Proceedings of the International Conference on Intelligent Computing and Communication for Smart World (I2C2SW-2018)
14-15 December 2018

Enhancement of Routing Protocol for Low Power Lossy Network for Internet of Things

S.B.Gopal
Department of ECE
Kongu Engineering College
Perundurai, India
gopal.ece@kongu.ac.in

C.Poongodi
Department of IT
Kongu Engineering College
Perundurai, India
poongs@kongu.ac.in

D.Nanthiya
Department of CT
Kongu Engineering College
Perundurai, India
nanthiya.ctug@kongu.ac.in

K.Harish
Department of ECE
*Kongu Engineering College*Perundurai,India
harishk.15ece@kongu.edu

E.Divya
Department of ECE
Kongu Engineering College
Perundurai,India
divyae.15ece@kongu.edu

N.Aarthy
Department of ECE
Kongu Engineering College
Perundurai,India
aarthyn.15ece@kongu.edu

***Abstract*—Internet of Things (IoT) is a keyword which coins various challenges to be addressed in the area of networking and security. 6LoWPAN act as bridge between wireless sensor networks (WSN) and Internet. IoT Operating system is one of the major area, where we can able to change the behavior of the WSN nodes which is to be deployed. The main challenge in WSN is power consumption, as most of the devices used are battery powered. Medium Access Layer (MAC) and Network Layer protocols plays major role in effective utilization of power consumption. Routing Protocol for Low Power and Lossy Networks (RPL) is universally accepted routing protocol for IOT which is designed for static environment. This paper deals with conversion of static environment to mobile environment where the nodes be dynamic in its position for mobilitybased application and modifying the parameters in existing trickle timer algorithm for mobile environment. Cooja Simulator from Contiki Operating System is used to implement the simulation experiments. Comparison was done between Static trickle timer algorithm with modified trickle timer algorithm based on power consumption for different number of nodes in a common environment.**

Keywords—RPL, Trickle timer, Power consumption

I. INTRODUCTION

6LoWPAN is a background protocol that connects WSN with internet in IOT.It comprises more number of battery powered physical devices which is used to sense, collect and to transport the data from end user to remote location. Operating system (OS) plays a major role in any of the systems, as it controls entire operations of a particular device[1] . In IoT, OS plays key role in effective utilization of life time of battery powered devices. Different OS are available for IoT devices in market. This paper deals with Contiki OS, as it requires minimum random access memory (RAM) of 2 KB and minimum read only memory (ROM) of 30 KB. Contiki OS supports Cooja simulator, used for simulating the implementation before going for hardware testing. It divides entire processing of OS into four layers such as radio layer, MAC layer and Network layer. Radio layer is similar to physical layer in networks and MAC layer divided in to MAC and Duty cycling layers. Adaption, Routing, Transport and Application layers are combinely called as Network layer.

II. RPL

It is also called as Distance vector and source routing protocol [2]. Distance vector routing protocol will work by sharing its entire routing table with its immediate neighbors and also updates its table based on received routing table from others for calculating the link cost between any two nodes. Source routing protocol will define the entire path form source to destination before transmitting the packets. RPL works based on Directed acyclic graph (DAG) which in-turn divided into one or more Destination oriented DAG. All the nodes connected under a single parent will follow same RPL instance ID. The nodes will share DODAG Information Object (DIO) with its neighbours in order to update its DIO.Nodes having same Instance ID will share the functions in order to know its position. This Protocol holds good for Static environment. But the static environment is changed into dynamic and position of nodes at various time periods has generated from Bonn Motion tool.

III. TRICKLE TIMER ALGORITHM

In RPL, nodes will share its DIO with its neighbour's in-order to update and construct DODAG based on Trickle timer Algorithm [3]. Whenever the trickle timer gets expired, sharing of DIO will take place. First it will check, is there any change in environment. If any changes exists then it will update DIO between any two DODAG. Three parameters Imax, Imin and Redundancy Constant (K) plays key role in updation of DODAG. Time interval starts at Imin and ends at Imax. First the Value I will be set between Imin and Imax, counter (c) will be initialized to 0 and random value will be assigned to t. Next if the received DIO message is consistent then counter value will be incremented and DIO message get suppressed or else if received DIO message is inconsistent then DIO message will be transmitted. This algorithm will suits for Static environment. For Dynamic environment Modified trickle timer algorithm was proposed. As in dynamic environment the position of

nodes will vary frequently will leads to frequent transmission of DIO messages. Hence the redundancy constant is increased from 0 to 1 and Imax from 2.0 to 4.0 for reducing the transmission of control packets.

