Wireless Sensor Network Communication Protocols

by

Weiqi Zhang and Bradford G. Nickerson

TR11-208, May 25, 2011

Faculty of Computer Science
University of New Brunswick
Fredericton, N.B. E3B 5A3
Canada

Phone: (506) 453-4566
Fax: (506) 453-3566
E-mail: fcs@unb.ca
http://www.cs.unb.ca

Introduction

CS6999 Wireless Sensor Network (WSN) Communication Protocols aims at understanding how WSN communication protocols work. We focus on the data link, network and transport layer of the Open Systems Interconnection (OSI) model. Medium Access Control (MAC) protocols provide addressing and channel access control mechanisms that make it possible for several terminals or network nodes to communicate within a multi-point network. In this survey, the assumption is that nodes in the network are fixed in one location, but the radio communication space is highly dynamic due to interference from the environment or from competing trafic in the same (or nearby) frequency bands.

1 A Survey For Energy Conservation In Wireless Sensor Networks

The wireless sensor network architecture talked about in Anastasi et al. [2009] is illustrated in Figure 1: one sink node and a number of sensor nodes are deployed over a large geographical area of diameter equal to 10 times the transmission range of the node transceiver. The transmission range of a sensor node is typically less than 150 meters. Data are transferred from sensor nodes to the sink node through a multihop communication protocol. The sink is also called a gateway.

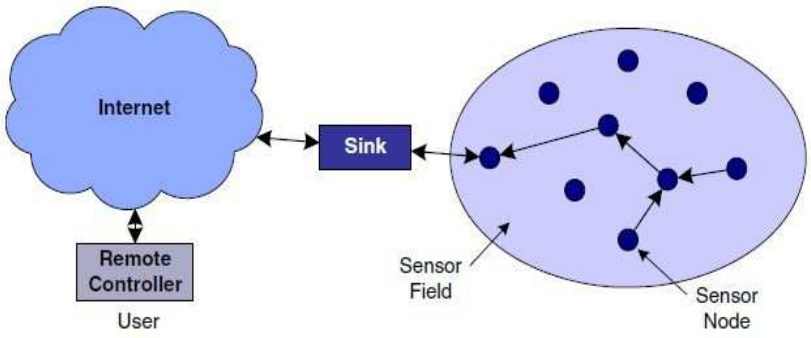


Figure 1: Sensor network architecture, from [2].

A typical wireless sensor node architecture consists of power supply subsystem, sensing subsystem, processing subsystem, and communication subsystem. Generally, data transmission is very expensive in terms of energy consumption(e.g. 1 nanojoule per bit transmitted), while data processing consumes significantly less energy. Typically, processing and sensing subsystems are assumed to use much less energy than the communication

subsystem.

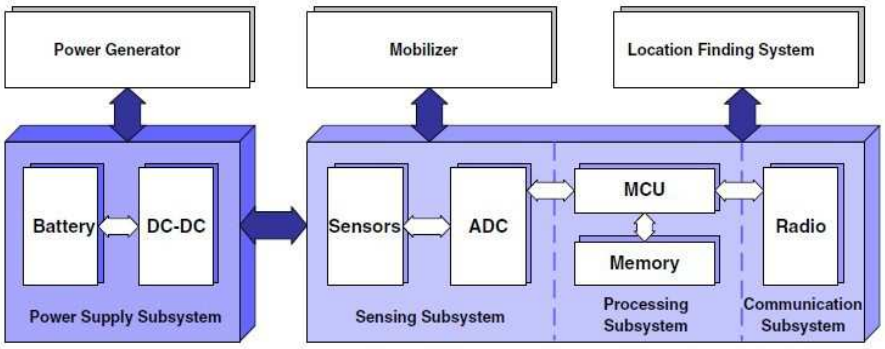


Figure 2: Sensor node architecture, from [2].

As discussed in Anastasi's paper at [2], three main enabling techniques are used for energy conservation, they include duty cycling, data-driven approaches, and mobility.

The basic idea of duty cycling is to eliminate the network redundancy to prolong the network longevity (Topology Control), and set the radio to sleep mode whenever possible (Power Management). The focus for this technique is to minimize the number of nodes as well as the number of active nodes. In Topology Control, we prefer to use Connectivity-driven protocols because Location- driven protocols need an expensive GPS system. Adaptive Self-Configuring sensor Network Topologies (ASCENT) is a connectivity-driven protocol. In ASCENT a node decides whether to join the network or continue to sleep based on information about connectivity and packet loss that are measured locally by the node itself [2]. Only some nodes are initially active while all others are passive, they only listen to packets but do not transmit them. The sink node may suffer a serious packets loss if the number of active nodes is not large enough. At that time, the sink node will send help messages to request passive neighbouring nodes to join the network by changing their states from passive to active.

The basic idea behind data-driven approaches is to use a data model to predict data instead of the actually sensed data (data prediction). Sometimes the sensing subsystem consumes more energy than the communication subsystem or even more than the rest of the sersor node. So evergy conservation schemes try to reduce the number of acquisitions (Energy-efficient Data Acquisition). Data-driven approaches focus mainly on how to reduce unneeded samples.

Another useful way for reducing energy consumption is mobility. The nodes' density in the network has to be large enough to ensure successful communication between two nodes. Mobility has been considered as an alternative solution for energy-efficient data collection

in wireless sensor networks. For example, sensors can be equipped with mobilizers for changing their location [2]. A mobilizer itself is always expensive, and a mobilizer might also be more energy consuming than the node itself. Considering the cost, sensor nodes can be placed on vehicle or animals which are mobile instead of providing mobilizers. An example of this kind is Zebranet, a system for wildlife tracking that focuses on monitoring zebras [7].

2 Low Power Media Access For Wireless Sensor Networks

The primary concern of MAC protocol energy efficiency is to reduce energy consumption by selecting a duty cycle that permits the radio transceiver to be switched off as long as possible. In wireless sensor network deployment, reliably reporting data while consuming the least amount of energy is the ultimate goal [10]. This section talks about general ideas of two slotted access protocols S-MAC and T-MAC, and a sampling, asynchronous protocol called B-MAC.

2.1 Data Link Controls

S-MAC [15] is Sensor-Medium Access Control protocol which is designed for wireless sensor networks. S-MAC is in the class of slotted access protocols. Slotted access protocols require nodes to synchronize on a global notion of time, which is then organized as a sequence of slots [8] (illustrated in Figure 3). S-MAC periodically sleeps, wakes up, listens to the channel, and then returns to sleep. Each active period is of fixed size, 115ms, with a variable sleep period [10]. The length of the sleep interval indicates the duty cycle of S-MAC. Nodes process synchronization at the beginning of the active interval. S-MAC was updated with an adaptive listening capability that requires node to maintain their neighbour's schedules. Adaptive S-MAC has to maintain more neighbours' schedules or incurs additional repeated rounds of resynchronization when the density of the network grows.

