2020 IEEE 17th International Conference on Mobile Ad Hoc and Sensor Systems (MASS) | 978-1-7281-9866-8/20/$31.00 ©2020 IEEE | DOI: 10.1109/MASS50613.2020.00072

Routing via Multiple Paths and Multiple Technologies in IoT Networks: Proof-of-Concept Demonstration

Vijeth J Kotagi, S P Vinayaka, and C Siva Ram Murthy, *Fellow, IEEE*
Indian Institute of Technology Madras, Chennai – 600036, India
vijethjk@gmail.com, sp.vin157@gmail.com, murthy@iitm.ac.in

***Abstract*—With the invent of many new wireless technologies, the concept of Internet of Things (IoT) is becoming a reality and is generating an enormous amount of data collectively. To handle routing of this massive traffic in a wireless network we require specialized, cost-effective routing devices and protocols. Transmitting IoT data over a wireless back-haul network using a single technology may lead to low throughput and increased delay due to high congestion. Critical IoT applications may not tolerate such a low throughput and high delay. To tackle this problem, in this paper, we propose *Routing via Multiple Paths and Multiple Technologies* (RMPMT) protocol to realize the concept of Parallel Opportunistic Routing (POR) where multiple paths and multiple technologies are exploited in an IoT network to route the data. We have implemented the proposed RMPMT protocol by mounting multiple wireless interfaces (technologies) on Raspberry Pi and measured the effectiveness of the proposed protocol by transmitting data in real-time. Furthermore, we also propose a split window automatic repeat request to provide guaranteed packet delivery in multi-path and multi-technology environment.**

***Index Terms*—IoT Network, IoT Gateway, Routing, Bluetooth, Zigbee, WiFi HaLow, Raspberry Pi.**

I. Introduction

THE Internet of Things (IoT) is a paradigm of controlling the real world remotely through the cyber world using the IoT devices equipped with sensors and actuators. Any daily usable things such as watch, refrigerator, television, washing machine, air conditioner, and vehicle can be an IoT device. Over the past few years, the usage of IoT devices has increased exponentially. It is estimated that by 2022 there will be 3.6 per capita connected devices to the Internet [1]. According to [2] by 2025 healthcare and manufacturing sector will have a larger share in IoT usage. The current trend in IoT is to save power by making power efficient devices. As a trade-off these devices will experience low data rate for transmission. Critical applications where continuous [3] and high speed connectivity [4] are required, existing transmission methods fail to fulfill the requirements.

An IoT network is usually deployed in unlicensed Industrial, Scientific and Medical (ISM) radio bands. As many technologies are deployed in an unlicensed spectrum, the gateways in the IoT network are to be provided with interfaces to support all these technologies [5]. In [6] the authors define two such gateways called Solution Specific Gateway (SSGW) and IoT Gateway (IGW). An SSGW provides support to a subset of technologies present in the network, where as an IGW is considered to support all the technologies present in the network. Furthermore, an IGW is considered to have connectivity to the Internet via wired back-haul network. SSGWs communicate with the sensing layer and forward the data to IGWs which in turn forward data to the Internet for further processing. Figure 1 shows the architecture of an IoT network. It consists of mainly three layers, viz., sensing layer, networking layer, and application layer. The sensing layer consists of sensors, actuators, and sinks which collect the data from the environment/IoT devices and passes the information to the networking layer. The networking layer consists of set of SSGWs and IGWs. The data so collected by the sensing layer is transmitted to the nearest SSGW or IGW. The SSGWs further transmit the data to the nearest SSGW until the data reaches an IGW which has connection to the Internet. The networking layer will further transmit the data to the application layer for processing the data. The key thing to note here is that, the SSGWs have to transmit the

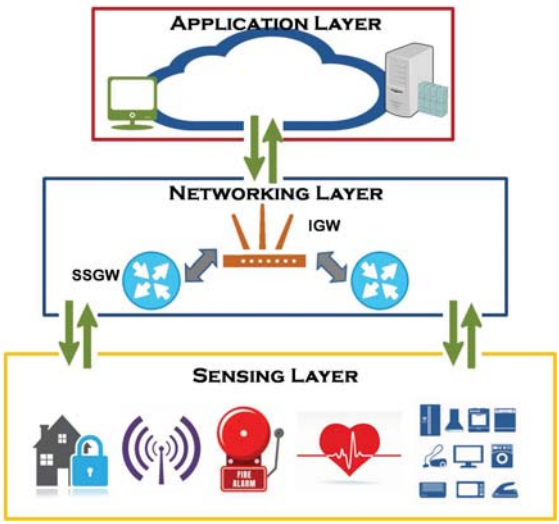


Fig. 1: Architecture of IoT network.

2155-6814/20/$31.00 ©2020 IEEE
DOI 10.1109/MASS50613.2020.00072

data to an IGW. This can involve multiple routes via different technologies to the nearest IGW.

The importance of SSGWs is as follows:

- In a real world scenario, in any considered subset of region of interest, the sinks/sensors of all the existing technologies may not be present. Therefore, it would suffice to have only limited technologies in a gateway.
- Limiting the number of technologies in a gateway will reduce the cost of the gateway, thereby reducing the cost of the system.
- Furthermore, not all places can have the Internet access. Therefore, it would be necessary to forward data to a node which has connection to the Internet (in this paper, we have considered it to be an IGW) via intermediate nodes (SSGWs).