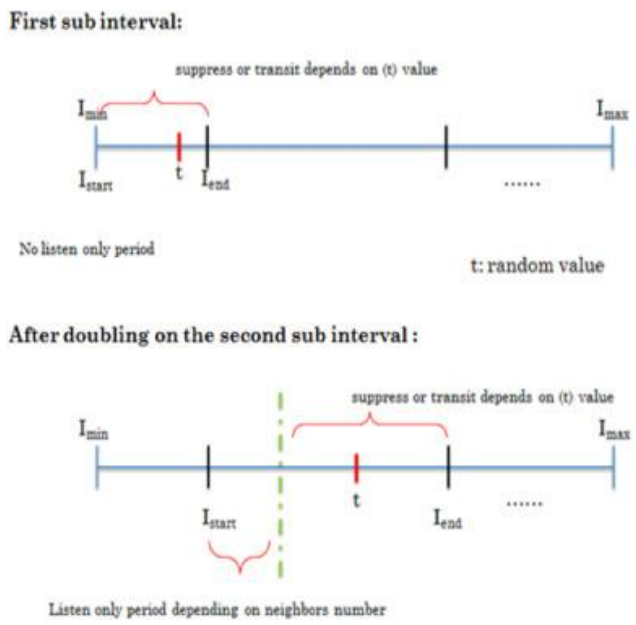

Fig. 1. Standard trickle algorithm

IV. Simulation And Its Performance

The simulation shows the performance of existing trickle algorithm and modified trickle in terms of power consumption. The Cooja simulator in Contiki operating system was used to perform the simulation. The Contiki operating system requires minimum memory allocation having atleast 2kB of random access memory and 30kB of read only memory.The environment is created with nodes, placed randomly in position. Table 1 list the parameters used for Simulation Environment.

TABLE I. Simulation Environment

Parameter	Values
Time duration	3600s
Node Density	20,40,60
Imin	1
Imax	4
Counter (C)	0
Redundancy Constant (K)	1
Network Topology	Random

20 Nodes with existing algorithm:

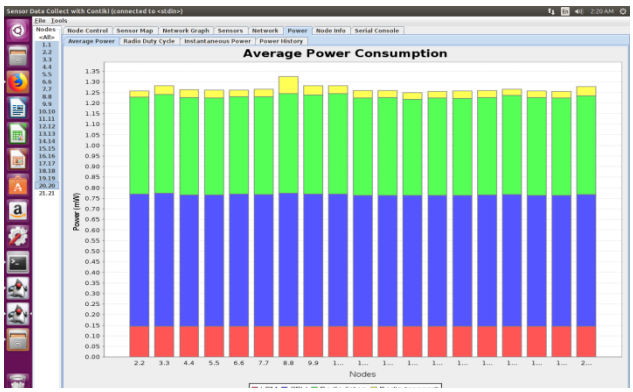

Fig. 2. Existing trickle algorithm for 20 nodes

The Fig. 2 shows the power consumption of the mobile nodes with values of the existing trickle timer for 20 nodes.

Environmental Setup
Number of Nodes: 20
Mobility: Dynamic
Sink node: 1
Slave nodes: 19
Firmware: Sky mote
Simulation time: 3600 seconds

40 Nodes with existing algorithm:

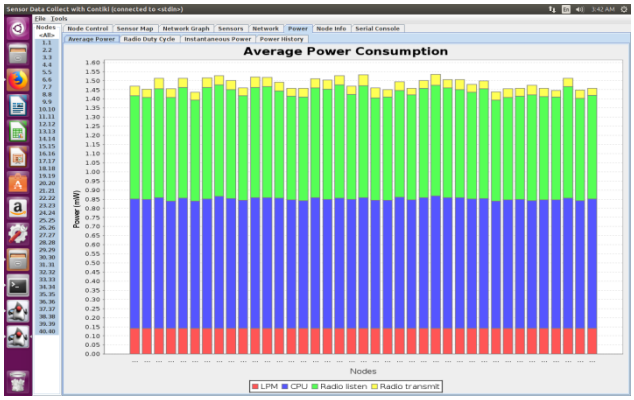

Fig. 3. Existing trickle algorithm for 40 nodes

The Fig. 3. shows the power consumption of the mobile nodes with values of the existing trickle timer for 40 nodes

Environmental Setup
Number of Nodes: 40
Mobility: Dynamic
Sink node: 1
Slave nodes: 39
Firmware: Sky mote
Simulation time: 3600 seconds

60 Nodes with existing algorithm:

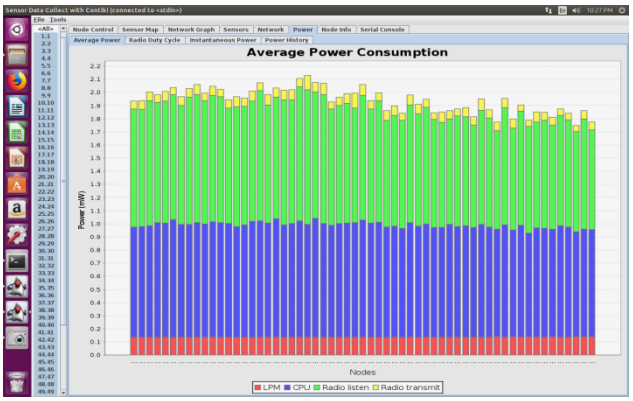

Fig. 4. Existing trickle algorithm for 60 nodes

The Fig. 4 shows the power consumption of the mobile nodes with values of the existing trickle timer for 60 nodes