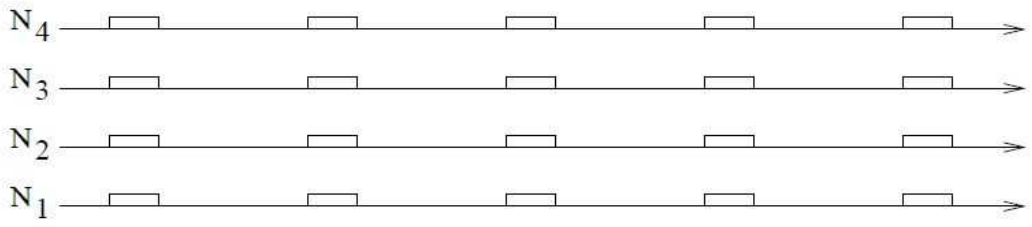


Figure 3: A synchronous slotted protocol, from [3].

T-MAC [12] is Timeout-Medium Access Control protocol which improves on S-MAC's energy usage by using a very short listening window at the beginning of each active period. After the SYNC section of the active period, there is a short window to send or receive RTS (Request To Send) or CTS (Clear To Send) packets [10]. The node will return to sleep mode if there is no activity detected during the active period. Although it outperforms than S-MAC on energy usage (consumes less energy), it still suffers the same complexity as S-MAC.

2.2 Design And Implementation Of B-MAC

To achieve the goals of effective collision avoidance, reliability and low power listening, a CSMA (Carrier Sense Multiple Access) protocol for wireless sensor networks called B-MAC, Berkeley-Medium Access Control [10] was designed. B-MAC is one of the most popular contention based MAC protocols [2].

B-MAC uses clear channel assessment (CCA) and packet backoffs for channel arbitration. B-MAC samples the channel before transmission. If the sample is below the current noise floor, the channel is considered clear, and packets are sent immediately. If five samples are taken to measure the channel energy and no outlier is found, the channel is considered busy, and a random backoff is invoked [10]. When the channel is known to be clear, the noise floor will be updated. B-MAC transmits a long preamble before sending data to a target node. The preamble must be at least as long as the check interval of target nodes so that the target node can detect that communication is requested, Figure 4 illustrates a low power listening model in a sampling MAC protocol.

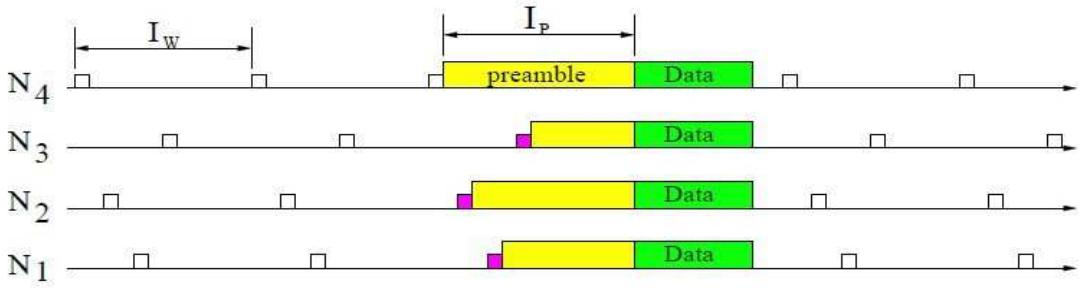


Figure 4: Illustration of a sampling MAC protocol, from [3]. I_w = check interval, I_p = preamble.

B-MAC provides optional link layer acknowledgment support. If acknowledgments are enabled, B-MAC immediately transfers an acknowledgment after receiving a packet.

B-MAC cycles the radio through periodic channel sampling, called Low Power Listening (LPL). Each time the node wakes up, it turns on the radio and checks for activity. If

there is activity detected, the node powers up and stays awake for the time required to receive the incoming packet. After reception, the node returns to sleep. If no activity is detected, a timeout fources the node back to sleep [10].

Each node executes a single application in wireless sensor networks. The application is implemented in a micro controller, since the RAM and ROM size of micro controller are limited, making the code size of implementation as small as possible is very important. Table 1 shows a comparison of the size of B-MAC and S-MAC in bytes. Both protocols are implemented in TinyOS.

Protocol	**ROM**	**RAM**
B-MAC	3046	166
B-MAC w/ ACK	3340	168
B-MAC w/ LPL	4092	170
B-MAC w/ LPL & ACK	4386	172
B-MAC w/ LPL & ACK + RTS-CTS	4616	277
S-MAC	6274	516

Table 1: A comparison of the size of B-MAC and S-MAC in bytes, from [10].

To evaluate the effect of increasing the latency to reduce power consumption, the authors fixed the throughput to one 100 byte packet per 10 second interval. They measured the end-to-end latency of the 10 hop network and varied the sleep period of S-MAC. The results are shown in Figure 5. The points are the optimal tradeoff of latency and energy consumption. B-MAC has a lower latency for the same power, up to the bound of 6 seconds.

3 Analyzing MAC Protocols For Low Data-Rate Applications

The fundamental need for energy-efficient operation has been a driving force behind the development of many WSN-specific MAC protocols [8]. Langendoen and Meier [2010] have taken an analytical approach to answering the question "which protocol is best?" given a set of external conditions including radio hardware characteristics, network topology, and workload. To keep the analysis tractable, Langendoen and Meier did not model MAC-level retransmissions, but included specific boundary conditions safeguarding the contention-free (messages would not collide, no transmission occur) operation of each protocol.

