

INTELLIGENT IPv6 BASED IOT NETWORK MONITORING AND ALERTING SYSTEM ON COOJA FRAMEWORK

J. Subramaniam¹, Y.L. HO¹ and S. Manickam^{2,*}

¹Faculty of Information Science and Technology, Multimedia University, 75450 Bukit Beruang, Melaka, Malaysia

²National Advanced IPv6 Centre (NAv6), School of Computer Sciences, Universiti Sains Malaysia, Georgetown, 11800 Penang, Malaysia

Published online: 10 November 2017

ABSTRACT

Internet of Things (IoT) has pretty much taken the digital world by storm. The conventional way of information being produced and outputted is through machine to machine communication. But, with sensors acting as middleware between these communications, it brings us with even higher state of connectivity and interactivity. On the basis of these, this paper discusses and addresses the issues of packet monitoring of IoT device simulations in Cooja framework. Addressing this issue is tantamount to addressing the security and packet loss issue in Cooja framework; a simulator provided by Contiki operating system to test, develop and run IoT enabled systems.

Keywords: IoT; Cooja framework; Contiki OS; packet monitoring.

Author Correspondence, e-mail: selva@nav6.usm.my, selva@usm.my

doi: <http://dx.doi.org/10.4314/jfas.v9i6s.51>

1. INTRODUCTION

Internet of Things refers to a mesh of interconnecting devices, sensors and network works



together and operate automatically to cater our needs [1]. Technology is an important factor [2-3] in one's daily routine and it is an ever expanding in its nature. New phenomenon is continuously discovered and this has been a big help in all aspects where the technology plays a major role. This includes the emergence of Internet of Things (IoT). IoT orbits around machine to sensors to machine communication which built on cloud computing. The conventional machine to machine communication relates solely around device talking to another device. But, it is limited to getting instruction and executing them as per the rules. Thus, when discuss about making machine "smart", it doesnot solely referring to machine to machine but about sensors. A sensor is a device that detect changes in the ambient conditions or in the state of another device or system and conveys or records this information in a certain manner [4]. Every prototypes, research products and software needs simulation before they were put use in real world. Simulations can be done with many software such as Netsim, ns-3, Omnet, etc.[5].

When it comes to formulating low level simulations for sensor nodes, the choice of tools is limited. Emulators like Cooja makes the task easier by allow large and small network of motes to be simulated. Cooja is specifically designed for wireless sensor network which supports IoT enabled devices which runs on Contiki operating system[6]. The contiki operating system is an embedded operating system that centered on sensor networks [7].Cooja also allow hex dump output of packets that are transmitted between nodes. During the simulation of sensor nodes, packets will be sent and received by the subsequent nodes which goes through border router; an intermediary to connect one network to another. The basic set up for a simulation consist of border router, UDP server; to set up non border router nodes in the simulation network and TUNSLIP6; tool use for bridging wireless IPv6 network into PC via USB connection or bridge IP traffic between host and border router. Tunslip act as bridging between the actual motes such as (Tmote, Zolertia) to establish connection to simulation nodes.

Due to the humongous exchange of information among the nodes, network congestion occurs and this series of chain effects leads to packet loss and congestion which is inevitable during the transmission of packets among each nodes[8].Packet loss in IoT devices can bring severe disruption during the real time usage. Packet loss can lead to jittering in VOiP audio and video communication, absence of incoming signal from another device and distortion of received

images. Combatting this problem in simulation phase will make way for better understanding and innovation of the devices before being put use in real world.

This paper presents the exploration of possible solutions, which is intended to monitor and analyze packet flow during simulation of IoT enabled devices. Comparison with existing system also been done to point out the advantage of this system compared to others.

2. EXISTING SYSTEM

Table 1 shows comparison between existing systems by stating out the essential features in packet monitoring system. Firstly, operating system plays a major role in cross compatibility software's. Since this research is aimed at Cooja framework, the OS is limited to Linux only. Other tools such as Ngrep and Wireshark supports more than one OS platform. Apart from that, all the tools compared and listed above is open source which is a good thing for developers to try innovate and works around with the software to come up with better and safer tools in the future. This tool is ipv6 and ipv4 supported just like Scapy and Snort. The main advantage of this project is that it comes with dashboard visualization whilst other tools were merely use for monitoring and analyzing of packets only. Dashboard feature will make the tool more interactive when comes to user appliances. Thus, they can easily interact with data logs provided and charts rendered in the dashboard.