The importance of IGWs is as follows:

- The IGW is considered to have all the available technologies, and therefore, it can receive data from all the SSGWs in its vicinity.
- The IGW is also considered to have the Internet connection, and therefore, acts as the destination node for all the data collected in an IoT network before it is sent to the cloud for processing.

There are many protocols for IoT networks in the literature. In [7] the authors use Constrained Application Protocol (CoAP) as a protocol for healthcare applications. It works on TCP/IP in the background and it does not serve the purpose of avoiding congestion without trading off the throughput. The authors in [8] defined Multi-Radio Access Technology (M-RAT) as the integration of multiple RATs in a single device to support a wide range of wireless devices. The authors state that in an M-RAT, a common data plane protocol running on a single control plane handles multiple radio technologies. The authors have proposed this for a 5G network, but it can be generalized to all RATs.

The authors of [9] propose an energy efficient message scheduling algorithm in an IoT network to improve overall efficiency of an IoT system. In [10], the authors discussed the possibilities of increased Quality of Experience (QoE) for users with a concept of MultiHomed routing, where either multiple IP addresses are used on a single interface, or multiple interfaces with dedicated IP address are used to achieve multi-path routing. It provides a few benefits such as traffic engineering, load balancing, reliable communication, and network resource utilization. All the protocols work on a single technology, and fail to exploit multiple paths created in an environment consisting of multiple heterogeneous wireless interfaces. The authors of [11] proposed a fitness function to be used on Ad-hoc On-demand Multipath Distance Vector (AOMDV) and Ad-hoc On-demand Multipath with Life Maximization (AOMR-LM). The authors use the AOMDV and AOMR-LM protocols along with fitness function to achieve multipath routing based on the lifespan of a mobile node in the path. AOMR-LM chooses a single path for transmission from all available paths, based on the life expectancy of a path and cost (in terms of the number of nodes) but it will not exploit other possible paths. The authors in [12] give a detailed survey on multipath routing protocols for QoS assurances in wireless sensor networks.

In our earlier work [6] we have proposed a Parallel Opportunistic Routing (POR) technique where data is transmitted in parallel across multiple technologies and multiple paths available in an IoT network to increase the throughput of the system. Furthermore, in [13] we have also proposed a way to balance the load in the IoT network which decreases the latency of the transmission of a packet in the network and increases the channel utilization. In this work, we develop and implement a protocol called *Routing via Multiple Paths and Multiple Technologies* (RMPMT) to realize POR. Furthermore, we also propose a split window automatic repeat request to provide guaranteed packet delivery in multi-path and multi-technology environment.

The proposed RMPMT differs from conventional multi-path routing protocols in the following ways:

1) In RMPMT we consider a multi-path scenario where a route exists between two nodes due to the presence of multiple technologies.
2) As one route may have different technologies, an intermediate packet may have to be converted to a suitable format for those technologies.

The major contributions of this paper are as follows:

1) We present a new protocol called RMPMT to realize POR in an IoT network.
2) We propose a split window automatic repeat request to ensure reliable packet delivery in a multi-path and multi-technology environment.
3) We provide a proof of concept for the proposed RMPMT protocol by mounting multiple wireless interfaces (technologies) on Raspberry Pi and transmitting the data in real-time.
4) With rigorous testing on a testbed, we show the effectiveness of the RMPMT protocol.

II. Routing via Multiple Paths and Multiple Technologies

A. Neighbor

As mentioned above, in this paper we use the architecture as given in Figure 1. The sensing layer collects the data and forwards the data to the nearest SSGW/IGW. An SSGW supports a subset of technologies which depends on the set of technologies of sensors from which it is collecting data. Therefore, different SSGWs may have different set of technologies (wireless interfaces) in them. Two SSGWs in the networking layer are said to be neighbors if they have a common technology between them. For example, consider the topology given in Figure 2. It shows the set of SSGWs present in the network layer. As it can be seen, not all SSGWs support all technologies, and Nodes A and B are neighbors as they have a common technology (Bluetooth LE) between them.

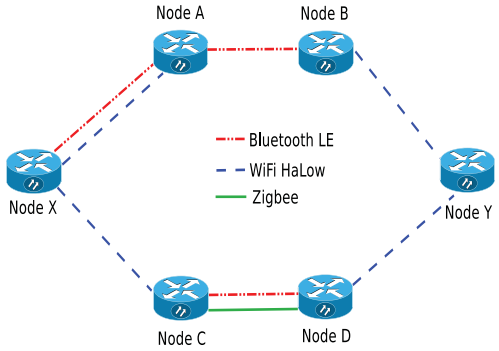


Fig. 2: Example topology.