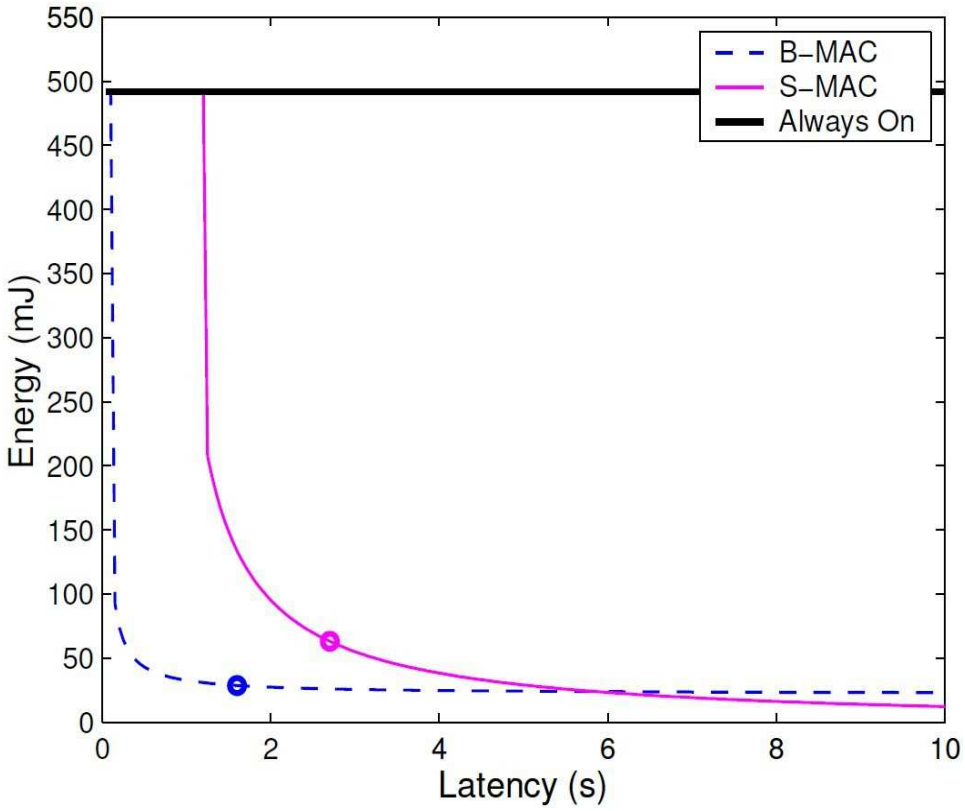


Figure 5: As the latency increases, the energy consumed by both S-MAC and B-MAC decreases, from [10].

3.1 Traffic Model

A spanning tree in the network was constructed that is based on shortest-hop routing to the sink located in the center (see Figure 6). Assuming a uniform node density on the plane and a unit disk graph communication model, there are $C + 1$ nodes in the unit disk (disk containing nodes with $d = 0$ and $d = 1$). Hence, all nodes are in communication range with a fixed number of C neighbours. The nodes are grouped into rings according to their distance d (minimal hop count) to the sink ($d = 0$). The first ring contains C nodes, from which we can derive the node density, and subsequently the number of nodes N_d in ring d. If $d = 0$, $N_d = 1$, otherwise, $N_d = Cd^2 - C(d-1)^2 = (2d-1)C$ [8].

3.2 Analysis

3.2.1 Data Load vs. Energy Consumption

In this experiment they keep the network size (and topology) fixed and vary the sampling frequency F_s (#packets/node/min) at which messages are injected into the network. The aggregate rate F_I^{Sink} of the incoming traffic at the sink is reported in Hz (# of messages

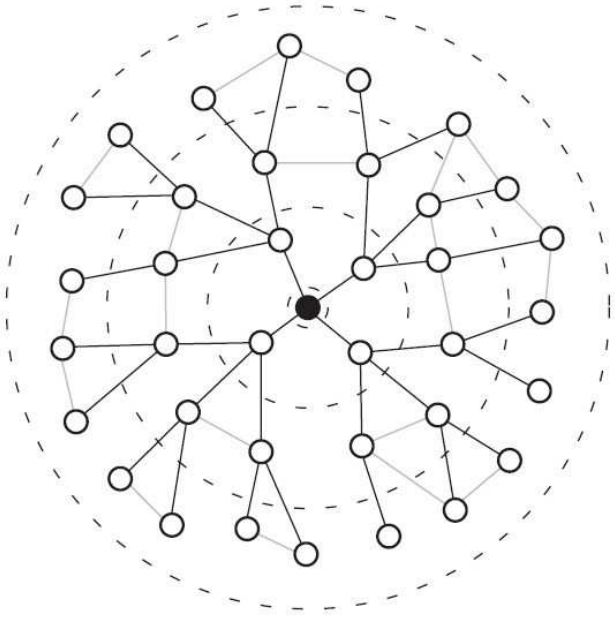


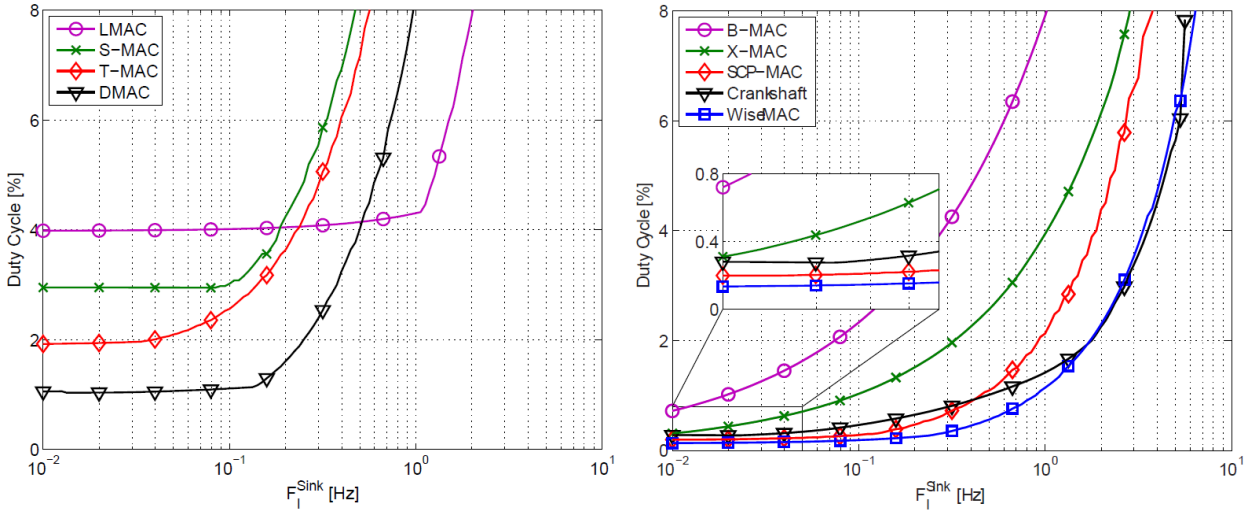
Figure 6: Sample spanning tree with the sink at level 0 and a depth of 3 and $C = 4$ from [8].

received per second). The network is structured as a set of four rings (D=4) with a uniform density of eight neighbours per node (C=8), resulting in a network size of 108 nodes. The popular CC1000 radio was used and external interference was not taken into account.