Table 1. Comparison between existing systems

Features Systems	OS Supported	Open Source	Dashboard	IPV6/IPV4 Supported
Tcpdump	Unix based	Yes	No	No
Wireshark	Window and Unix	Yes	No	Yes
Snort	Unix	Yes	No	Yes
Scapy	Multi-platform	Yes	No	Yes
Ngrep	Multi-platform	Yes	No	No
This Project	Unix based	Yes	Yes	Yes

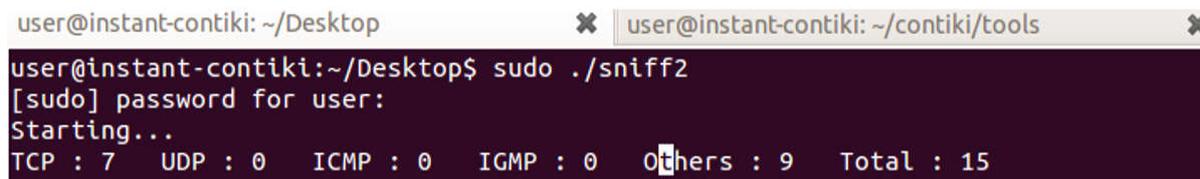
Similar system will be only used as a guidance to develop the proposed system. The features of all the system mentioned above have a lot of redundancy. The researcher will study the

similar system and choose the features that can be of a help for the proposed system and will make use of the feature as a proper guidance. Most of the functions in the system mentioned above are coexist among other system, thus plans to enhance the tool by creating a dashboard to aid the users during interacting with the output of the logs. Moreover, the researcher also focuses on establishing a simple, efficient, low resource monitoring tool for the Cooja framework.

3. PROJECT PROTOTYPE

The packet capturing tool can start capturing once the desired number of motes and bridged are linked in the Cooja simulator. Sniff2 will be initiated from terminal window. For the initial phase, the researcher only allowed the capturing of TCP and UDP packets then followed by ICMP and others. When the packet capturing started, data packets from the motes will be automatically feed into database.

3.1. Sniff2 Terminal Window and Packet Capturing



```
user@instant-contiki: ~/Desktop x user@instant-contiki: ~/contiki/tools x
user@instant-contiki:~/Desktop$ sudo ./sniff2
[sudo] password for user:
Starting...
TCP : 7  UDP : 0  ICMP : 0  IGMP : 0  Others : 9  Total : 15
```

Fig.1. Terminalwindow of Sniff2

Sniff2 will be executed from command line to capture the packets during the simulation in Cooja emulator. It has the command to capture various protocol packets such as TCP, UPD, ICMP and others such as ARP.

