Proceedings of The National Conference
On Undergraduate Research (NCUR) 2018
University of Central Oklahoma
Edmond, Oklahoma
April 5-7, 2018

# Disruption Tolerant Hybrid Mesh Network Using Raspberry Pi and XBee

Andrew Hughes, Luis Cepeda, Stephanie Lopez, Eric Mendez
San Jacinto College
5800 Uvalde Rd
Houston, TX 77049
Faculty Advisor: Dr. Nathanial Wiggins

## Abstract

A wireless mesh network will become a vital technology for NASA communications. The configuration allows for peer to peer communications where each node is responsible for routing traffic to other nodes in the network. It is decentralized allowing the network to be self-healing, if any one node is unavailable it simply finds the next available node. Mesh networks offer low cost alternatives given the commercial aspect and mature technologies that are readily available. This paper explores the feasibility of mesh networks in a space habitat environment, specifically its utility in a life-support function. Additionally, disruption tolerant software was tested on this network configuration to provide stability and security. The results using a mixed mesh network composed of Raspberry Pi and XBee hardware with fire and gas sensors were successful.

**Keywords: ION-DTN, Raspberry Pi, Ad-hoc Mesh Network, Disruption Tolerance**

## 1. Introduction:

Mesh networking using ZigBee hardware has been around for almost a decade. The most common configurations include a coordinator, XBees connected to either a Raspberry pi or Arduino microcontroller, a router, and sensors to record data. Data gathered by an XBee endpoint and is routed from each node to the coordinator.

There are a variety of uses for mesh networks. These networks have been used to enable community wide communication during natural disasters, are well suited agricultural applications to monitor soil quality, and research is ongoing to develop a mesh network for CubeSats in low earth orbit.

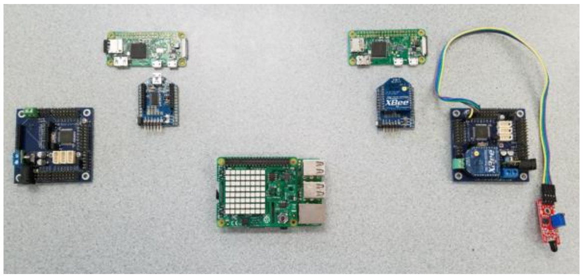

Figure 1: Basic hardware setup of mesh nodes and sensors. Raspberry Pi 3 mesh node in center, Raspberry Pi Zero W mesh nodes at top left and right.

Mesh networks are typically composed of a router, nodes, and an endpoint. The router is configured with OpenWRT and qmp to provide a network for the nodes. Nodes are able to communicate directly with other nodes within range, however this data can be routed through the network to reach nodes beyond range. Our mesh network is completely ad-hoc, forgoing the router and instead each node is a component of the decentralized network topology.

Our team seeks to answer several questions. Can we control a robot using a cellphone connected to a mesh network undergoing the loss of several nodes? Can a space habitat life support system be extended by a mesh network? Are there protocol improvements necessary to minimize routing delays? What other improvements will be necessary to reduce power consumption? What security improvements are available to prevent intrusion?

Our objective is to communicate with life support sensors such as CO2, fire, and humidity, and create a robot that can be used for remote excursion to a hostile surface. The network will display sensor readings and allow for the control of the robot, as well as demonstrate disruption tolerance.

Limitations include issues in routing path recovery using HRP. Data overhead in AODV. Energy consumption is increased when AODV must reroute in the event of the loss of a node. Wireless Mesh Networks suffer from various routing issues, QoS issues, channel allocation, and sustainability of routes.

A disruption tolerant mesh network will contribute information about self-healing mesh networks that can be used in future research. This network can be used to create a solar-system wide network that can route around celestial objects that block line of sight communication between probes and landers with earth or may be used in a space habitat.

## 2. Definitions:

### 2.1 batctl:

Configuration interface for BATMAN-adv with built-in network monitoring tools.

### 2.2 BATMAN-adv:

Mesh routing protocol, this is the software used to set up the mesh network and allow the computers to talk to each other.

### 2.3 ION-DTN:

The Interplanetary Overlay Network is a Delay-Tolerant Networking suite intended for use in embedded systems on spacecraft flight computers.

