I. J. Computer Network and Information Security, **2017, 7, 36-44**
Published Online July 2017 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2017.07.05

Comparative Performance Analysis between nRF24L01+ and XBEE ZB Module Based Wireless Ad-hoc Networks

Himadrinath Saha
Department of Computer Science & Engineering, Institute of Engineering & Management, Kolkata, India
E-mail: himadri@iemcal.com

Shashwata Mandal
Department of Computer Science & Engineering, Institute of Engineering & Management, Kolkata, India
E-mail: shashwata275@gmail.com

Shinjan Mitra
Department of Computer Science & Engineering, Institute of Engineering & Management, Kolkata, India
E-mail: shinjanxp@gmail.com

Soham Banerjee
Department of Computer Science & Engineering, Institute of Engineering & Management, Kolkata, India
E-mail: soham2008xyz@gmail.com

Urmi Saha
Department of Computer Science & Engineering, Institute of Engineering & Management, Kolkata, India
E-mail: urmisaha.88@gmail.com

***Abstract*—**Among the common wireless communication modules like Bluetooth and Wi-Fi, XBee modules are embedded solutions providing wireless communication standard with self-healing mesh networks, which has longer range than Bluetooth and lower power consumption than Wi-Fi. An alternative to the XBee radio modules is nRF24L01+ radio modules which are cheap and powerful, highly integrated, ultra-low power (ULP) 2Mbps RF transceiver ICs for the 2.4GHz ISM (Industrial, Scientific, and Medical) band. In this paper, performances of nRF24L01+ modules have been analyzed and compared with that of XBee ZB modules in wireless ad-hoc networks. The performance metrics for the analytical study are - 1) Throughput measurement, 2) Mesh routing recovery time and 3) Power consumption. This work has revolved around an open source library released by the developer, tmrh20 which builds a complete TCP/IP suite on top of the nRF24L01+ modules.

Index Terms—Ad hoc networks, Multi-hop networks, Network throughput, nRF24L01+, Wireless mesh networks, Zigbee.

I. Introduction

A wireless ad-hoc network refers to a group of nodes that interconnect in an ad-hoc fashion without the need of a supervising or permanent infrastructure. Wireless ad-hoc sensor networks [1] are used for surveillance, widespread environmental sampling, security and health monitoring. They find particular application in scenarios where infrastructure-based networks cannot be easily deployed, due to constraints of the environment; terrain or where pre-existing network infrastructure cannot be rapidly deployed, especially if the nodes are mobile. Zigbee [2] and Bluetooth [3] are the prevalent technologies which cater to ad-hoc network solutions. In addition to these, there are other technologies [17] which provide limited range but can be used in conjunction with these, especially in Internet of Things applications [23]. Although quite robust, the Zigbee modules are also expensive when compared with other standards such as Bluetooth, RFID or NFC; hence they are unfeasible to be deployed in scenarios such as wireless sensor networks where the number of nodes exists in the order of thousands. On the other hand, the aforementioned cheaper alternatives either do not support the creation of ad-hoc networks or are limited by their area of coverage. But there has been a new addition to the family of commercial wireless technologies called nRF24L01+, which provides a cheaper and a more configurable alternative.

In this paper, we aim to compare and contrast the efficiency and performance of nRF24L01+ against the current widely using wireless ad hoc standard i.e. XBee

[4], a module based on the Zigbee standard. The paper focuses on the performance of these two devices when used in a wireless ad hoc sensor network. Since a wireless ad-hoc network is intended to be deployed in scenarios where there is no pre-existing network infrastructure or readily available power supply, we have to consider several factors such as power consumption, routing protocol efficiency and interoperability with existing network protocols while making the choice of wireless media. Some technologies we have mentioned such as Bluetooth (based on IEEE 802.15.1 and IEEE 802.15.2) and Zigbee, based on IEEE 802.15.4[5] are all plagued by bottlenecks such as network throughput, new node discovery, mesh routing recovery delay and so on. In this paper, we shall study the nRF24L01+ as a medium for wireless ad-hoc sensor networks, and also compare its performance as compared against Zigbee with respect to the IEEE 802.15.4 PHY specification.

The nRF24L01+ [6] integrates a complete 2.4GHz RF transceiver, RF synthesizer, and baseband logic including the Enhanced ShockBurst™ hardware protocol accelerator supporting a high-speed ubiquitous SPI (Serial Peripheral Interface) for the application controller. It is thus able to operate in conjunction with a variety of microcontrollers without the addition of any external hardware, as long as they support the SPI protocol. A corollary to this is that it has to be used in conjunction with a microcontroller since it can't work on its own resulting in a bulky unit; not an ideal fit for a wireless sensor network. The nRF24L01+ has gained a lot of popularity due to its simplicity in interacting with most standard microcontrollers, wide bandwidth at a high data rate and availability at a low price of under $3. Since it works on the 2.4GHz ISM band, it is possible to emulate standard radio protocols working at that bandwidth such as Bluetooth Low Energy (BLE) or the 2.4GHz IEEE 802.11 Wi-Fi standards using this transceiver with a supporting microcontroller.