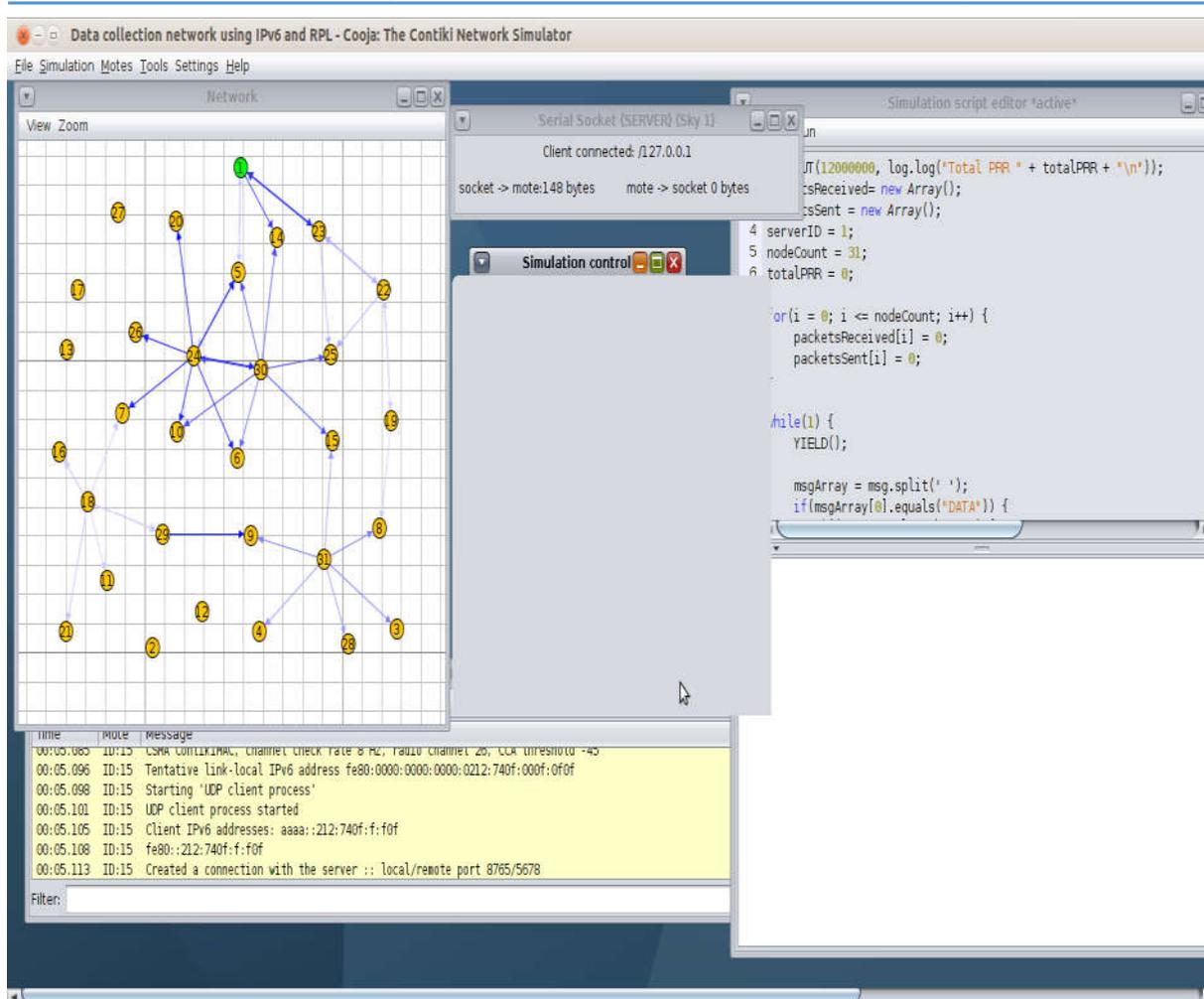


Fig.2. Cooja simulation with desired number of motes

Fig. 2 shows the execution of the sniffer tool, Cooja emulator is loaded with sky motes and ready to be activated. In this simulation, the user tested with Border Router and UDP server motes. Physical motes with desired set of number and configuration can be added on this motes. The green node is Border router which act as server side for the nodes.

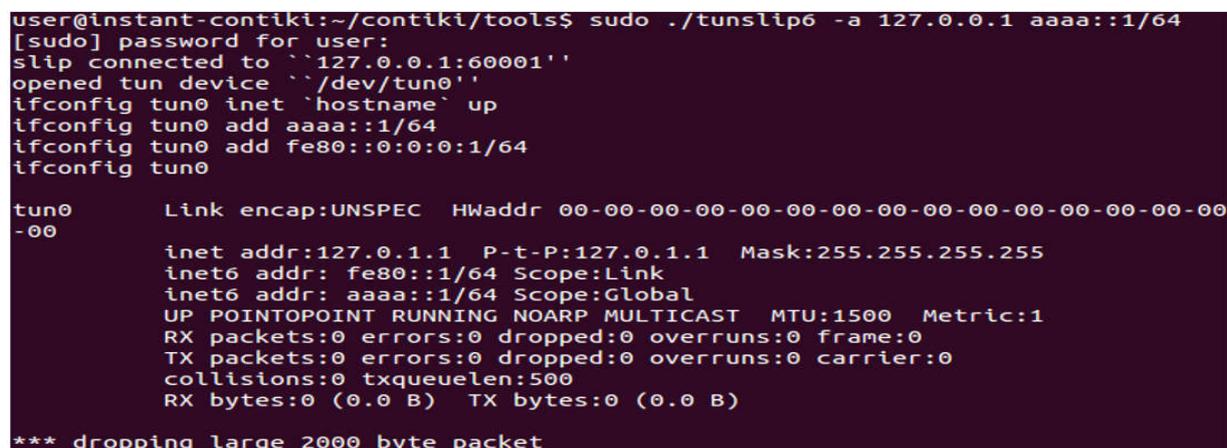


Fig.3. Successful connection of Tunslip6

Even though the network is setup, a bridge must be created with RPL network. Border Router will be used as serial socket server. Serial port will be created on node 1 that is accessible on the local machine via UDP port 60001. To enable the bridging, tunslip6 which sets up an interface on the Linux IP stack and connects this interface via a socket to the border router node in Cooja. The Linux interface will be configured, so that all IP traffic destined to addresses starting with aaaa: will go to the interface and into Cooja [9].