The RMPMT protocol is built to increase the throughput in IoT networks using the concept of POR, which transmits the data in parallel by exploiting multiple paths and multiple Wireless Interfaces (WIs) available in the network. Unlike conventional multi-path routing, the POR has to deal with multiple WIs. For example, in Figure 2 if nodes X and Y are source and destination, respectively, then the data can be transmitted via X-A-B-Y or via X-C-D-Y. However, at the source X, the data can be transmitted simultaneously routed via two paths using two different technologies (in figure, say WiFi HaLow and Bluetooth LE). This requires that the data at the source X must be converted into packets of two different protocols and transmitted. Again in the intermediate nodes, protocol conversion may be required. For example, at node C, the data received in WiFi HaLow interface must be then converted to either Zigbee/Bluetooth LE to transmit to node D. Note that, multi-path between source and destination can mean that there are two different intermediate nodes, or it could also mean that there are two paths via two different technologies between same pair of nodes. For example, in the Figure 2 between nodes X and A there are two paths via two different technologies, where as between nodes X and Y there are two different paths altogether via different nodes. Furthermore, as the technologies are different and work in different channels, we consider that there is limited inter-technology interference.

B. Packet Structure

As POR requires protocol conversion in the intermediate hops, there is overhead of packet format modification because of the standardized packet structure for each of the wireless technologies and repeated fragmentation due to the varying payload size in different technologies. To overcome this, protocol enveloping approach is used in the work. A common packet format is used by appending a custom packet header to the payload data. This packet is given to WI as payload for transmission. Hence the need for packet format modification and repeated fragmentation is eliminated.

The packet structure of the RMPMT is shown in Figure 3. Each packet is composed of a mandatory header followed by the payload. The header occupies 10 bytes, whereas payload size is variable according to the wireless technologies

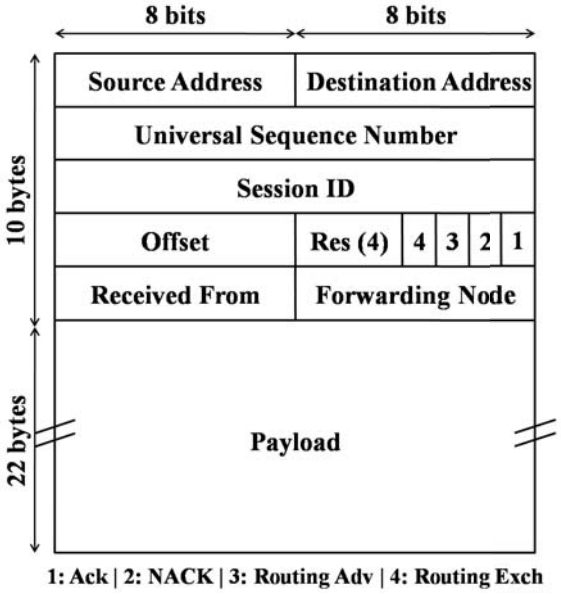
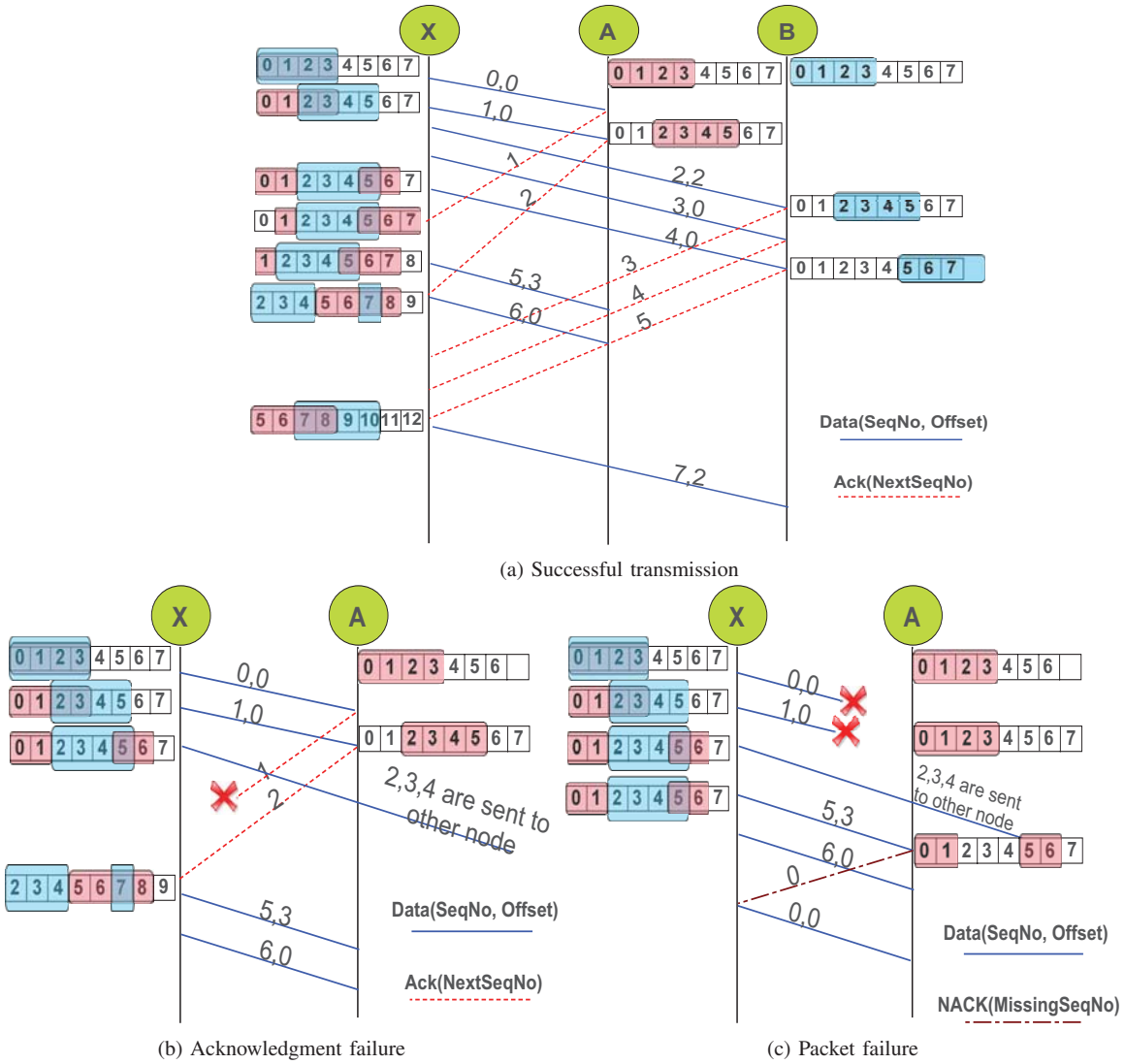