As we can see from figure 7, the slot-based protocols all have a very high offset for very low data rates, that is, a lot of energy is consumed even when almost no data is communicated through the network because a node must listen in all slots in addition to its own to check for incoming data [8]. A consequence of this "hot" idle mode is that a certain traffic load can be accommodated for free as indicated by the initial flatness of the curves. The CP-based (Channel polling-based) protocols consume significantly less energy in idle mode since the nodes only perform short carrier sensing and do not have to listen into long slots [8]. WiseMAC (Wireless Sensor MAC) [4] exhibits the best energy efficiency for very low data rates, followed closely by Crankshaft (a hybrid MAC that combines scheduled with conention-based access) [5].

3.2.2 Energy Consumption vs. Latency

Figure 8 shows the fundamental trade-off between average per-hop latency and energy consumption (duty cycle) for a six-hop event message (that is, the message has six hops from the source to the destination) injected into an idle network. WiseMAC stands out with its superior energy-latency trade-off. The reason is two-fold: Firstly, WiseMAC was already



(a) Slot-based protocols (b) CP-based protocols

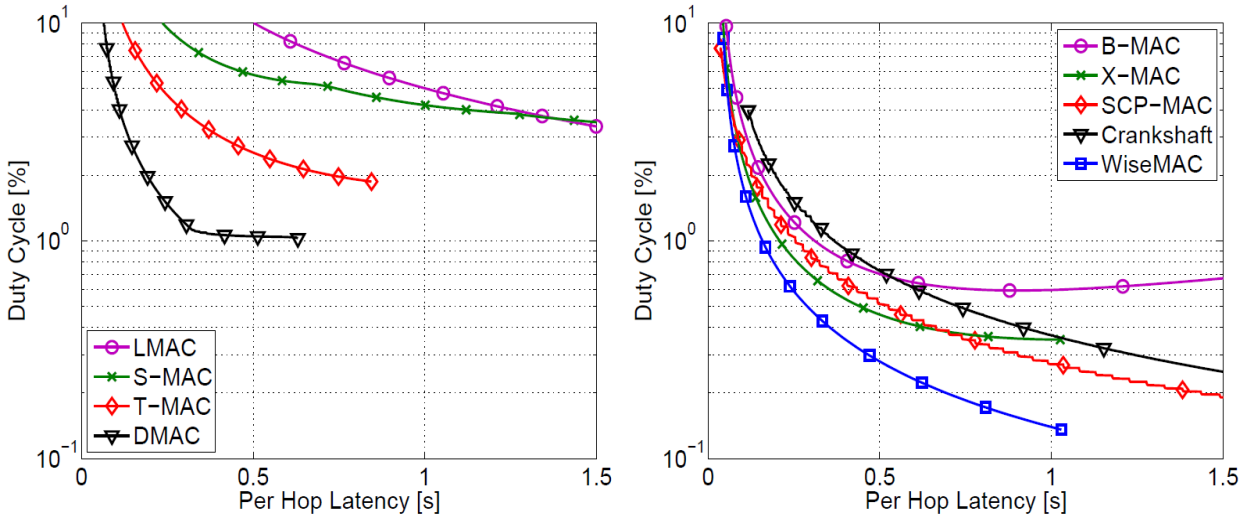
Figure 7: Data Load vs. Energy Consumption, from [8].

shown to operate in a very energy efficient manner for low data rates in the previous section. Secondly, due to the random access times of the nodes, the average waiting time for the parent to wake up is $T_w/2$ [8]. This is in contrast to SCP-MAC (Scheduled-Channel-Polling MAC) [16], which also operates very energy efficiently for very low data-rates, yet delays the message by T_w at every hop. For SCP-MAC, a message is generated somewhere during the wake-up interval before the first contention window, which results in average delay of $T_w/2$ at the first hop. For every additional hop, the packet will be delayed for another T_w.

Although announcing an absolute winner is impossible due to different details of the application requirement and hardware characteristics involved, Langendoen et al. [2010] did observe that the WiseMAC protocol showed a remarkably consistent behavior across a wide range of operational conditions, always achieving the best, or second-best performance.

4 Reliable Multihop Routing In Sensor Networks

Reliable and self-organizing multihop network is challenged by the dynamic and lossy nature of wireless communication. In this section, we study and evaluate link estimator, neighbourhood table management, and reliable routing protocol techniques. We focus on a many-to-one (node can collect data from many neighbours) data collection workload.



(a) Slot-based protocols (b) CP-based protocols

Figure 8: Energy Consumption vs. Latency, from [8].

4.1 Link Estimation

Individual nodes estimate link quality by observing packet success and loss events at the data link layer. Routing level protocols use these estimations to build routing structures. It was found that window mean with exponentially weighted moving average (WMEWMA) performs best over all estimators [14]. The WMEWMA algorithm works as follows. Let l be the number of packet losses we feed into the estimator, and k be a guess on the number of missed packets based on message rate. For every T event, $l = k$, for every M event, $l = max(m - k, 0)$. The tuning parameters are t and α. Let t be the time window represented in number of message opportunities between two T events, and $\alpha \in [0, 1]$. $\hat{P}$ is the current estimation of reception probability. $\hat{P}$ is only updated at each T event. In the time window t between two T events, let r be the number of received messages (i.e. number of 1's in M events), and f be the sum of all losses. The mean $\mu = \frac{r}{r+f}$, and $\hat{P} = \alpha\hat{P} + \mu(1 - \alpha)$ [13].

4.2 Neighbourhood Table Management

Neighbour discovery is performed by a sensor node, when a sensor node receives packets, the node records information about nodes from which it receives packets. When density of network become larger, the number of nodes from which a node recieves packets may be largers than the number of entries it can keep in the neighbour table. The problem is that when the neighbour table is full and a node is not in the table, there is no chance for it build up its link quality and become a neighbour. There are three essential components in neighbour management: insertion, reinforcement, and eviction. The source node is considered to be inserted or reinforced for each incoming packet upon which neighbour management is

performed. If there is one entry of the source node in the table, a reinforcement operation may be performed to keep it there. If there is no entry of the source node in the table, the node has to decide whether to evict another node from the neighbour table or discard the information associated with the source node. Finding a neighbour management algorithm that can keep as many as possible good neighbours in the table regardless of network density is the goal. The focus is on passive neighbourhood discovery, where nodes snoop on periodic data messages. Insertion operation is performed when there are spaces for new entries in neighbour table, eviction operation is performed only when there are no spaces for new entries. FREQUENCY algorithm is an effective and simple algorithm that can maintain a subset of good neighbours over a fixed-size neighbour table. Figure 9 shows performance of different algorithms, we can see that FREQUENCY algorithm outperforms others, it can maintain most good neighbours in the table. On insertion, a node is reinforced by incrementing its count. A new node will be inserted in the table if there is an entry with a count of zero; otherwise, the count of all entries is decremented by one and the new candidate is dropped [14].