Fig. 4 shows Packets which have been captured will be feed into MySQL database before outputted in webserver. Each protocols packet information will be stored and showed in webserver. Figure below shows information of TCP and UDP packets.

2016-05-18	38203	68.232.44.121	68.232.44.121	80	53262	1502890457	2147483647	5	0	1	0	0
2016-05-18	38204	192.168.190.153	192.168.190.153	53266	80	1605317536	394421120	5	0	1	1	0
2016-05-18	38205	68.232.44.121	68.232.44.121	80	53266	394421120	1605317978	5	0	1	0	0
2016-05-18	38206	192.168.190.153	192.168.190.153	53270	80	1352621321	498145665	5	0	1	1	0
2016-05-18	38207	68.232.44.121	68.232.44.121	80	53270	498145665	1352621763	5	0	1	0	0
2016-05-18	38208	192.168.190.153	192.168.190.153	53275	80	421267222	779031665	5	0	1	1	0
2016-05-18	38209	68.232.44.121	68.232.44.121	80	53275	779031665	421267664	5	0	1	0	0
2016-05-18	38210	192.168.190.153	192.168.190.153	53274	80	2107522838	906956918	5	0	1	1	0
2016-05-18	38211	68.232.44.121	68.232.44.121	80	53274	906956918	2107523280	5	0	1	0	0
2016-05-18	38212	192.168.190.153	192.168.190.153	53268	80	2147483647	1980321545	5	0	1	1	0
2016-05-18	38213	68.232.44.121	68.232.44.121	80	53268	1980321545	2147483647	5	0	1	0	0
2016-05-18	38214	192.168.190.153	192.168.190.153	53276	80	2147483647	0	10	0	0	0	0
2016-05-18	38215	68.232.44.121	68.232.44.121	80	53276	206645745	2147483647	6	0	1	0	0
2016-05-18	38216	192.168.190.153	192.168.190.153	53276	80	2147483647	206645746	5	0	1	0	0
2016-05-18	38217	192.168.190.153	192.168.190.153	53269	80	2147483647	1299588193	5	0	1	1	0
2016-05-18	38218	68.232.44.121	68.232.44.121	80	53269	1299588193	2147483647	5	0	1	0	0
2016-05-18	38219	192.168.190.153	192.168.190.153	53263	80	105828850	2139053817	5	0	1	1	0
2016-05-18	38220	68.232.44.121	68.232.44.121	80	53263	2139053817	105829292	5	0	1	0	0
2016-05-18	38221	68.232.44.121	68.232.44.121	80	53262	1502890457	2147483647	5	0	1	1	0
2016-05-18	38222	192.168.190.153	192.168.190.153	53262	80	2147483647	1502890966	5	0	1	0	0
2016-05-18	38223	192.168.190.153	192.168.190.153	53271	80	400278897	150996277	5	0	1	1	0
2016-05-18	38224	68.232.44.121	68.232.44.121	80	53271	150996277	400279339	5	0	1	0	0
2016-05-18	38225	192.168.190.153	192.168.190.153	53273	80	2147483647	1550476025	5	0	1	1	0
2016-05-18	38226	68.232.44.121	68.232.44.121	80	53273	1550476025	2147483647	5	0	1	0	0
2016-05-18	38227	68.232.44.121	68.232.44.121	80	53266	394421120	1605317978	5	0	1	1	0
2016-05-18	38228	192.168.190.153	192.168.190.153	53266	80	1605317978	394421628	5	0	1	0	0

Fig.4. Capturing of TCP packet

3.2. Dashboard and Visualization

Visualizing the data will be easier for the user to pinpoint certain queries such as jittering, latency or time delay in each packets. The main advantage of this packet capturing tool is the dashboard features. The data captured will be automatically feed into and visualized in web server using Google chart and d3.js. Both serve as interactive for data visualization as charts.



Fig.5. Dashboard feature of Sniff2 tool

Fig. 5 shows the many features of sniff2 dashboard tool. User can view packet statistics, packet loss of each relevant packets captured on the go. Besides that, user can also view specific type of protocols and the capture data from the dashboard for easier viewing and monitoring.