Fig. 3: Packet structure.

employed in the testbed. The total size of the packet should not exceed the maximum payload size of the standard protocols of WI. During the experimentation, in this paper, due to the limit of UART buffer in Zigbee, the size of the packet is set to 32 Bytes. Hence the payload size is restricted to 22 Bytes.

The packet header fields are as follows:

- **Source address**
  The source address is of 8 bits which defines the source address of the node in the IoT network.
- **Destination address**
  The destination address is of 8 bits which defines the destination address of the node.
- **Universal sequence number**
  Universal sequence number is of 16 bits and holds the sequence number of fragmented data. The destination node arranges the packets according to this universal sequence number. Note that, there are 16 bits for the sequence number. Therefore, the sequence number gets reset after every 2^{16} packets of a session.
- **Session ID**
  The session ID is of 16 bits and carries the session ID of the data being transmitted.
- **Offset**
  The offset of 8 bits carries the information about the packets which are sent in another path. In other words, it shows the number of packets a node won't be receiving after a particular sequence number it received. For example, if the packets with sequence number 1 and 5 are sent to one node, and packets with sequence numbers 2,3,4 are sent via some other node, then the offset is set as 3 while



(a) Successful transmission

(b) Acknowledgment failure

(c) Packet failure

Fig. 4: Split window automatic repeat request.

sending packet with sequence number 5 and offset is set as 1 while sending packet with sequence number 2. In all the other cases it is set as 0. Offset value 0 indicates that the consecutive sequence numbered packets are sent to that node.

- **Reserved**
  4 bits are reserved for future use.
- **ACK**
  ACK of 1 bit is set when the packet is an acknowledgment packet.
- **NACK**
  NACK of 1 bit is set when a particular sequence number is lost in the transmission which a node is supposed to receive. The lost sequence number is set in the " universal sequence number" field. Upon receiving the NACK packet, the source resends the lost packet.
- **Routing Adv**
  This 1 bit is used for advertising the node details to neighboring nodes.
- **Routing Exch**
  A 1 bit field is used to exchange the routing table with neighboring nodes.
- **Received from address**
  Received from address is an 8-bit field which carries the address of the node from which the packet is received at a node.
- **Forwarding node address**
  Forwarding node address is an 8-bit field which holds the address of the node to which the packet must be transmitted/forwarded.

Here, we consider that each node in the network builds its own routing table based on the algorithm given in [6]. As we have used 8 bits for node ID, RMPMT currently supports 2^8 nodes in the network. Also, since there are 16 bits for universal sequence number, the protocol supports a maximum of 2^{16} fragments for a particular session.

The Quality of Service (QoS) is crucial in critical IoT applications such as healthcare, traffic safety and control, and Industrial applications [4]. To ensure that the RMPMT provides guaranteed packet delivery, we propose a unique acknowledgment (ACK) scheme called *Split Window Automatic Repeat Request* (SWARQ) which is capable of working with multiple path and multiple technology routing scenarios.

C. Construction of Routing Table

We implement the routing table as given in our previous paper [6]. Here we consider that there is one destination IGW to which there are multiple paths via multiple technologies through various intermediate SSGWs. At every SSGW, the routing table mainly consists of three entries - $nextHopID$,$hopCount$,$InterfaceID$. The $nextHopID$ represents the ID of the next SSGW to which the data has to be transmitted, $hopcount$ represents the number of hops to the destination IGW, and $interfaceID$ represents the technology (interface) through which the data has to be transmitted.

To begin the construction of routing table, the IGW first broadcasts its routing information to the nearest SSGWs via all the technologies. The nearest SSGWs will then update their routing table entries and further broadcast to other SSGWs via the interfaces they possess. For detailed explanation of the construction table we refer the reader to our earlier work [6].