### 2.4 Raspberry Pi:

A small (the size of a credit card) $35 computer. Comes with Raspbian OS installed.

### 2.5 rc.local:

Text file within the Raspberry Pi OS that runs terminal commands upon system startup.

### 2.6 XBee:

Radio module based on the ZigBee protocol.

### 2. 7 XCTU:

Software used to configure and pair XBee radio modules.

### 2.8 ZigBee:

Communication protocol for low-power, low-bandwidth digital radios used to create personal area networks.

## 3. Literature Review:

Throughput for three different topologies were compared using ZigBee hardware. A tree topology saw the greatest throughput because network load is divided between the router and coordinator, reducing the number of collisions. However, a tree topology is affected by delays due to the parent-child relationship of the network hardware. Though a mesh network is unable to achieve the throughput of a tree network it does not suffer the same delays.[1]

The routine protocol used by XBee mesh networks, AODV, is unsuitable for large networks due to the high number of packet collisions, data loss, and energy expenditure. An improved protocol, AODV-FL, reduced delays in communication, packet and data overhead, and saw improvements in route efficiency and packet overload. Ad-hoc On Demand Distance Vector (AODV) routing improvements were made by adding fuzzy logic to the decision making process which selects a route through the best nodes and introducing a timer that evaluates identical RREQs and forwards the best. Input parameters considered were the number of hops, local battery level, and Received Signal Strength Indicator (RSSI).[2]

ZigBee combines AODV with the Hierarchical Routing Protocol (HRP) to achieve a routing path. HRP provides fragile and inefficient routing paths because of its reliance of parent-child links. In the event one of these links is broken, HRP is unable to recover the routing path. Performance of cluster based hierarchical routing is improved when using Enhanced HRP (EHRP) which uses links to neighbor nodes based on the hierarchical addressing scheme.[3] This in effect reduces the broadcast and memory overhead of AODV because RREQ flooding is no longer required to find the shortest routing path.

Raspberry Pi and XBee mesh network. XBees are connected to a solar-powered battery. The Raspberry Pi is connected to an AC source. XBees were configured by the Raspberry Pi using a breadboard and jumper cables. A mixture of Node.js and Python was used to configure sensors and data was stored into a MongoDB database. The full setup was composed of three Raspberry Pis connected to XBees, a coordinator, one router, and an end device. Each node recorded sensor data to a local database and then broadcast to all nodes within range.[4] The major underlying issue in this setup was insufficient power from the solar panels. No communication errors were observed, however the network was small and contained. Communication issues may arise in larger networks.

A mesh network using peer-to-peer topology removes the possibility of a single point of failure because it does not rely on a central router.[5] This network is dynamically configurable and spacecraft may be added or removed from the network. A time division multiple access (TDMA) communication scheme was used for communication between nodes. In this scheme, time is divided into segments called Frames. Frames are composed of Cycle and Sleep periods. Each Cycle is divided into Slots in which only one node is transmitting and all others are listening.

A key design advantage in this study was the integration of the gateway node, database server, and web server into on Raspberry Pi. The advantage of such a design is the ability to access data via local network or through the internet. This base station connected to an XBee coordinator module. In the network, the coordinator node can query sensor nodes using either multicast or unicast communication modes.[6] The web interface allowed for dynamic display and an interactive user interface.

A Raspberry Pi was compared to popular wireless sensor nodes. Although the Raspberry Pi suffers from higher power consumption, its many advantages include a comparatively powerful ARM processor, much more RAM, and the ability to run independently with its own OS which can be accessed remotely via SSH. It is also possible to form an expandable system with a variety of electrical components using I^2C or SPI digital input/output. Aside from power consumption, the Raspberry Pi lacks a Real Time Clock and cannot boot from USB storage.[7]

The IEEE 802.11s WLAN standard amendment lacks network monitoring and management. Rethfeldt et al. developed a management solution based on SNMP which covers dynamic mesh bootstrapping, error recovery, status monitoring, and remote configuration.[8] Using Optimized Link State Protocol (OLSR) implemented using a Raspberry Pi, Sumarudin et al. saw major QoS improvements in throughput and range and a reduction in packet loss.[9] OLSR is based on Djikstra's Algorithm which obtains the shortest routing path available. Currently, the OLSR is limited to ad-hoc connections.