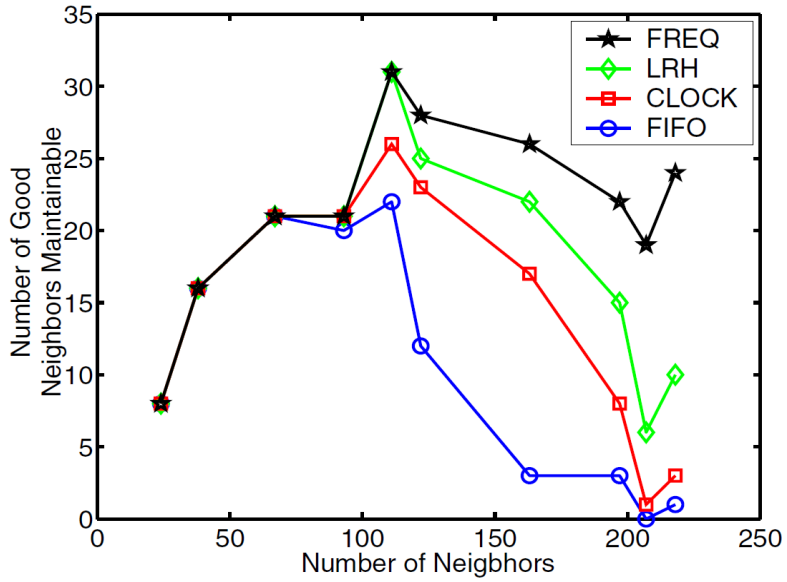


Figure 9: Number of good neighbours maintainable at different densities with a table size of 40 entries, from [14].

4.3 Routing Protocols Evaluation

Shortest Path (SP and SP(t)) protocols are conventional approaches where each node picks a minimum hop-count neighbour and sets its hop-count to one greater than its parent. For SP a node is a neighbour if a packet is received from it. For SP(t) a node is a neighbour

if its link quality exceeds threshold t [14].

Minimum Transmission (MT) protocol assumes that the best path is the one that minimizes the total number of transmissions (including retransmissions) in delivering a packet over potentially multiple hops to the destination [14]. MT protocol uses the expected number of transmissions as its cost metric. In considering the expected number of transmissions of a link, it is important to determine link quality for both directions since losing an acknowledgment would also trigger a useless retransmission. For each link the MT cost is estimated by the product $(1/Q_f)(1/Q_b)$ [14], where Q_f if the forward link quality, and Q_b is the backward link quality. For MT, if additionally consider the effect of using the FREQUENCY algorithm to manage a neighbour table of only twenty entries, it is called MTTM.

The first method of protocol evaluation is graph analysis. Given a static connectivity graph with probabilistic link qualities of all edges derived from inter-node distance, optimal trees are computed for each routing algorithm based on different cost metrics [14]. The SP protocol was eliminated from further consideration due to its poor path reliability and SP yields a very shallow network (narrow hop distribution).

The second simulation used packets to capture the effect of collisions, an empirical study was performed using three nodes at a time (a sender, a receiver, and a collider that also transmits). The MT protocol using the FREQUENCY algorithm to manage a neighbour table (MTTM) yields the best stability (see Figure 10).

4.4 Experimental Results

The test-bed in Woo et al. [2003] is a 50-node network placed as a 5x10 grid with 8 foot spacing using Mica2 style mote with TinyOS used 916 MHz transceivers. As shown in Figure 11, we can see that MT routing with WMEWMA ($t = 30$, $\alpha = 0.5$, neighbour table size = 30) has the best end-to-end success rate. MT delivers roughly 80% of the originated data consistently, the SP(40%) is lower. To further test the robustness of MT, MT is examined under a high enough load to cause substantial congestion in the network. At 3 times the data origination and route update rate, the success rate drops to roughly 50%.

The Woo et al. [2003] study has shown that link quality estimation and neighbourhood management are essential to and tightly coupled with reliable routing in sensor networks. WMEWMA is a simple, memory efficient link estimator that reacts quickly, yet is stable enough for path characterization in cost-based routing. The FREQUENCY algorithm performs well in maintaining a subset of good neighbours in a constrained neighbour table regardless of cell density. Minimum expected transmissions is an effective metric for cost-based routing [14].

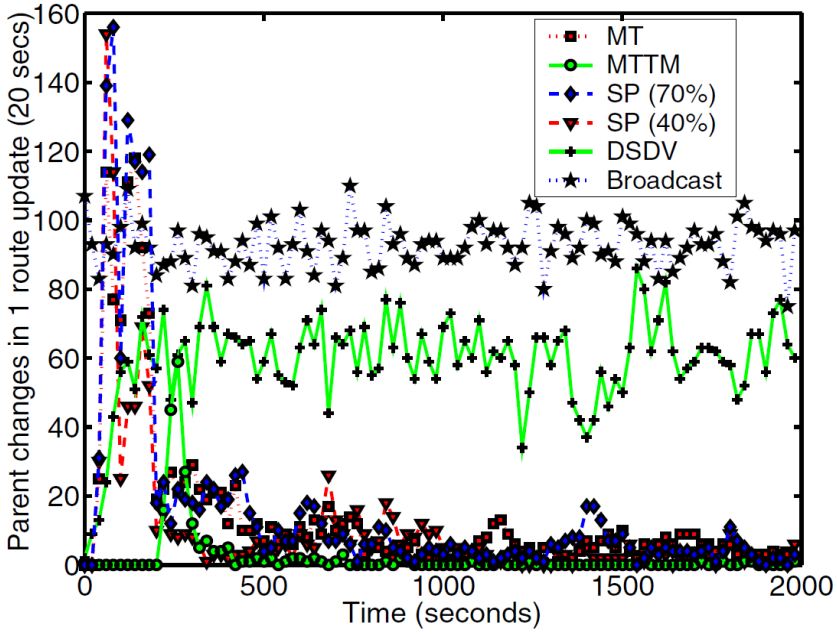


Figure 10: Stability from simulations, from [14].

5 IPv6 In Low-Power Wireless Networks

The IPv6 protocol has recently been adapted for use with low power wireless sensor networks [11] to provide low memory footprint, high reliability, and low energy use in embedded applications.

5.1 Internet Protocol Version 6

Internet Protocol Version 6 (IPv6) is the designated successor of IPv4 as the network protocol for the Internet. 32-bit IPv4 addresses cannot represent all the hosts since the number of hosts on the Internet grows vastly nowadays. To overcome this, IPv6 expands the IP address space from 32 to 128 bits. The IPv6 header is shown in Figure 12. IPv6 packets can vary in size, but are always at least 1280 bytes in length, including the header, and no more than 64 kBytes in length. Figure 13 shows an example IPv6 packet.