Type of Protocols

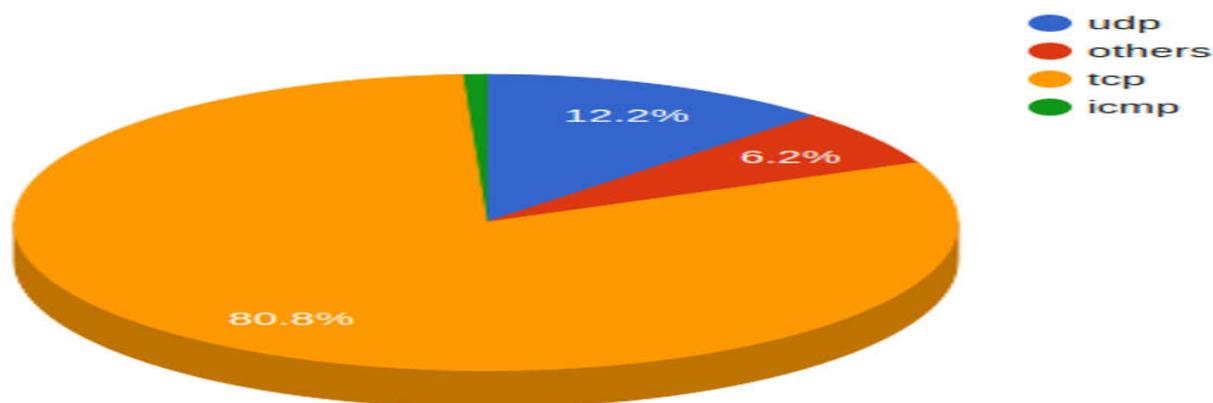


Fig.6. Type of protocols

Fig. 6 shows the pie chart of the type of protocols and their percentage count in easier form of viewing. User can also modify and logs into data anytime, anywhere for their ease of use.

3.3. Workflow of Sniff2

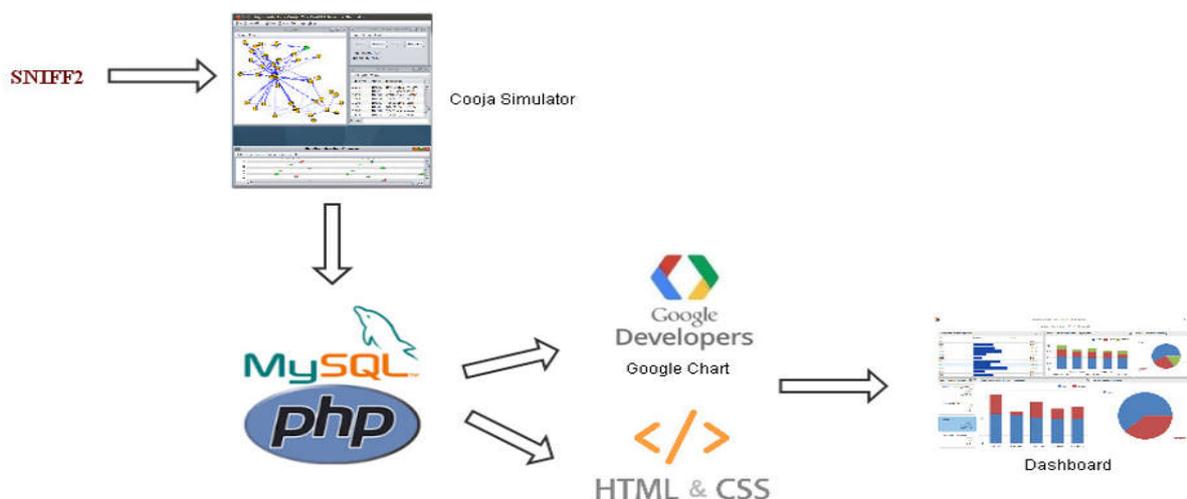


Fig.7. Workflow of Sniff2 packet monitoring tool

The system starts with Sniff2 packet capture tool. The system is started when Cooja simulation begins and the sniffer will capture the appropriate packets and store them in logfile format. MySQL database fetches the logs collected from the packet and store it in readable format. Important checklist and headers such as source IP and destination IP address, source and destination port, list of flags and date will be recorded in the database. PHP on the other hand parse the data to output into web server to produce a dynamic web page content or images on websites. HTML and CSS are used to design the layout of the website. JavaScript is added for interactive functionality and should be referenced in separate files. Google chart is used in this research project because it is user friendly, scalable and easy referencing. The list of Google API regarding the type of charts are given in their website. Thus, modifying the codes according to our data input and type of charts is the main thing in here.