Wireless Mesh Networks suffer from various routing issues, QoS issues, channel allocation, and sustainability of routes. Currently many problems affect WMN implementation. For nodes to work effectively they must be able to switch frequencies quickly necessitating the selection of a precise radio technique over the physical layer. TCP affects the effective utilization of network resources and is inappropriate for use in WMNs due to routing issues, channel allocation, and sustainability of routes.[10] As the number of nodes increase it is difficult to optimize the channel capacity, degrading communication performance. This could be solved by either increasing channel capacity to offer equal QoS to all connected nodes or by adding new gateway nodes. A cross-layer distributed network monitoring mechanism using reinforcement learning techniques could be used to mitigate TCP's suboptimal resource allocation for nodes within a wireless mesh.[11]

IEEE 802.11s with DSS and Wireless Distribution Systems apply multi-bounce network methods and allow nodes to auto-arrange communication to the router. The protocol has three fundamental parts. A network Portal serves as a passage to different systems, the Network Station acts as a switch, and the Network Access Point enables administration.

IEEE 802.15.4 ZigBee networks are a Wireless Personal Area Network standard with single-bounce and multi-jump capabilities. A coordinator can bolster network topology by designing the system in multi-jump style. Its use is appropriate for Wireless Sensor Mesh Networks.

There are also many benefits to a WMN. The diminished setup time and costs of support due to self-organization and configuration allows administrators to easily change, grow and adjust the system. Remote switches that benefit from multi-hop transmission allow for expansive network coverage with a low deployment cost. WMNs non-dependability of any specific node results in enhanced reliability. In the event of a node failure, communication between working nodes is not lost. This allows for various data exchange mechanisms to minimize bottlenecks and facilitates load balancing in heavy traffic conditions. Hybrid WMNs using several standards may be use to improve interoperability.

## 4. Methodology:

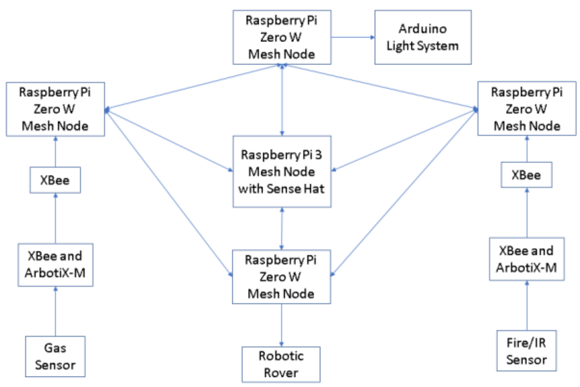


Figure 2: Functional diagram of Mesh Network

Initially a wireless ad-hoc mesh network between a Raspberry Pi 3 B and a Raspberry Pi Zero W was created. Attached to the Raspberry Pi Zero W is an XBee coordinator paired with an Arbotix-M Robocontroller with an integrated XBee end-point. The Raspberry Pi nodes run Raspbian OS. BATMAN-adv and batctl were installed on each node, and shell commands were added to rc.local to enable mesh network configuration at startup. An Arduino program was uploaded to the Arbotix board allowing it to send a simple ACK message upon receiving a SYN packet from the XBee coordinator which was then displayed on the Raspberry Pi Zero W using Node.js.

Installation of BATMAN-adv and batctl:

```
sudo apt-get install batman-adv
sudo apt-get install batctl
sudo modprobe batman-adv
```

Modifications to rc.local to enable mesh network upon boot:

```
sudo ifconfig wlan0 mtu 1528
sudo ifconfig wlan0 down; sudo iwconfig wlan0 mode ad-hoc essid mesh-things-up ap 02:12:34:56:78:90
channel 1
```

```
sudo batctl if add wlan0
sudo ifconfig wlan0 up
sudo ifconfig bat0 up
sudo ifconfig bat0 172.27.0.1/16
```

Each mesh node is then configured with ION-DTN. Our configuration has each mesh node with a single egress point to a host Raspberry Pi which collects data from each sensor on the network and displays on a web page accessible from the internet.