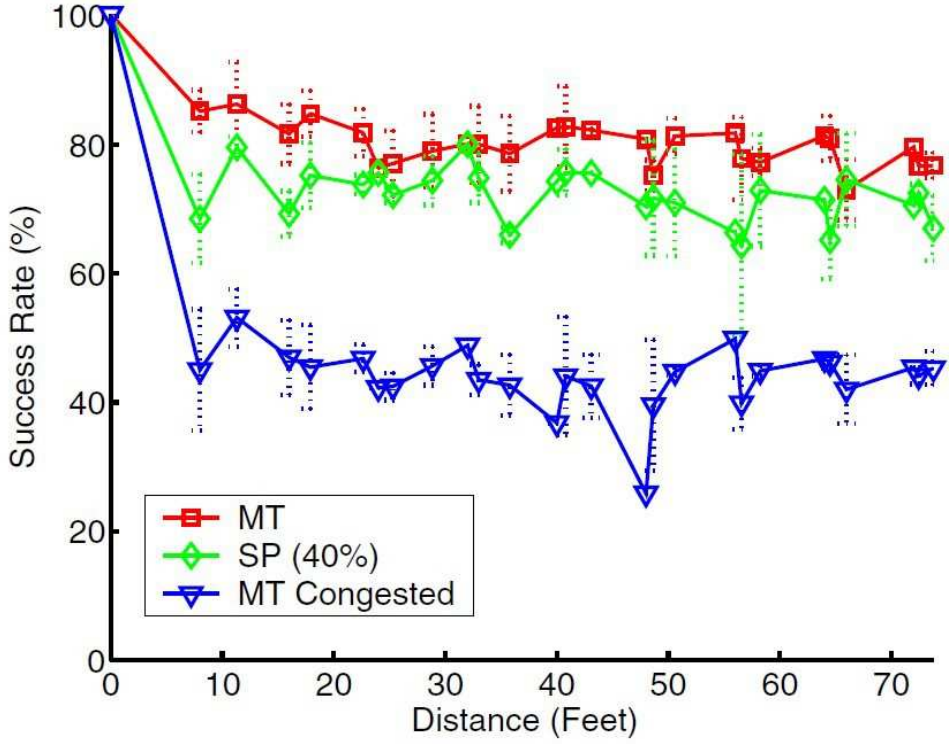


Figure 11: End-to-end success rate over distance in the foyer, from [14].

5.2 IPv6 Neighbour Discovery

IPv6 neighbour Detection (ND) uses a neighbour cache to maintain neighbour information such as the link-layer address mapping, reachability information, and whether the neighbour is a router or a host. As with any cache, its utility depends on the difference between cache hits and misses. With IPv6 ND, for example, always incurring a cache miss would require an address resolution exchange before sending any unicast transmission [11]. Instead, IPv6 uses a neighbour table to maintain neighbour information. Nodes can only communicate with neighbours resident in the neighbour table. Limited memory bounds the number of neighbours a node can communicate with, so the insertion/eviction policies are left to the routing protocol [14].

5.3 IPv6 Forwarding

The forwarder is responsible for enqueuing incoming datagrams, determining the next hop and dequeuing datagrams. Since traditional IP forwarders assume that the link layer delivers datagrams with high success rates, they do not perform hop-by-hop neibour discovery. Furthermore, traditional IP forwarders readily drop datagrams when queues are congested to improve queuing fairness between different flows and the overall responsiveness of the

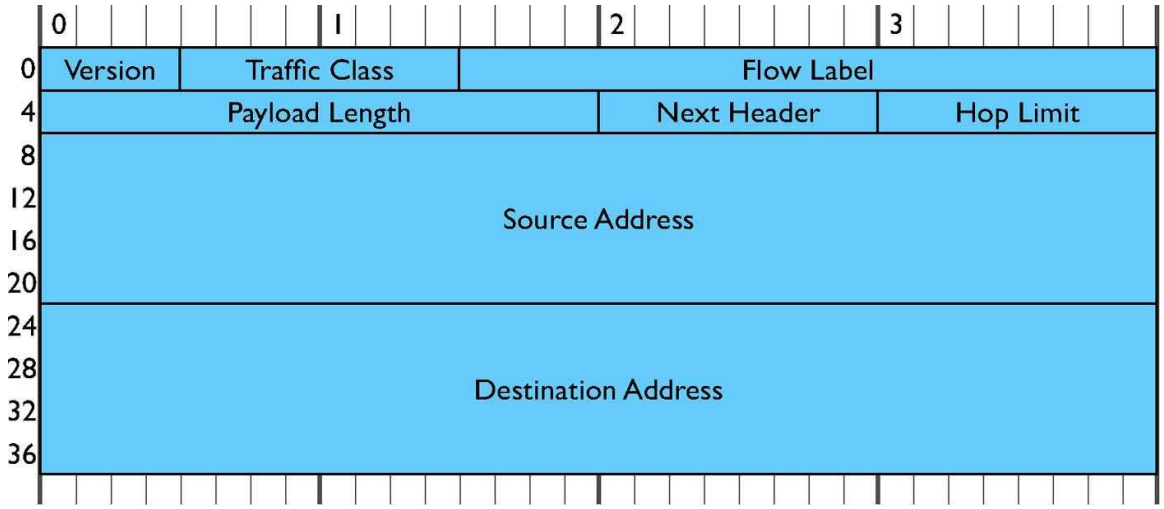


Figure 12: IPv6 header, from [11].

network [11]. However, the low-power wireless links is unreliable and the memory of sensor nodes is extremely limited, the traditional assumptions are not valid any more. Limited memory may constrain forwarding queues to hold only a few messages, full queues is a relatively common occurrence in this situation. Dropping packets freely will vastly reduce end-to-end successful delivery rates and energy efficiency. The IP forwarder must implement hop-by-hop recovery due to the memory constrains and the unreliable nature of low-power wireless environment. The forwarder dequeues messages only on positive indication that it was successfully received at the network layer by next hop. The two most common reasons for delivery failures are 1) link transmission failures, and 2) queue congestion at the receiver [11]. When link transmission failures occur, the forwarder performs another next-hop lookup and resubmits the datagram to the link layer. When queuing failures occur, the forwarder performs congestion control by slowing the forwarding rate to the same next hop [11].