Although the nRF24L01+ may appear overtly similar to the XBee communication modules based on the Zigbee protocol, there are some important differences. The Zigbee protocol [7], based on top of IEEE 802.15.4 PHY and MAC specifications, specifies two physical layers based on direct sequence spread spectrum (DSSS) techniques, with data rates varying between 10kbps to 40kbps. The nRF24L01+ uses GFSK modulation [8], with data rates from 250kbps to 2Mbps. Also, no MAC specification is provided in the nRF24L01+, which must be provided by the accompanying microcontroller. However, even with the accompanying microcontroller, the nRF24L01+ is available at one-third the cost of an XBee module. The nRF24L01+ also allows a greater distance of operation than XBee modules. With an external antenna, the range can be nearly 500m at maximum power, compared to the 40m operating range of the XBee modules.

The paper is written in conjecture with a publication [9] which as the name suggests provides an overview on the performance of XBee when subjected to a similar environment. It provides a detailed analysis of XBee on the following parameters- received signal strength indicator (RSSI), network throughput, packet delay, mesh routing, recovery time and energy consumption. Hence in this paper, we shall focus on these parameters in detail and mention differences in measuring certain parameters due to the hardware architectural difference between the two devices.

II. Experimental Setup

The Nordic nRF24L01+ IC are commercially available embedded in radio modules which come in various form factors [10]. We have made use of the low-cost modules with a built-in antenna for our experiments. Certain hardware modifications were necessary to make it easier to interface with these modules as illustrated in the next section. Also, we have made use of several open source libraries in our programming.

A. Hardware Setup

We have used the nRF24L01+ 2.4GHz antenna wireless data transmission module (hereby referred to as the nRF) for conducting our experiments. Each module comprises a patch antenna, unlike the XBee which is equipped with a wire antenna [4]. The nRF operates on the 2.4 GHz ISM band with channel spacing of 1 MHz which leads to 125 possible channels from 2.4 GHz to 2.525 GHz [10]. Wi-Fi devices work on most of the lower channels hence it is advisable to use the highest 25 channels for projects using the nRF. Here, we have used channel 97 for all the experimental procedures.

The nRF module needs to be interfaced with a microcontroller unit using SPI since it is not programmable. We have used the very popular open source Arduino platform for controlling and sending messages using the nRF modules. We assembled our own custom nRF shield (Figure 1) which fits onto the Arduino Uno conveniently connecting the corresponding pins of the nRF to the Arduino [10], [11].

The nRF module suffers from occasional intermittent operation attributed to insufficient current or noise on the 3.3V power supply [10], especially when powering it using the Arduino Uno since the Arduino cannot supply

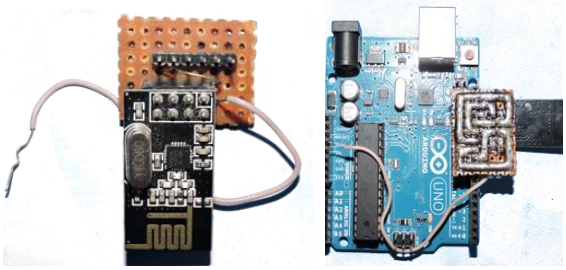

Fig.1. Custom Built nRF shield with nRF Module Plugged in

more than 50 mA current. Keeping this in mind, the RF24 library used in our experiments provides several configurable power settings: RF24_PA_MIN,

RF24_PA_LOW, RF24_PA_HIGH and RF24_PA_MAX. Hence all experimental procedures are conducted setting the nRF module to operate at the lowest of the available 4 power settings. In this scenario, the wireless transmission range of the nRF reduces considerably to less than 3m. All experimental tests were carried out in an indoor environment with a 1m distance between 2 nRF modules.

B. *Software Setup*

1) *Library*

We have made use of the excellent RF24 library family developed by TMRH20 for Arduino and Raspberry Pi which effectively builds a TCP/IP suite (Table 1) on top of the nRF module. For our experiments, we have made use of the RF24 [12], RF24 Network [13] and the RF24 Mesh [14] libraries for Arduino.

Table 1. Equivalence between TMRH20 libraries and TCP/IP suite

TMRH20 RF24 Library	TCP/IP Suite Equivalent
RF24Mesh	Application Layer(DHCP)
RF24Ethernet	Transport Layer
RF24Network	Network/Internet Layer
RF24	Link Layer

2) *Algorithm*

The nRF chip implements a variation of the ANT protocol (up to transport layer). Routing of packets in the multihop network is handled by the library [13] which implements an algorithm similar to [16].

We have designed a pingpair setup [18], [19] on the Arduino enclosed in an infinite loop. A pingpair setup is an experimental setup where a Node A sends a message with a predefined size to Node B and Node B echoes back to Node A. Node A then calculates the round trip time between sending the original message and receiving the echoed message.

3) *Measurements*

For each size of a message transmitted, measurements have been taken three times for varying bit rates (250 KBPS, 1 MBPS, 2 MBPS). The results of the above algorithm are output on the Arduino serial monitor, from where we take 10 readings and calculate the average.