D. Flow Control

Existing standard Automatic Repeat Request (ARQ) algorithms fail in multi-path multi-technology environment since the packets are transferred via different paths. This leads to starving at an intermediate node as it waits infinitely for a packet which is never transmitted to that node [14]. To avoid this we propose SWARQ where for every receiver there is a window of size 2^{n-p}, where n is number of bits in sequence number field and p is number of receivers. Window size is restricted to 2^{n-p} because the sum of window sizes of all receivers should not cross 2^{n-1} to discard invalid data packets correctly. Here, p must be in power of 2. Furthermore, the offset field represents the number of packets which a node will not be receiving and is set to $n-p$ bits, i.e., even though n bits are allocated for offset field, only $n-p$ bits will be utilized. For example, if there are two paths (two receivers) then $p=2$ and $n=4$, then for each receiver there is a sliding window of size 2^2 as shown in Figure 4a and offset field will be only of 2 bits.

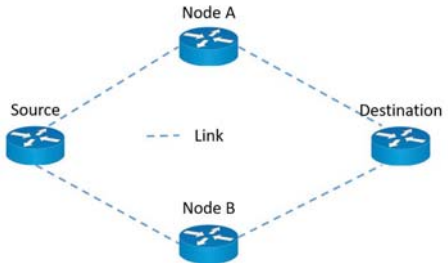


Fig. 5: Demo topology.

For a two path transmission from source X via intermediate nodes A and B to the destination (Figure 5), Figure 4a shows the flow of data for a successful transmission. As it can be seen, initially both sender window and all receivers' window will have sequence numbers 0-3. The data packet is represented by $D(seqNo,Offset)$ where $seqNo$ represents the sequence number of the packet and $Offset$ represents the offset value. Now, let us assume first packet with sequence number 0 is sent to node A. As there are no packets before this, offset is set to 0. Second packet with sequence number 1 is also sent to node A with offset value 0. This offset represents that the previous packet (with sequence number 0) is also sent to node A. Now, if packet with sequence number 2 is sent to node B, the offset value is set to 2. This represents that the previous two packets to sequence number 2 will not be sent to node B. If $seqNo-Offset$ value lies in the current window of the receiver, then it immediately slides window by two positions as shown in Figure 4a. Now, at the sender side, after packets 0 and 1 are sent to A, the sender window of B will be moved by two places. Furthermore, after sending D(2,2), D(3,0), and D(4,0) to node B, the sender window of node A is now split to neglect 2,3, and 4 sequence numbers (hence the name Split Window ARQ). Only after receiving acknowledgment will the sender move the sliding window of node A. Similarly the process continues which guarantees the data delivery.

Now, let us consider a scenario where acknowledgment will fail. Assume that the acknowledgment of packet 0 is lost as shown in Figure 4b. The sliding window at the sender side will not be slided. Now, when the acknowledgment of packet 1 is received, it will act as the cumulative acknowledgment and hence the sender will move the sliding window by two positions. Furthermore, if acknowledgment of packet 1 also fails, then once the timeout occurs at the sender side, the sender will resend the packet to the node for which the receiver node will resend acknowledgment. And only after that, the sender will slide the sliding window.

Similarly, let us consider the scenario where the transmission of packet itself fails as shown in Figure 4c. If transmission of packets D(0,0) and D(1,0) fails, and when the packet D(5,3) is sent to the node A, then the node understands that the 3 packets before sequence number 5 will not be received by it. However, $seqNo-Offset=5-3=2$ is in the current window of the node A. Therefore, it splits its own window

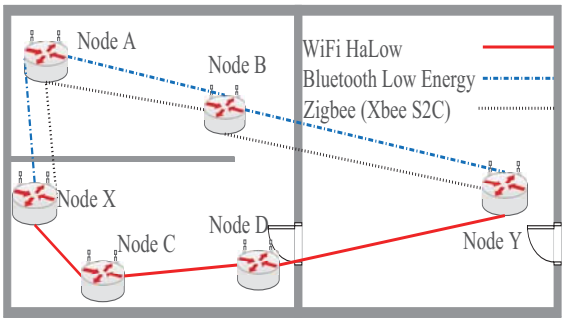


Fig. 6: Testbed floor plan (HPCN Lab, IIT Madras).

to discard three sequence numbers before 5 and accept the new packet and send a NACK(0) showing that it has not received the packet with sequence number 0. Once this is received by the sender, it will resend the packet 0 again, there by guaranteeing the delivery of the packets. One thing that has to be noted is that the difference between split windows cannot be greater than the maximum value the offset can hold (2^{n-p}). This will ensure that the invalid packets are discarded correctly.

These intermediate nodes will further transmit the received fragments towards destination. Since different fragments of a data are sent via multiple paths and multiple technologies, the destination node will rearrange all the fragments based on the universal sequence number which is never changed by any intermediate nodes.