IP protocols generally assume that the link is always-on. Such links can deliver datagrams to neighbouring nodes with relatively low latency. Traditional IP links are actually always-on by constantly listening for packets, while low-power operations create the illusion that a receiver is always-on although it is actually off more than 99% of the time by utilizing duty-cycling techniques (e.g. sampled listening). The tradeoff is decreased communication throughput and increased communication latency [11].

5.4 Evaluation

The authors implemented a production-quality IPv6 network stack for sensornets. The implementation is built using TinyOS 2.x for an Epic-based platform that consists of a

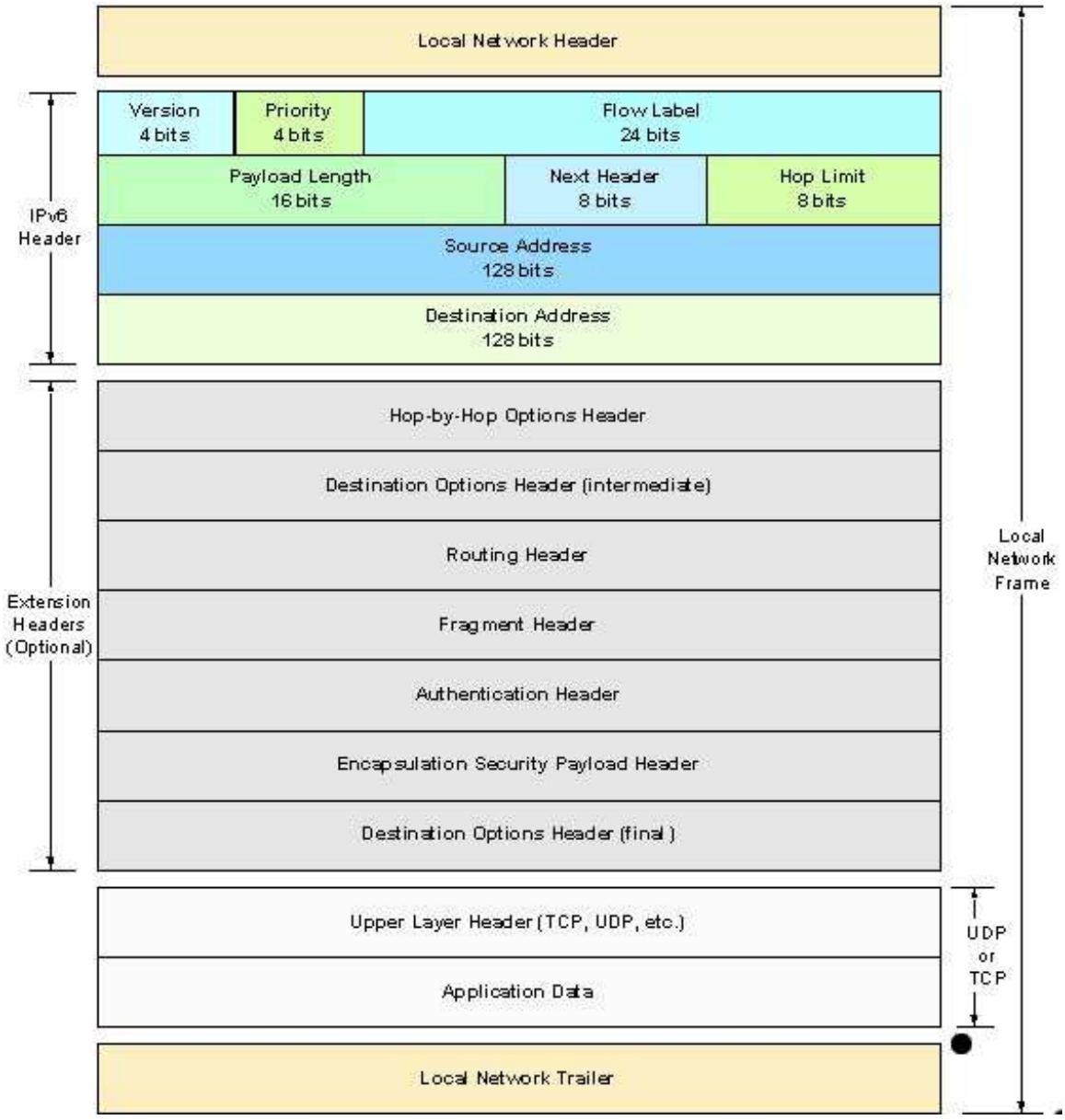


Figure 13: An example IPv6 packet, from [1].

16-bit TI MSP430 MCU with 48-kB ROM and 10-kB RAM and a 2.4-GHz, 250kb/s TI CC2420 IEEE 802.15.4 radio. An embedded kernel that supports one UDP socket and one TCP connection consumes 24038 bytes of ROM and 3598 bytes of RAM. The kernel includes OS-level services required to support the IPv6 network stack. The memory usage of communication components is shown in Table 2.

Utilizing this IPv6 network stack to support commercial environmental monitoring applications in a variety of deployment environments. The characteristics of representative

Component	ROM	RAM
UDP	352	6
TCP	1996	64
DHCPv6 Client	544	8
DHCPv6 Proxy	172	0
ICMPv6	688	0
IPv6 Neighbor Discovery	820	65
IPv6 Forwarder	2226	2048
IPv6 Router	1790	124
6LoWPAN Fragmentation	1136	22
6LoWPAN Compression	1626	18
Link Interface	654	0
CC2420 Driver	4104	390

Table 2: Memory requirements for communication components, from [11].

deployments are summarized in table 3. Deployment in Table 3 indicates the test places, Period in Table 3 indicates the period between reports for each node, depth in Table 3 represents the number of hops to the nearest edge router for paths chosen by the routing protocol. In all deployments, the link layer was configured with a channel sample period of 250ms. Over a 12-month period, all deployments achieved a delivery success rate well over 99% and average duty cycle well below 1%.

Deployment	Dimensions (m) l x w x h	Edge Routers	Sensor Nodes	Period (m)	Depth (hops) mean	Depth (hops) max	Success Rate	Duty Cycle
Baltimore Housing	90 x 50 x 30	7	220	15	2.19	5	99.85%	0.41%
West Virginia Housing	80 x 40 x 15	4	75	15	2.67	5	99.93%	0.37%
Commercial Office	200 x 300 x 5	1	51	1	2.11	6	99.97%	0.38%
Datacenter	30 x 20 x 4	5	211	1	1.2	3	100%	0.40%

Table 3: Production Deployments of IPv6 on 16-bit sensor nodes, from [11].