III. Experimental Measurements and Results

A. *Throughput Measurement*

Throughput is expressed as the volume of data being sent successfully in unit time. It is a measure of how fast or how slow the network being studied is. Mathematically, throughput can be expressed as:

$$Throughput = \frac{\text{number of bytes sent}}{\text{total transmission time}(\text{in sec})} \quad (1)$$

We record the throughput of the nRF modules at different bit rates and payload sizes, then plot the measurements on charts to analyze the best performance criteria.

1) *Throughput measurement for different payload size in point-to-point (PtP) link at different data rates*

For all point to point transfer test experiments we have used only the RF24 library. The library exposes only the bare transmission functionality through member functions of the RF24 class. For point-to-point communication, tests were carried out by sending messages of size 4 Bytes through 32 Bytes with an increment of 4 Bytes. The network comprised of Arduino with attached nRF modules as illustrated in Figure 2.

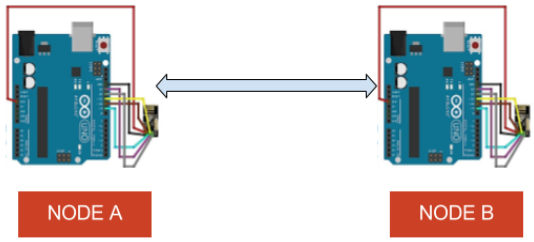

Fig.2. Network Layout of Point-to-Point Link

The nRF modules communicate using a set of logically shared pipes which enables the modules to simultaneously write to 1 pipe and read from 6 pipes. Each pipe has a unique 40-bit address. The RF24 library provides a function to open one writing pipe and up to 6 reading pipes. Pipes 0 and 1 can store a full 5-byte address. Pipes 2-5 will technically only store a single byte, borrowing up to 4 additional bytes from pipe #1 per the assigned address width. For our experiments we use 2 pipes to send and receive data between 2 nodes:

- 0xABCDABCD71 – writing pipe for Node 1 and reading pipe for Node 2
- 0x544D52687C – writing pipe for Node 2 and reading pipe for Node 1

Every successful message reception is followed by an acknowledgment sent by the receiver node. There is also the provision to send ACK payloads i.e. optional messages piggybacked via the ACK frame. We have taken 2 sets of measurements for Point-to-Point transmission - one set where the message is echoed back via the ACK payload and another where messages are echoed via a new data frame (ACK payload disabled). Recorded measurements have been illustrated in Figures 3 -4.

Observations: The throughput measurement in PtP link at different data rates and payload size provides the following insights:

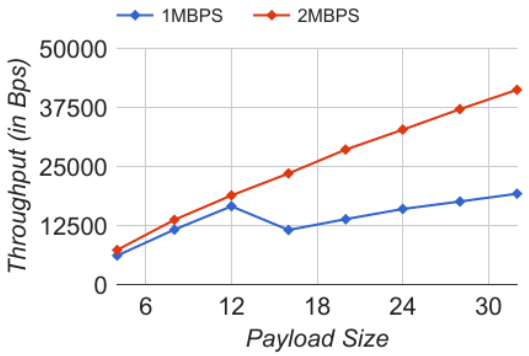

Fig.3. Throughput when ACK Payload is Enabled

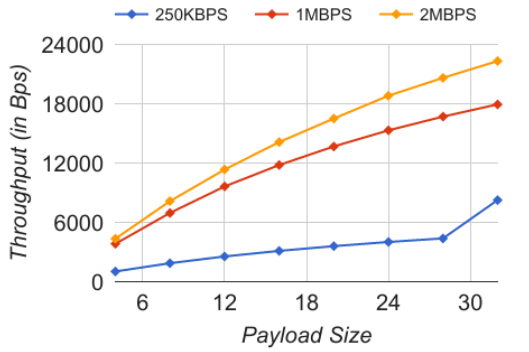

Fig.4. Throughput when ACK Payload is Disabled

1. Throughput increases with increase in bit rate and message size. This is an expected behavior in wireless networks since lesser overhead is required per data byte.
2. ACK payloads do not work at 250 setting.
3. While testing pingpair with ACK payloads enabled at 1 MBPS bit rate it was observed that although messages were received by Node 2, the echoed message was never received when message size exceeded 4 Bytes. Hence we reconfigured the receiver node to echo back only ACK payloads of size 4 Bytes irrespective of the size of the message received. A similar situation was observed when bit rate was set to 2MBPS after message size exceeded 16 Bytes. In this case, we set an upper limit of 16 Bytes for ACK payloads. To summarize, the ACK payloads are limited to:

   - 16 bytes for 2 MBPS
   - 4 bytes for 1 MBPS

4. There is an upper limit of 32 bytes for each message sent using RF24 layer. When we tried to send messages larger than 32 bytes, only the first 32 bytes of the message was sent and echoed back.
5. At 250 KBPS data rate without ACK payload, an interesting observation is found where the delay is found to increase with an increase in payload size till 28 bytes and then falls back remarkably to that corresponding to the payload size of 4 bytes.