Several sensors are connected to the network, one per mesh node. The temperature/humidity sensor is directly connected to a Raspberry Pi's GPIO pins. Temperature and gas data is collected by a python program which is locally stored in a log file and accessed through another python program. The fire sensor is connected to an Arbotix board with an integrated XBee endpoint which communicates to a Raspberry Pi connected to an XBee coordinator. The Circadian Lighting System is a 16x16 NeoPixel RGB LED Matrix programmed by an Intel Currie board connected to a Raspberry Pi via USB. The robot is based on an Arbotix board with XBee endpoint and controlled through the mesh network via XBee endpoint.

Each pair of XBee radios is configured using XCTU. Careful consideration was made to put each pair of XBees on different frequencies to avoid interference. Data transferred over the XBee network is received through COM software which then stores the data in a local log file. This sensor data is then written to a log file on a host Raspberry Pi which then displays the data in a web page.

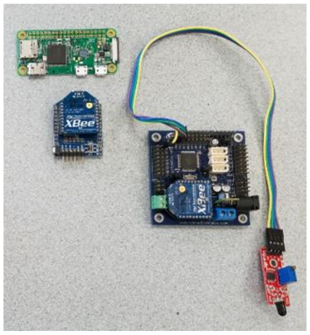

Figure 3: Fire Sensor attached to Arbotix board with XBee and Mesh Node. This is the basic sensor configuration for the four sensor mesh nodes.

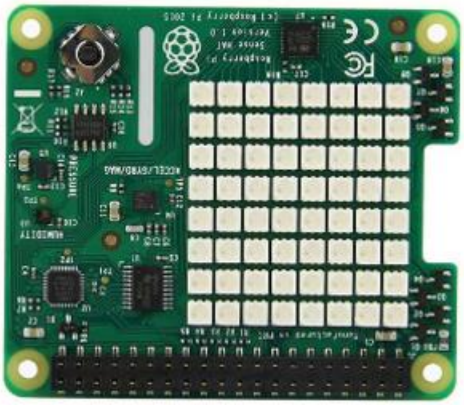

Figure 4: Raspberry Pi Sense Hat for Temperature and Humidity

ArbotiX-M allows the sensors to transmit the information wirelessly through the Arduino XBee. The XBee attaches to the ArbotiX-M. The ArbotiX-M connects to the PC through a UartsBee v4 and uses a DC 9V adapter for power. The ArbotiX board requires six double-ended connector wires to attach to the UartsBee; then attach the UartsBee to the PC. To initially test the responsiveness of the board, run a blink code in the sketches menu of the Arduino IDE and check that the ArbotiX LED light is blinking. If the board responses accordingly, attach the fire sensor. The fire sensor requires four double-ended female connector wires. Connect one end to the fire sensor and the other to the Arbotix board make sure the wires match the corresponding labels on each side. (GND, 5V, AO, DO) The sensor should display a red LED light if properly connected. Then run Fire Code (see Appendix) in a new sketch window. Use a lighter or match to verify the sensor is working properly. If the sensor reads the flame, it will output a much lower value without the flame. This ensures the sensor is working. Once that is confirmed, continue to program the XBee to wirelessly transmit data. To program the XBee, download program XCTU.


Figure 5: FT232RL XBee USB to Serial Adapter V1.2 Board Module for Arduino. Used for programming both the XBee and Arbotix-M boards.

XCTU is a software that allows the XBees to be setup to communicate. (see figure 6) Using these values set up the commander XBee and the endpoint (Robocontroller as shown in figure 1). An XBee with previously used configurations may be set back to default in XCTU by pressing the “default” button. Then set up the XBee (as shown in figure 2) and when set up is complete press the “write” button.

To test configuration of XBee, connect the endpoint XBee to the Arbotix board with a sensor ready to use. The Arbotix board does not have to be connected to the computer. Attach the coordinator XBee to the Uart board, which then connects to the PC via USB for power and reads the data being transmitted in the Arduino IDE serial monitor.

| Parameter | XBee # 1 (For ArbotiX Robocontroller) | XBee # 2 (For ArbotiX Commander) |
|---|---|---|
| Pan ID (Hex) | 100 | 100 |
| MY ID | 1 | 2 |
| DL ID | 2 | 1 |
| Data Rate | 38400 [5] | 38400 [5] |
| Node Identifier | Robot | Commander |

Figure 6: XBee Configuration. XBee #1 configuration used for the endpoint and #2 for the coordinator.