E. Testbed

We have deployed a static network testbed consisting maximum of six nodes. Each node comprises (i) RPi as an SSGW (ii) BLE, ZigBee (ZB), and WiFi as radio access technologies. Figure 6 gives the floor plan of the testbed topology. Nodes X and Y are source and destination nodes, respectively. Node X has two paths to reach node Y. One of the paths to node Y from node X is established using BLE and ZB with nodes A and B as intermediate nodes. The other path has the connection of WiFi HaLow with nodes C and D as intermediate nodes. The distance between each node is between 1 meter and 9 meters. Note that we consider the data rate of WiFi HaLow in our experimental setup by limiting the data rate of the WiFi to 512 Kbps. In this paper, to demonstrate the effectiveness of our proposed RMPMT protocol, we show the results with 1 hop, 2 hop, and 3 hop testbed topology. We compare our results with traditional single path single technology routing in an IoT network. The native operating system of Raspberry Pi, an open source distribution of Debian called Raspbian [15] is used for the experiments.

The floor plan, location of Raspberry Pis, and the technologies associated with each Raspberry Pi of a 6-node topology used for demonstration in this paper are given in Figure 6.

The implementation of the proposed protocol is not straight forward using Raspberry Pi and to the best of our knowledge, there are no devices in the market which can convert packets from one protocol to another. Following are the challenges faced during the implementation of the protocol:

1) We used Raspberry Pi 3 as nodes for our protocol. It has limited USB ports to attach peripherals.
2) Zigbee module used in the experiment allows only unidirectional communication, and hence, we had to utilize two modules for each node to ensure bidirectional communication.
3) Zigbee module and USB adapter we used had a physical buffer size of 32 bytes, which forced us to limit packet size below 32 bytes to avoid further packet fragmentation by Zigbee module.
4) Existence of co-channel interference in Bluetooth technology due to personal digital assistants and other nodes.
5) Proprietary software (device drivers) of all 3 technology modules made it difficult to modify the existing behavior of respective wireless interface according to the need of experiment. Especially, Zigbee module source code was not available publicly.

F. Distribution of Node ID

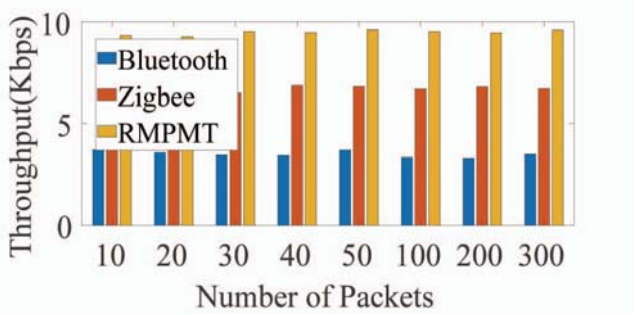
We have built the testbed according to the architecture shown in Figure 1. The responsibility of distribution of the node ID to all the SSGWs is given to the IGW. An IGW will assign node IDs to all the SSGWs from which it receives the data. In the proposed protocol, source and destination addresses are considered to be 8 bits, and therefore, each IGW can receive the data from 256 SSGWs. However, this can be modified to suit the need of the size of the network.

In a six-node topology shown in Figure 6, node Y is the destination node (which is considered to be an IGW). All the SSGWs will be transmitting data to node Y, and therefore it will be the responsibility of the node Y to distribute the node IDs to all the nodes in the testbed.

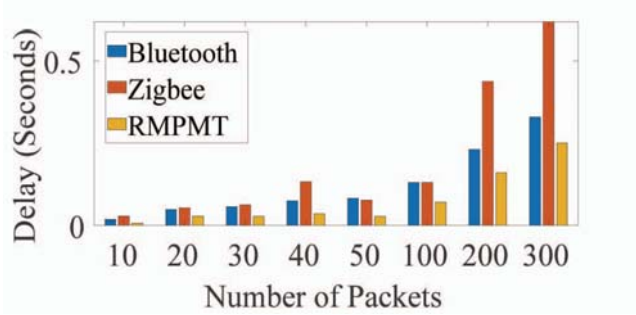
III. Results and Analysis

In this work, we conduct several experiments to test the effectiveness of the RMPMT on the IoT testbed built using RPi. We measure throughput and delay for various topologies identified within the testbed. We also measure the variation in throughput by varying the payload size. All the experiments are carried out on the testbed topology during night to avoid possible external interference from Bluetooth modules of personal digital assistants (PDAs).

Figures 7, 8, 9, and 10 show the results obtained from the testbed. As mentioned earlier, we consider 2-node topology where there is only one source and one destination node, 3-node topology where there is one intermediate node between source and destination, 4-node topology where there are two intermediate nodes between source and destination, and a 6-node topology as given in Figure 6. In the cases between 2-4 nodes topology, only BLE and ZB technologies are considered, where as for a 6-node topology BLE, ZB and WiFi HaLow technologies are considered. We compare our results with a network containing only one technology (such as BLE or ZB or WiFi HaLow) and only one path from source to destination.