Environmental Setup
Number of Nodes: 60
Mobility: Dynamic
Sink node: 1
Slave nodes: 59
Firmware: Sky mote
Simulation time: 3600 seconds

20 Nodes with modified algorithm:

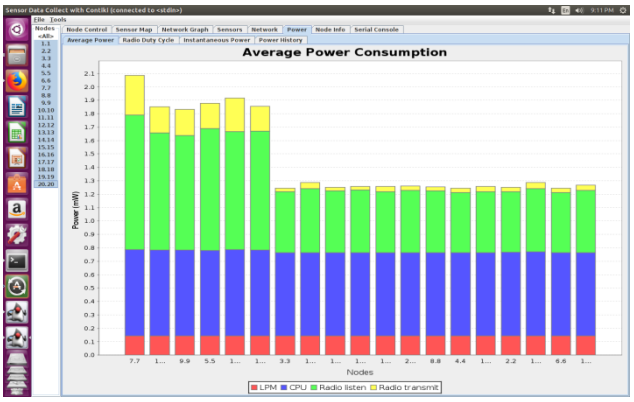

Fig. 5. Modified trickle algorithm for 20 nodes

The Fig.5. shows the power consumption of the modified trickle timer for 20 nodes. The modified values include k=1, Imin=2 and Imax=4.

Environmental Setup
Number of Nodes: 20
Mobility: Dynamic
Sink node: 1
Slave nodes: 19
Firmware: Sky mote
Simulation time: 3600 seconds

40 Nodes with modified algorithm:

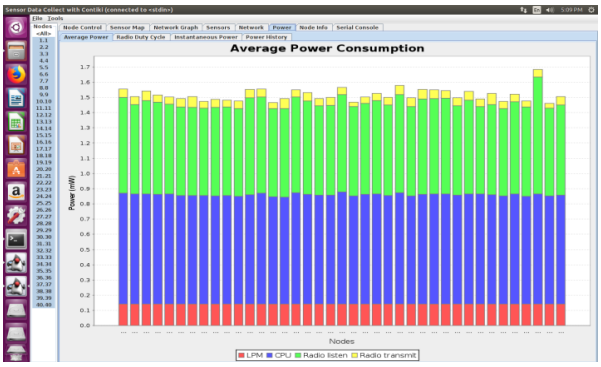

Fig. 6. Modified trickle algorithm for 40 nodes

The Fig. 6. shows the power consumption of the modified trickle timer for 40 nodes. The modified values include k=1, Imin=2 and Imax=4.

Environmental Setup
Number of Nodes: 40
Mobility: Dynamic
Sink node: 1
Slave nodes: 39
Firmware: Sky mote
Simulation time: 3600 seconds

60 Nodes with modified algorithm:

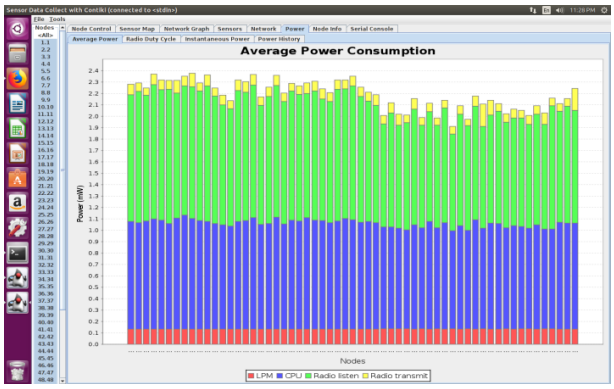

Fig. 7. Modified trickle algorithm for 60 nodes

Fig. 7. shows the power consumption of the modified trickle timer for 60 nodes. The modified values include k=1, Imin=2 and Imax=4.

Environmental Setup
Number of Nodes: 60
Mobility: Dynamic
Sink node: 1
Slave nodes: 59
Firmware: Sky mote
Simulation time: 3600 seconds

V. CONCLUSION

Thus the conversion of static environment to mobile environment was done using Cooja simulator. The performance was measured between the static and dynamic trickle timer by varying the values of the trickle timer for 20, 40 and 60 nodes. The observed performance reveals that the measured power consumption for the modified trickle timer

is comparatively high than the existing trickle time. As the power consumption is relatively high in modified trickle timer when compared to the existing trickle timer, further study can be made to reduce the power consumption by varying the values of k, Imin and Imax. In addition with that, the other parameters like Latency, convergence time, Power density ratio can be measured and analyzed.

REFERENCES

[1] Yousaf Bin Zikria a, Muhammad Khalil Afzal b, FarruhIshmanoSung Won Kim a,Heejung Yu a 'A survey on routing protocols supported by the Contiki Internet of things operating system',(2018)

[2] T. Winter, Ed, P. Thubert, Ed.A. Brandt, J. Hui 'RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks', RFC 6550, (2012) pp 1-157.

[3] Yassein M.B, S.Aljawarneh, E.Masa'deh, A new elastic trickle timer algorithm for Internet of Things, J. Netw. Comput. Appl. 89 (2017) pp 38 47.

[4] Contiki Development at ANRG, University of Southern California