Figure 7: Arbotix-M board without integrated XBee endpoint.

## 5. Data and Analysis:

Disruption tolerance using ION-DTN transmits data only when a connection is established. Should a disruption in communication between mesh nodes occur, the Bundle Protocol prevents data loss. Data is stored temporarily and this stored data is transmitted upon restoration of the network.

The sensors were tested to find a baseline and a threshold that is used to alert the system of a problem in the habitat. The ad-hoc mesh network does not rely on a central router. By a third Raspberry Pi Zero W node to the network removes the possibility of a single point of failure. Our network is cost effective, with a cost of $20 for each Raspberry Pi Zero W node and $35 for the Raspberry Pi 3 node. Initially the nodes were separated by a range of one meter. When that range was expanded to five meters a packet loss of less than 5% was recorded. The maximum theoretical range is one kilometer.

The gas sensor, sensitive to smoke and ethyl alcohol showed a drop in values when an alcohol solution was allowed to evaporate in the sensor's vicinity. The fire sensor also saw a drop in values when a flame was near the receiver.

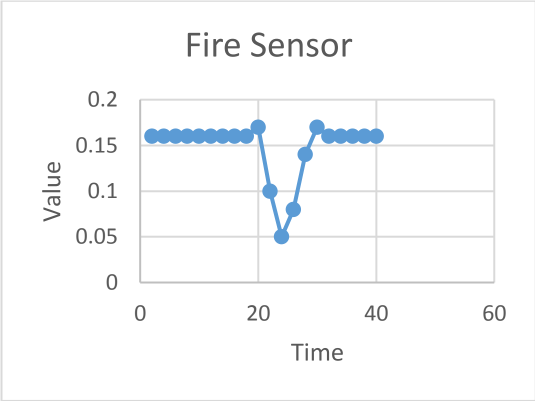


Figure 8: Fire sensor data. Flame placed in range of the sensor at about 20 seconds.

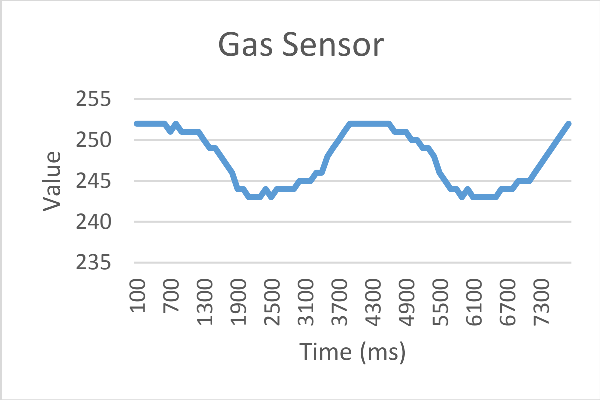


Figure 9: Gas sensor data. Ethyl alcohol wafted over the sensor for a period of 8 seconds. The first pass from .7 seconds to 3.7 seconds, and a second pass from 4.3 seconds to 7.3 seconds.

## 6. Research Conclusions:

ION-DTN improves many aspects of our original ad-hoc mesh network. The software prevents data transmission during a disruption event, and sends the stored data upon reconnection. It also introduces a layer of security through the Bundle Security Protocol which allows for communication in between members of the mesh network.

In the future sensor data could be used for an active life support system. Should a sensor detect an event that exceeds its set threshold it could then activate a device that alleviates the issue (e.g. dehumidifier or fire suppression system). A ground based robot could be replaced by an aerial drone allowing for greater line of sight which would be less likely to be blocked by hills or other planetary features.

## 7. Acknowledgements:

The authors wish to express their appreciation to Professor Nathanial Wiggins, Chatwin Lansdowne, San Jacinto College Honors Program, and the Texas Space Grant Consortium.