Comparison with Xbee: It was observed that a throughput of 5.4 kbps at a baud rate of 115200 bps was achieved using maximum offered payload in an XBee PtP network [9]. We have recorded a maximum throughput of 41.2 kbps, significantly faster than the XBee network However it has to be taken into account that the XBee network is a routed network, i.e. the nodes perform routing functions while transmitting data. Even though there are only 2 nodes in the network, the routing process contributes significant overhead to the network, while our nRF network is purely Point-to-Point free from routing functions. Even for the nRF network, the maximum throughput achieved is far lower than the transmission rate of 2 Mbps. Nonetheless, nRF outperforms XBee nodes even at its lowest transmission rate of 250 kbps (which is the same as the transmission rate of XBee network) with a throughput of 8.3 kbps.

2) Throughput measurement for different payload size in multihop link at different data rates

XBee modules come in multiple form factors and editions but all of them use routing protocols based on the Zigbee protocol on top of the IEEE 802.15.4 standard. These protocols denote each XBee node as either a Coordinator, Router or End device, thereby creating a logical hierarchy for communication. XBee nodes can be configured beforehand to communicate with only a certain set of other nodes or to set up a link with any node of a particular type in its vicinity.

The multi-hop network realized through RF24Network library is slightly different compared to the XBee modules. Instead of there being 3 types of nodes there is only one Master in the whole network and the rest are Child Nodes. This protocol is compatible with IEEE 802.15.4, which specifies that two types of network nodes must be present: full-function devices (FFD) serving as the coordinator of a personal area network or as a common node, and reduced-function devices (RFD) which can only communicate with FFDs and can never act as coordinators. The network is arranged in a Tree Topology unlike current multipath routing protocols discussed in [22].Here the Master is the base, and all other nodes are children either of that node or of another. Unlike a true mesh network, multiple nodes are not connected together, so there is only one path to any given node. Each node logically connects to one parent node and routes all communication through it.

In case only the RF24Network library is used, then we need to design the tree topology ourselves and allocate each node its corresponding address within the tree. However if RF24Mesh library is used in conjunction with RF24Network, then we need only allocate a unique ID to each node irrespective of its position in the tree, and the library will take care of assigning each node a network address (and hence designing the topology) depending on its relative distance to other nodes and the master.

Tests were carried out by sending messages of size 4 Bytes to 32 Bytes with increments of 4 Bytes, and then from 32 Bytes to 128 Bytes with an increment of 16 Bytes

since after 32 Bytes, increments of 4 Bytes did not yield any significant change. The network topology is illustrated in Figures 5 – 7 while the corresponding recorded measurements are illustrated in Figures 8 – 10.

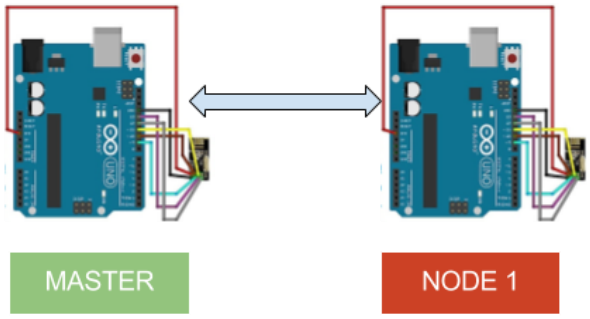

Fig.5. Throughput versus Payload Size for Transmission in 1 hop Network (M-N)

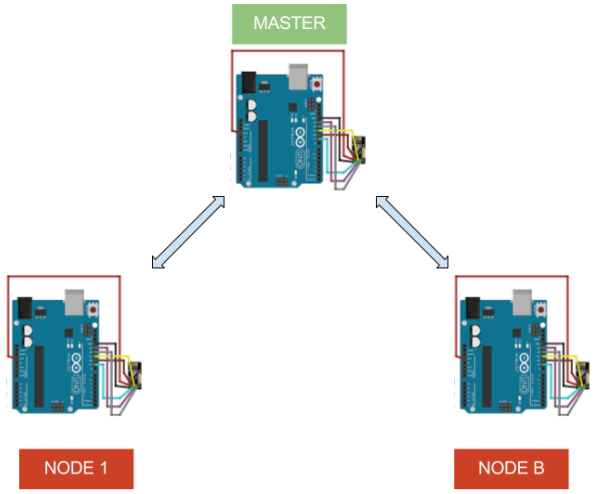

Fig.6. Throughput versus Payload Size for Transmission in 2 hop Network (N-M-N)

