:98–105.
36. Gubbi J, Buyya R, Marusic S, Palaniswami M. Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions. Future Gener Comput Syst 2013;29:1645–60. doi:10.1016/j.future.2013.01.010.
37. Swan M. Sensor Mania! The Internet of Things, Wearable Computing, Objective Metrics, and the Quantified Self 2.0. Journal of Sensor and Actuator Networks 2012;1:217–53. doi:10.3390/jsan1030217.
38. Latvakoski J, Hautakoski T, Iivari A. Situated Service Oriented Messaging for Opportunistic Networks. Bioinspired Models of Network, Information, and Computing Systems, Springer; 2010, p. 50–64.
39. Riley GF, Henderson TR. The ns-3 network simulator. Modeling and Tools for Network Simulation, Springer; 2010, p. 15–34.
40. What is ns-3 « ns-3 n.d. https://www.nsnam.org/overview/what-is-ns-3/ (accessed April 8, 2015).
41. Hodges S, Taylor S, Villar N, Scott J, Bial D, Fischer PT. Prototyping connected devices for the internet of things. Computer 2013;46:26–34.
42. Digile N4S n.d. http://www.n4s.fi/en/ (accessed April 21, 2015).