## 8. References

1. Ranjan, Rohit, Harsukhpreet Singh, and Geetika Manan. "Comparative Analysis of WSN Topologies using ZigBee/IEEE 802.15. 4 standards." International Journal for Science, Management and Technology (IJSMT) 9, no. 9 (2016), http://www.cfrde.com/pdf/Rohitharsukh.pdf.
2. Ortiz, Antonio M., Fernando Royo, Teresa Olivares, and Luis Orozco-Barbosa. "Intelligent route discovery for ZigBee mesh networks." In World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2011 IEEE International Symposium on a, pp. 1-6. IEEE, 2011, https://www.researchgate.net/profile/Fernando_Royo/publication/221567256_Intelligent_route_discovery_for_ZigBee_mesh_networks/links/00b7d5335a8e0eeef2000000.pdf.
3. Ha, Jae Yeol, Hong Seong Park, Sunghyun Choi, and Wook Hyun Kwon. "EHRP: Enhanced hierarchical routing protocol for zigbee mesh networks ee." IEEE communications letters 11, no. 12 (2007), https://www.researchgate.net/profile/Sunghyun_Choi2/publication/3418240_EHRP_Enhanced_hierarchical_routing_protocol_for_ZigBee_mesh_networks/links/53f305530cf2bc0c40ecded2/EHRP-Enhanced-hierarchical-routing-protocol-for-ZigBee-mesh-networks.pdf.
4. Forcella, Michael. "Creating a mesh sensor network using Raspberry Pi and XBee radio modules." PhD diss., 2017, https://dspace.sunyconnect.suny.edu/bitstream/handle/1951/69280/Forcella_Thesis.pdf?sequence=1.
5. Becker, Christopher, and Garrick Merrill. "Mesh Network Architecture for Enabling Inter-Spacecraft Communication." (2017), https://digitalcommons.usu.edu/cgi/viewcontent.cgi?article=3804&context=smallsat.

6. Ferdoush, Sheikh, and Xinrong Li. "Wireless sensor network system design using Raspberry Pi and Arduino for environmental monitoring applications." Procedia Computer Science 34 (2014): 103-110, https://www.sciencedirect.com/science/article/pii/S1877050914009144/pdf?md5=865de2d11431f616a88eb473e4ce81e3&pid=1-s2.0-S1877050914009144-main.pdf&_valck=1.

7. Vujovic, Vladimir, and Mirjana Maksimovic. "Raspberry Pi as a Wireless Sensor node: Performances and constraints." In Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2014 37th International Convention on, pp. 1013-1018. IEEE, 2014, https://www.researchgate.net/profile/Vladimir_Vujovic/publication/269291637_Raspberry_Pi_as_a_Wireless_Sensor_node_Performances_and_constraints/links/557953db08aeacff2002966b.pdf.

8. Rethfeldt, Michael, Peter Danielis, Guido Moritz, Bjorn Konieczek, and Dirk Timmermann. "Design and Development of a Management Solution for Wireless Mesh Networks based on IEEE 802.11 s." In Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on, pp. 902-905. IEEE, 2015, https://pdfs.semanticscholar.org/d73a/b9528dc4addc5d364ef1e5c98cca9e3fa8a8.pdf.

9. Sumarudin, A., and Trio Adiono. "The design of high throughput Wi-Fi mesh networked wireless sensor network using OLSR protocol." In Automation, Cognitive Science, Optics, Micro Electro-Mechanical System, and Information Technology (ICACOMIT), 2015 International Conference on, pp. 192-195. IEEE, 2015, https://www.researchgate.net/profile/Ahmad_Sumarudin/publication/304412660_The_design_of_high_throughput_Wi-Fi_mesh_networked_wireless_sensor_network_using_OLSR_protocol/links/585b8d0308ae6eb8719c0585/The-design-of-high-throughput-Wi-Fi-mesh-networked-wireless-sensor-network-using-OLSR-protocol.pdf.

10. Naveen, T. H., and G. Vasanth. "Qualitative Study of Existing Research Techniques on Wireless Mesh Network." INTERNATIONAL JOURNAL OF ADVANCED COMPUTER SCIENCE AND APPLICATIONS 8, no. 3 (2017): 49-57, https://pdfs.semanticscholar.org/0a16/acfd3b7bc26807c2bd613a543b90a17e63d7.pdf.

11. Arianpoo, Nasim, and Victor CM Leung. "How network monitoring and reinforcement learning can improve tcp fairness in wireless multi-hop networks." EURASIP Journal on Wireless Communications and Networking 2016, no. 1 (2016): 278, https://link.springer.com/article/10.1186/s13638-016-0773-3.