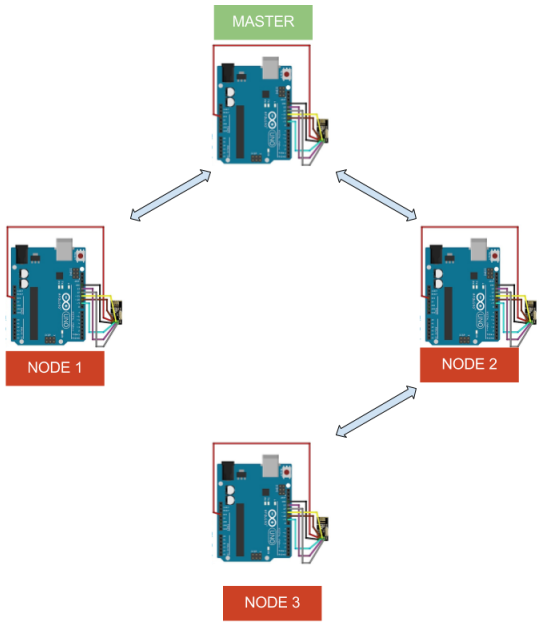

Fig.7. Throughput versus Payload Size for Transmission in 3 hop Network (N-M-N-N)

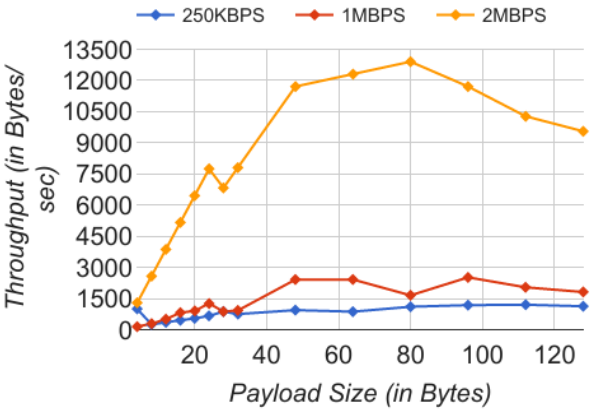

Fig.8. Throughput versus Payload Size for Transmission in 1 hop Network (M-N)

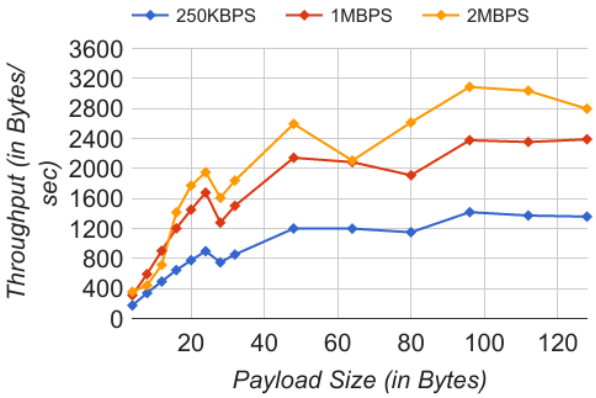

Fig.9. Throughput versus Payload Size for Transmission in 2 hop Network (N-M-N)

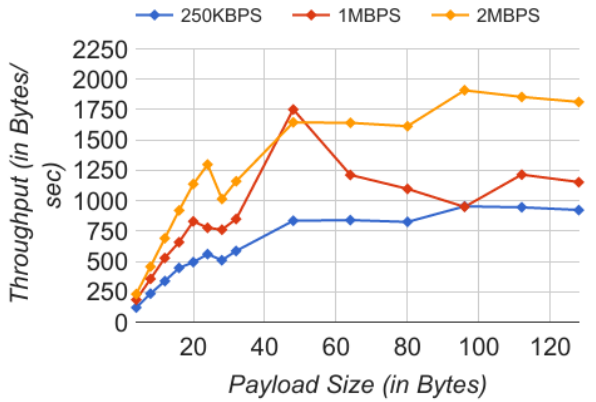

Fig.10. Throughput versus Payload Size for Transmission in 3 hop Network (N-M-N-N)

Observations: The throughput measurement in multihop topology at different data rates and payload size provides the following insights:

1. For all topologies except one, there is an observable dip in the throughput characteristics at a message size of 32 Bytes and again at 64 Bytes. Although debatable, this may be attributed to the fact that

these message sizes are multiples of the maximum data length of a single RF24 frame. Hence at these points, an extra overhead is added to handle the next data frame of 32 Bytes. However, since no such change is observed at the next multiple (96 Bytes), we may infer that from the 3rd frame onwards, frame overhead is insignificant compared to the actual message size. There is no strict relation between the data rate and throughput since on certain occasions a message transmitted at a higher data rate takes longer to be received than a similar sized message sent at a lower data rate.

2. We make an interesting and unexpected observation while comparing delays for different topologies keeping data rate constant. The delay for a message echoed via the N-M topology is greater than that for a message echoed via N-M-N topology, when the data rate is either 250 KBPS or 1 MBPS. This is illustrated in Fig. 11 for a message size of 128 Bytes. This is probably because in the N-M topology the echoing node is the master node which performs additional routing functions. We can simplify the Master Node's code block to two functions:

So effective delays for corresponding topologies are:

$$Delay_{N-M} = 1RTT + T(mesh.update) + T(echomessage) \quad (2)$$

$$Delay_{N-M-N} = 2RTT + T(echomessage) \quad (3)$$

where T(function) refers to the time taken to execute the function. Hence we can conclude that for data rate 250 KBPS or 1 MBPS, 1RTT < T (mesh.update()), and the converse holds true for a data rate of 2 MBPS.