The main window of this sniffing tool will be the command line. When the packet capturing started. Data will be automatically feed into database. The output of the terminal window will show type of protocols and packet count. TCP packet count will be the largest followed by UDP. The data will be outputted in web server by fetching the information from database. All the relevant headers and packet information are stored in there and will be fetched for easy viewing. Database are also used to create charts based on user preferences. The main reason to create a dashboard is for easy user interactivity. Information such as packet count of each type of protocols, packet capture information, charts for various selected field will be yielded in the dashboard. Future scope may expand into more interactivity and information features.

4. CONCLUSION

Internet of Things (IoT) refers to uniquely classifiable objects, things, and their visual representation in an internet like structure [10]. Since the IoT is growing exponentially, these sensor equipped devices will provide new benchmark and extraordinary ability to sense and monitor our homes, environment, industries and even our bodies. Smart sensors including radio frequency identification (RFID) tags, serve three broad purposes. They identify items, locate them and determine their environmental conditions, all of which have major implications for the supply chain and manufacturing [11]. Sensor data can be distributed through our device and sent to the data center to be analyzed[12].

Packet capturing tools are common in world of internet nowadays. The ability to examine packets is vital for the development of protocol during simulation of motes in Cooja. Intelligent IPv6 based Internet of Things (IoT) based Network Monitoring and Alerting system is a real time monitoring system intended for Cooja framework. Cooja is a Contiki OS network emulator which allow large and small networks of motes to be simulated.

This paper provides one of the many solutions for capturing packets in Cooja simulator in Contiki operating system using IPv6/IPv4 as connectivity. Starting with base which is from packet analyzing and alerting will be a stepping stone on overcoming this ever growing issues. There will be no denial that instigating monitoring system in IoT related physical motes simulations will be implemented in IoT enabled device in future.

5. ACKNOWLEDGEMENTS

Most of the work was done in collaboration with National Advanced IPv6 Centre (NAv6), Universiti Sains Malaysia.

6. REFERENCES

- [1] Johnson S. Using mesh networking to interconnect IoT devices. 2016, <http://internetofthingsagenda.techtarget.com/feature/Using-mesh-networking-to-interconnect-IoT-devices>
- [2] Morrison D, Natvig L. Internet of Things (IoT): Technology and applications for a good society. 2015,

<https://www.hipeac.net/events/activities/7236/internet-of-things-iot-technology-and-applications-for-a-good-society/>

[3] Weiss J, Wu R. Wireless sensor networking for the industrial IoT. 2015, <http://electronicdesign.com/iot/wireless-sensor-networking-industrial-iot>

[4] Füllner H. Simulation software: OMnet++ GTNetSGlomoSim / Qualnet. 2004, <http://pi4.informatik.uni-mannheim.de/pi4.data/content/courses/2004-ss/netsim/area51/2004-07-14-Tools2.pdf>

[5] Dunkels A, Osterlind F. Contiki programming course: Hands-on session notes. 2008, <https://www.sics.se/~thiemo/seniot09cccc-notes.pdf>

[6] Dunkels A, Gronvall B, Voigt T. Contiki-A lightweight and flexible operating system for tiny networked sensors. In 29th Annual IEEE International Conference on Local Computer Networks, 2004, pp. 455-462

[7] LX Group. Contiki-The low power IoT operating system. 2014, <https://lx-group.com.au/contiki-low-power-iot-operating-system/>

[8] Gonizzi I P, Duquennoy D S. Hands on Contiki OS and Cooja simulator: Exercises (Part II). 2013, https://george.autonomic-networks.ele.tue.nl/files/exercise_partII.pdf

[9] Weber R H. Internet of Things-New security and privacy challenges. Computer Law and Security Review, 2010, 26(1):23-30

[10] O'Donnell J. How smart sensors are transforming the Internet of Things. 2015, <http://internetofthingsagenda.techtarget.com/opinion/How-smart-sensors-are-transforming-the-Internet-of-Things>

[11] Schurgot M R, Shinberg D A, Greenwald L G. Experiments with security and privacy in IoT networks. In IEEE 16th International Symposium on a World of Wireless, Mobile and Multimedia Networks, 2015, pp. 1-6

[12] Business Dictionary. Sensor. 2017, <http://www.businessdictionary.com/definition/sensor.html>

How to cite this article:

Subramaniam J, HO YL, Manickam S. Intelligent ipv6 based iot network monitoring and alerting system on cooja framework. J. Fundam. Appl. Sci., 2017, 9(6S), 661-670