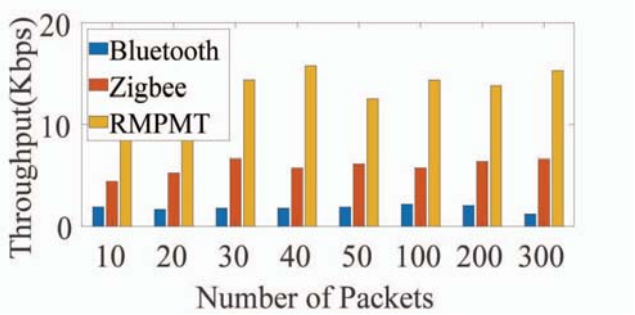


(a) Throughput in 2-node topology

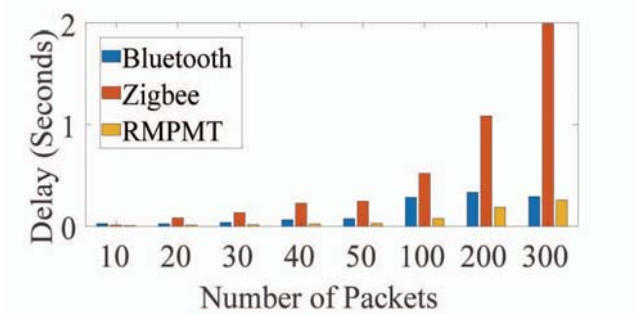


(b) Delay in 2-node topology

Fig. 7: Testbed results in 2-node topology.

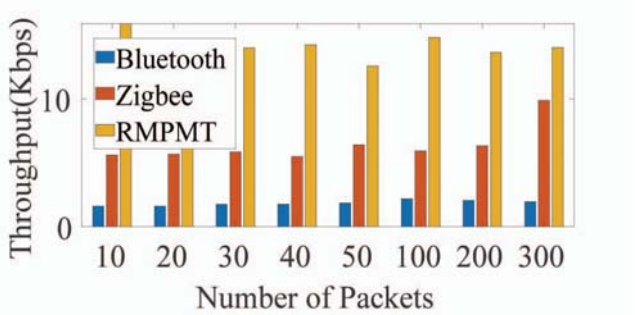


(a) Throughput in 3-node topology

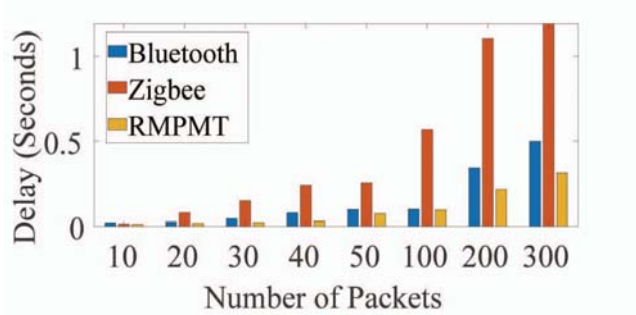


(b) Delay in 3-node topology

Fig. 8: Testbed results in 3-node topology.



(a) Throughput in 4-node topology



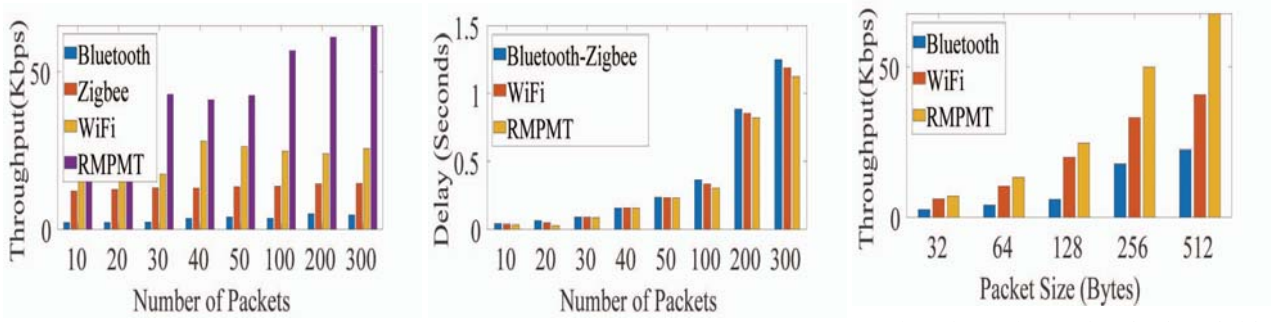
(b) Delay in 4-node topology

Fig. 9: Testbed results in 4-node topology.

We measure the performance of the network in terms of throughput and delay with respect to the number of packets in the network. Here, throughput is defined as the number of bits transmitted per second, and the delay is defined as the time difference between the data reaching the destination and data generated at the source.

Figures 7a and 7b show the throughput and delay in 2-node topology, respectively. Here multi-path is achieved by having two technologies in both the nodes, so that there are two paths (via two different technologies) between the nodes. As it be seen RMPMT significantly outperforms IoT network with only one technology both in terms of throughput and delay. Similarly, Figures 8a and 8b show the throughput and delay in 3-node topology, respectively. When compared to 2-node topology the throughput of the system is high as more data is transmitted in the network and delay is also high as contention for channel increases, but in any case, RMPMT outperforms the network with single technology.