Comparison with Xbee: The immediate difference noticed while comparing the performance of nRF and XBee routed networks is that the nRF performs considerably poor when routing algorithms are deployed even for a 2 node scenario. Hence the nRF offers us 2 alternatives when implementing a 2 node network – routed or PtP and most often one would choose the PtP version to gain higher throughput. The XBee does not provide any PtP alternative, it being inherently routed.

For the XBee, throughput decreases *as* $(1 / n_{hops})$ where n_{hops} is the number of hops traveled by a packet to reach its destination [9]. We do not observe any such a relation for multihop nRF networks, where peak performance does not monotonically change with reference to packet length. Hence it is difficult to establish any form of relation between throughputs of the different multihop scenarios. Also, the nRF has been observed under multiple transmission speeds while the XBee have been recorded only at a single serial interface speed.

Comparing absolute values of peak throughput it is evident that even a routed single hop nRF network is almost twice as fast (12.9kbps) as compared to the corresponding XBee network. In the 2-hop and 3-hop networks, nRF performs slightly better although the difference is almost negligible.

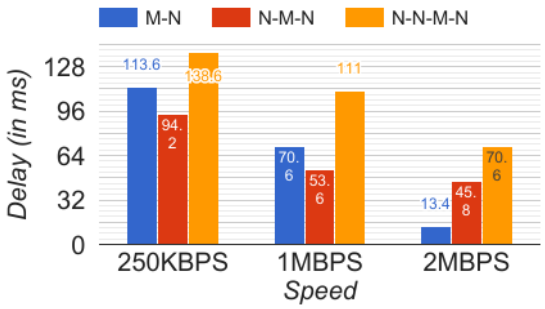

Fig.11. Comparison of Multihop Delays at Constant Payload (128B)

B. *Mesh Routing Recovery Time*

Mesh routing recovery time refers to the time taken to reconnect a node to the mesh after it gets disconnected. For this test, we programmed the Arduino to measure the time instant when it gets disconnected from the mesh and again once it reconnects and outputs the difference to the serial monitor. It was observed that the nodes frequently get disconnected from the mesh without external intervention, and automatically reconnect thanks to the RF24Network library. We recorded 60 instances of disconnection and reconnection as illustrated in Table 2:

Table 2. Mesh Routing Recovery Time (in ms)

Minimum	Maximum	Average
91	31590	3667.777778

C. *Power Consumption*

The nRF24L01+ is a low power device; with 4 different transmit power levels out of which we have used the lowest power configuration for all our experiments. We conducted a separate test at this power level to quantitatively measure how much power is consumed by the nodes. However, in our measurements, the power consumed by the Arduino is not included. The experimental setup was identical to that of III.A.1 at 250KBPS air data rate, with explicit radio power up and power down commands before and after each send cycle.

For conducting this test we set up a 1 Ohm shunt resistor in series with the supply to the nRF. By measuring the potential difference across the resistor leads, we were able to calculate current drawn by the nRF. We programmed another Arduino to measure this potential difference by feeding the supply voltage to AREF pin and then measuring the analog voltage at the other end of the resistor. Since Arduino provides resolution of 10 bits, we obtained an ultimate resolution of 3300/1024 = 3.22mV / level.

It was observed that during operation, nRF consumed between 12.88mA and 16.1 mA, as per our resolution of observation. This amount of current draw was steady

across the entire duration of observation (10 minutes). Although the current draw is consistent with the values set in the nRF datasheet [8] (11.3mA TX at 0dBm output power and 13.5 RX at 2MBPS air data rate), it does not explain why the current draw remained constant when data was being transferred only at an interval of 1 second. The current consumption of the nRF should be reduced to 26uA (.026mA) between sending and in full power down mode, the radio should consume approximately 900nA (.0009mA).

Comparison with Xbee: It is difficult to compare the power consumption of nRF with an XBee because their operating environments are different. The nRF requires a microcontroller in conjunction with which it operates, and which contributes to the overall power consumption of the unit. However, if we consider only the power consumed by the nRF module, we arrive at a linear relation between battery lifetime and battery capacity.

$$L \approx C \div 348 \; days \tag{4}$$

where C is the battery capacity and L is the lifetime.

Even a 2000mAh battery would last a mere 5 days when using the RF24 library, while the XBee lasts for years [9]. However, the nRF chip on its own is much more energy efficient as illustrated in [21].

IV. Conclusion

In this study, we have analyzed all the parameters mentioned in the introduction, focusing on the stark differences in the performance of nRF24L01+ against XBee. nRF24L01+ could provide a better throughput compared to XBee in almost all scenarios especially in point-to-point communication where it outshines XBee by a country mile. For multihop networks as well, the rate of transmission remains considerably higher.

One parameter which we had mentioned but could not analyze is the RSSI value. This happens to be a drawback of the nRF transceiver itself in that it does not provide any means of reading the received signal strength value. The only measurable quantity related to signal strength is Received Power Detector (RPD). As mentioned in section 6.4 of [8], it is positive triggered at received power levels above -64 dBm on the receiving RF channel. This does not offer any insight as to the relation between distance and received signal strength that we wanted to measure.

The nRF modules, particularly the RF24 family of libraries hold up quite well and in most cases outshine their more popular cousins, the XBee. It can be safely said that when compared in terms of speed and cost, nRF is the far better alternative. However, we observed that the nRF consumes significantly more power than XBee either because of inefficient sleep cycles or due to constant current draw by active and passive components in the external circuitry. When coupled with a microcontroller unit, the power consumption is far too high to be considered viable in sensor networks. There is a need to develop more energy efficient routing protocols to support the hardware.

It is possible to dramatically improve the power consumption by enabling dynamic sleep cycles. The nRF modules have a typical current draw of the order of 22 μA and 300 μA in its standby-I and standby-II modes respectively [8]. However, during our experiments, we were unable to activate the sleep modes dynamically during operation. It seemed that due to mesh routing functionality being operational all the time, it was not entering sleep mode at all. Solutions such as those illustrated in [20] may be implemented in the future to enable dynamic sleep cycles while preserving the mesh routing capabilities of the device.

It is also possible to improve the performance of the routing algorithms as depicted in [24] and [26]. Since we have implemented the open source code from the RF24 repositories we have not been able to introduce such energy-efficient routing techniques. There is a huge scope to improve on the routing algorithms used including the introduction of smart ant-colony optimization techniques [25] [27].

The radio, as well as well its supporting libraries have been under focus for quite some time now, however, in terms of development, it is still in its nascent phase. Nevertheless, its impact in the field has been nothing short of revolutionary because of the raw range and bandwidth it provides on paper at affordable prices. However, it remains to be seen whether with improvements in the sleep cycle implementation in the RF24 library or with a better alternative to the RF24 family, the nRF may be indeed considered a cheap viable replacement for XBee in wireless ad-hoc networks.

References

[1] Xiao, Yang, and Yi Pan. Emerging Wireless LANs, Wireless PANs, and Wireless MANs: IEEE 802.11, IEEE 802.15, 802.16 Wireless Standard Family. Yang Xiao & Yi Pan. John Wiley & Sons, 2009.

[2] Digi International Inc. “Zigbee Wireless Standard”, 2013, Available: http://www.digi.com/technology/rf-articles/wirelesszigbee

[3] Bluetooth SIG, Inc. “Bluetooth core specification”, 2016, Available: https://www.bluetooth.com/specifications/bluetooth-core-specification.

[4] Digi International Inc., “XBee®/XBee-PRO® RF Modules”, product datasheet, available at https://www.sparkfun.com/datasheets/Wireless/Zigbee/XBee-Datasheet.pdf.

[5] Zheng, Jianliang, and Myung J Lee. "A comprehensive performance study of IEEE 802.15. 4." 49 (2004).

[6] Nordic Semiconductor, “nRF24L01 overview” available at http://www.nordicsemi.com/eng/Products/2.4GHz-RF/nRF24L01 (2007).

[7] ZigBee Alliance “Zigbee Specification – 05-3474 Rev 20”, Sep. 2012 http://www.zigbee.org/wp-content/uploads/2014/11/docs-05-3474-20-0csg-zigbee-specification.pdf.

[8] Nordic Semiconductor, “nRF24L01 Single Chip 2.4GHz Transceiver Product Specification”, July 2007, available at http://www.nordicsemi.com/eng/content/download/2730/34105/file/nRF24L01_Product_Specification_v2_0.