Similarly Figures 9a and 9b show throughput and delay in 4-node topology. Clearly, RMPMT outperforms the network with single technology both in terms of throughput and delay. Figures 10a and 10b show throughput and delay of the network



(a) Throughput in 6-node topology

(b) Delay in 6-node topology

(c) Throughput when packet size is varied in 6-node topology

Fig. 10: Testbed results in 6-node topology.

with 6 nodes whose topology is shown in Figure 6. As it can be clearly seen, the throughput is maximum and the delay is minimum in case of RMPMT when compared to network with single technology. We also measure the throughput in 6-node topology by varying the packet size between 32 bytes to 512 bytes as shown in Figure 10c. As it can be seen, as the packet size increases, the payload also increases which in turn increases the throughput of the system. Clearly, again RMPMT outperforms the network with single technology.

IV. Conclusion

The world is moving towards the IoT enabled environment in every possible area. To make this a reality, multiple technologies are being designed and tested. In this paper, we proposed a new RMPMT protocol for POR which exploits multiple paths and multiple technologies available in an IoT network. Furthermore, we proposed SWARQ to ensure reliable data delivery. We provided the proof of concept for the proposed protocol for POR by implementing the same on a testbed using Raspberry Pi. By sending real-time data, we showed the effectiveness of the proposed protocol in terms of throughput and delay of the system.

Due to the limited number of interfaces to plug the peripherals and unavailability of proprietary softwares, implementing the protocol in a large scale environment was difficult. Therefore, performance when the network is considerably large is our immediate future concern. Furthermore, utilizing different channels at once may lead to increased energy consumption. Reducing energy consumption in such a scenario will be considered in our future work.

V. Acknowledgments

This research work was supported by the Department of Science and Technology (DST), New Delhi, India.

References

[1] I. Cisco Systems. (2019) Cisco Visual Networking Index: Forecast and Methodology. Accessed: 2019-03-06. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.html

[2] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[3] E. L. Ciemins, A. Arora, N. C. Coombs, B. Holloway, E. J. Mullette, R. Garland, S. (Walsh) Bishop-Green, J. Penso, and P. J. Coon, "Improving blood pressure control using smart technology," *Telemedicine and e-Health*, vol. 24, no. 3, pp. 222–228, 2018, pMID: 28930497. [Online]. Available: https://doi.org/10.1089/tmj.2017.0028

[4] P. Schulz, M. Matthe, H. Klessig, M. Simsek, G. Fettweis, J. Ansari, S. A. Ashraf, B. Almeroth, J. Voigt, I. Riedel, A. Puschmann, A. Mitschele-Thiel, M. Muller, T. Elste, and M. Windisch, "Latency Critical IoT Applications in 5G: Perspective on the Design of Radio Interface and Network Architecture," *IEEE Communications Magazine*, vol. 55, no. 2, pp. 70–78, February 2017.

[5] Intel, "Intel IoT Gateways," 2019, https://www.intel.in/content/www/in/en/internet-of-things/gateway-solutions.html, Last accessed on 2019-04-23.

[6] F. Singh, J. K. Vijeth, and C. S. R. Murthy, "Parallel Opportunistic Routing in IoT Networks," in *IEEE Wireless Communications and Networking Conference*, April 2016, pp. 1–6.

[7] D. Ugrenovic and G. Gardasevic, "CoAP protocol for web-based monitoring in IoT healthcare applications," in *Telecommunications Forum*, Nov 2015, pp. 79–82.

[8] S. Chandrashekar, A. Maeder, C. Sartori, T. Hohne, B. Vejlgaard, and D. Chandramouli, "5G Multi-RAT Multi-Connectivity Architecture," in *IEEE International Conference on Communications Workshops*, May 2016, pp. 180–186.

[9] S. Abdullah and K. Yang, "An energy-efficient message scheduling algorithm in Internet of Things environment," in *International Wireless Communications and Mobile Computing Conference*, July 2013, pp. 311–316.

[10] S. K. Singh, T. Das, and A. Jukan, "A Survey on Internet Multipath Routing and Provisioning," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2157–2175, 2015.

[11] A. Taha, R. Alsaqour, M. Uddin, M. Abdelhaq, and T. Saba, "Energy Efficient Multipath Routing Protocol for Mobile Ad-Hoc Network Using the Fitness Function," *IEEE Access*, vol. 5, pp. 10 369–10 381, 2017.

[12] M. Z. Hasan, H. Al-Rizzo, and F. Al-Turjman, "A Survey on Multipath Routing Protocols for QoS Assurances in Real-Time Wireless Multimedia Sensor Networks," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1424–1456, 2017.

[13] V. J. Kotagi, F. Singh, and C. S. R. Murthy, "Adaptive Load Balanced Routing in Heterogeneous IoT Networks," in *IEEE International Conference on Communications Workshops*, May 2017, pp. 589–594.

[14] R. Comroe and D. Costello, "ARQ Schemes for Data Transmission in Mobile Radio Systems," *IEEE Journal on Selected Areas in Communications*, vol. 2, no. 4, pp. 472–481, July 1984.

[15] Raspberry Pi Foundation, "Raspbian OS," 2019, https://www.raspberrypi.org/downloads/raspbian/, Last accessed on 2019-03-09.