pdf.
[9] Piyare, Rajeev, and Seong-ro Lee, "Performance analysis of XBee ZB module based wireless sensor networks", International Journal of Scientific & Engineering Research 4.4 (2013): 1615-1621.
[10] Terry King, Mary Alice Osborne, Jun Peng, Barry King and other contributors, "Nrf24L01-2.4GHz-HowTo", wiki page available at https://arduino-info.wikispaces.com/Nrf24L01-2.4GHz-HowTo
[11] Akshay Jha, "Wireless Remote Using 2.4 Ghz NRF24L01: Simple Tutorial Using of NRF24L01 & Arduino", blog post, http://www.instructables.com/id/Wireless-Remote-Using-24-Ghz-NRF24L01-Simple-Tutor/
[12] TMRH20, "Optimized High Speed NRF24L01+ Driver Class Documentation V1.0", Sep. 2016, library documentation, available at http://tmrh20.github.io/RF24/
[13] TMRH20, "Newly Optimized RF24Network Layer v1.0.7", Sep. 2016, library documentation, available at http://tmrh20.github.io/RF24Network/
[14] TMRH20, "RF24Mesh V1.0.3b" Nov. 2015, library documentation, available at http://tmrh20.github.io/RF24Network/
[15] Dynastream Innovations Inc. "ANT Message Protocol and Usage", July 2007, available at https://www.sparkfun.com/datasheets/Wireless/Nordic/ANT-UserGuide.pdf
[16] Rohit Singh, Himadri Nath Saha, Debika Bhattacharyya and Pranab Kumar Banerjee, "Administrator and Fidelity Based Secure Routing (AFSR) Protocol in MANET", Journal of Computing and Information Technology, Vol. 24, No. 1, March 2016, 31–43
[17] Jaideep Kaur, Kamaljit Kaur,"Internet of Things: A Review on Technologies, Architecture, Challenges, Applications, Future Trends", International Journal of Computer Network and Information Security(IJCNIS), Vol.9, No.4, pp. 57-70, 2017.DOI: 10.5815/ijcnis.2017.04.07
[18] Shinjan Mitra, "String pingpair with and without ack", Aug 2016, https://github.com/shinjanxp/RF24/commit/e483d9ec0eb7019c22f41e8030fcbfb274ed1c73
[19] Shinjan Mitra, "Mesh pingpair master and node", Aug 2016, https://github.com/shinjanxp/RF24Mesh/commit/1e00bdcafdd2f511b1ae363af48c27e42160c793
[20] Kaebeh Yaeghoobi S.B., M.K. Soni, S.S. Tyagi,"Dynamic and Real-time Sleep Schedule Protocols for Energy Efficiency in WSNs", International Journal of Computer Network and Information Security(IJCNIS), Vol.8, No.1, pp.9-17, 2016.DOI: 10.5815/ijcnis.2016.01.02
[21] Navid Amini, "Energy-Efficiency of Multihop Ad-hoc Sensor Networks", University of California, Los Angeles, pg6 available at http://web.cs.ucla.edu/~amini/files/publications/Amini_TR3.pdf
[22] P.Periyasamy and Dr.E.Karthikeyan, "Survey of Current Multipath Routing Protocols for Mobile AD Hoc Networks", I.J. Computer Network and Information Security, 2013, 12, 68-79
[23] Himadri Nath Saha, Abhilasha Mandal, and Sinha Abhirup, "Recent trends in the Internet of Things", Computing and Communication Workshop and Conference (CCWC), 2017 IEEE 7th Annual.
[24] Himadri Nath Saha, Debika Bhattacharyya, PK Banerjee, "Energy efficient administrator based secure routing in MANET", Advances in Computer Science, Engineering & Applications, pg 659-672.
[25] Himadri Nath Saha, Saptarshi Banerjee, Arnab Majumdar, Ratul Dey, "Modified Ant Colony Optimization (ACO) based routing protocol for MANET", 6th International Conference and Workshop on Computing and Communication (IEMCON -2015).
[26] Debaditya Choudhury, Debanjana Kar, Katha Roy Biswas, Himadri Nath Saha, "Energy efficient routing in mobile ad-hoc networks", 6th International Conference and Workshop on Computing and Communication (IEMCON -2015).
[27] Jayita Mandal, Himadri Nath Saha, "Modified Ant Colony Based Routing Algorithm in MANET", "International Journal of Computer & Organization trends (IJCOT)" Vol. 3 Issue 10, pg 473-477.

Authors' Profiles

Himadri Nath Saha is a professor in the Department of Computer Science & Engineering at Institute of Engineering & Management, Kolkata, India. He received his Ph.D. from Jadavpur University, Kolkata. His research interests include wireless communication, mobile ad-hoc networks, network security, cryptography, and algorithms. He is a member of IEEE, ACM, and IEI as well as a branch councilor of the IEEE and ACM student branch.

Shashwata Mandal was in the Department of Computer Science & Engineering at Institute of Engineering & Management, Kolkata, India. He is currently an Associate Software Engineer at Nomura Research Institute Financial Technologies Limited. His research interests include robotics, machine learning, and Internet of Things. He is a member of ACM and IEEE.

Soham Banerjee was in the Department of Computer Science & Engineering at Institute of Engineering & Management, Kolkata, India. He is currently the co-founder of the software startup Renderbit Technologies. His research interests include network security, ad-hoc networks, artificial intelligence and machine learning. He is a member of ACM.

Shinjan Mitra was in the Department of Computer Science & Engineering, Institute of Engineering & Management, Kolkata, India. He is currently an independent researcher. His areas of interest include ad-hoc networks, Internet of things, UAV, network security and autonomous robotics. He is also a member of ACM.

Urmi Saha was in the Department of Computer Science & Engineering, Institute of Engineering & Management, Kolkata, India. She is currently an associate developer at Abzooba India Infotech Pvt Ltd. Her areas of interest include autonomous systems, machine learning, data mining, distributed systems, cloud computing.

How to cite this paper: Himadrinath Saha, Shashwata Mandal, Shinjan Mitra, Soham Banerjee, Urmi Saha,"Comparative Performance Analysis between nRF24L01+ and XBEE ZB Module Based Wireless Ad-hoc Networks", International Journal of Computer Network and Information Security(IJCNIS), Vol.9, No.7, pp.36-44, 2017.DOI: 10.5815/ijcnis.2017.07.05