6 Compression Format For IPv6 Datagrams In Low Power And Lossy Networks

6.1 Fragmentation

6LoWPAN allows IPv6 packets to be sent to or received from IEEE 802.15.4 based networks. The IEEE 802.15.4 frame size is 127 bytes. Figure 14 shows an IEEE 802.15.4 frame. As we can see, there is space for only 31 bytes of data. The size of an IPv6 frame is at least 1280 bytes. How can we fit an IPv6 frame into an IEEE 802.15.4 frame? The solution is fragmentation. The data is first encapsulated by an IPv6 header, and then encapsulated by an Adapt header, which contains fragment information, and lastly by a MAC header. The encapsulation is shown in Figure 15 for UDP packets. TCP packets have a longer header (20, 24, or 28 bytes). Figure 16 shows the first fragment of the Adapt header. Figure 17 shows a noninitial fragment of the Adapt header.

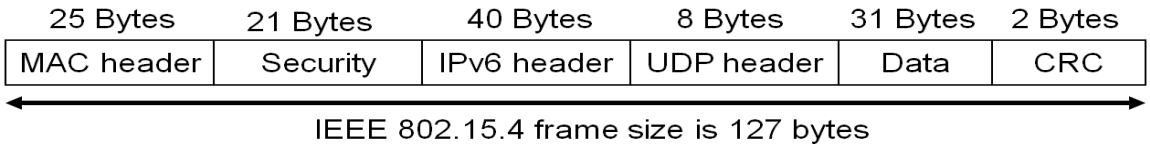


Figure 14: IEEE 802.15.4 frame, from [17].

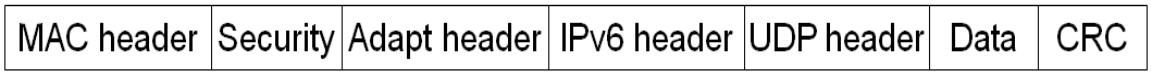


Figure 15: Data encapsulation.

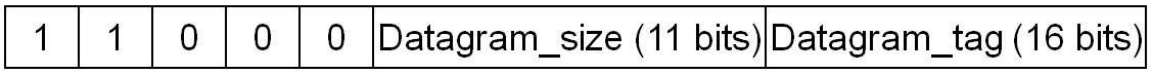


Figure 16: Initial fragment of Adapt header, from [9].

The layers of 6LoWPAN networks is shown in Figure 18. The packet is adapted to 6LoWPAN packet in the 6LoWPAN adaptation layer of 6 LowPAN networks. In the gateway (usually a router), the frames are adapted to IEEE 802.15.4 frame. After converting to IEEE 802.15.4 frame, the packet can be transmited into high speed networks (such as Internet).

6.2 Header Compression

The space left for data is already very small. If the Adapt header is introduced, the space will become smaller, i.e. 26 or 27 bytes. To guarantee efficiency, the IPv6 header shown in

1	1	1	0	0	Datagram_size (11 bits)	Datagram_tag (16 bits)
Datagram_offset (8 bits)						

Figure 17: Noninitial fragment of Adapt header, from [9].

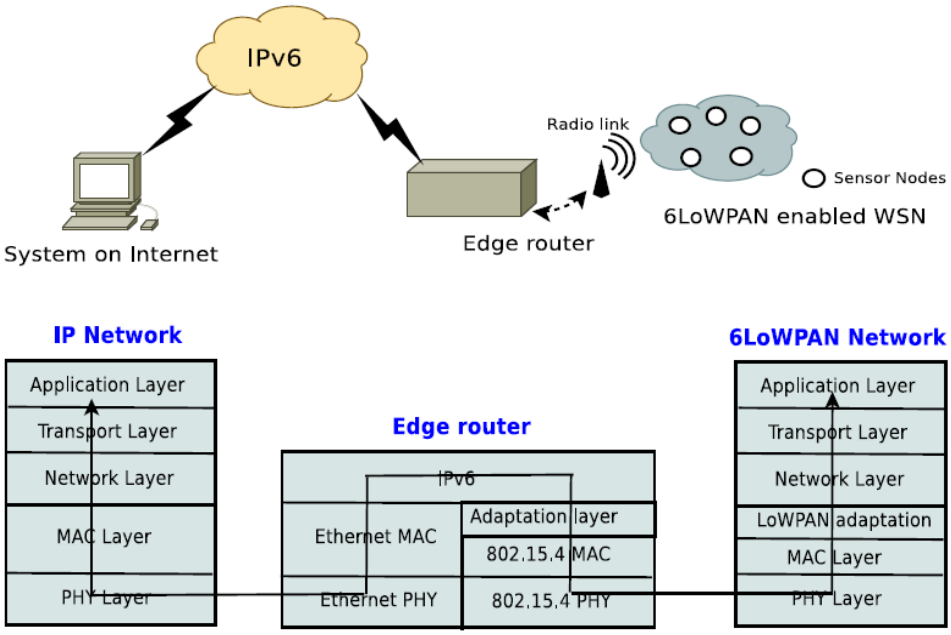


Figure 18: Dual stack, integrating 6LoWPAN with IPv6, from [17].

Figure 12 must be compressed.

Hui et al [6] define an encoding format for 6LoWPAN called LOWPAN_IPHC. LOWPAN_IPHC assumes the following will be the common case for 6LoWPAN:

1. Version is 6.
2. Traffic label class and flow label are both zero.
3. Payload length can be inferred from low layers.
4. Hop limit will be set to a well-known value by the source.
5. Addresses assigned to 6LoWPAN prefix will be formed using the link-local prefix or a small set of routable prefixes assigned to the entire 6LoWPAN.
6. Addresses assigned to 6LoWPAN interfaces are derived directly from either the 64-bit extended or 16-bit short IEEE 802.15.4 MAC addresses [6].

This results in the IPv6 40 byte header being reduced to 2 bytes for 6LoWPAN [17].

6.3 Addressing

An IPv6 address consists of two parts: the first part is a 64 bit prefix, the second part is a 64 bit interface ID. The prefix is formed using the link-local address $fe80 :: /64$ (this notation means the first 16 bits are $FE80_{16}$, followed by 48 zeroes) [9]. The interface ID is derived from the MAC address. 16 bits $fffe$ follow the first 24 bits of the MAC address, which are then followed by the remaining 24 bits of the MAC address. IEEE 802.15.4 has two forms of MAC address, 64-bit EUI-64 (Extended Unique Identifier-64) and 16-bit short addresses. If the MAC address uses a 64-bit long address, the 6LoWPAN interface ID directly uses the MAC address as its interface ID. If the MAC address uses a 16-bit short address, the 6LoWPAN interface ID first uses the 16-bit short address and PAN ID (Personal Area Network Identifier) to generate a 48-bit pseudo MAC address. Then the interface ID is derived from this pseudo MAC address as discussed above. The 6LoWPAN address is shown in Figure 19.

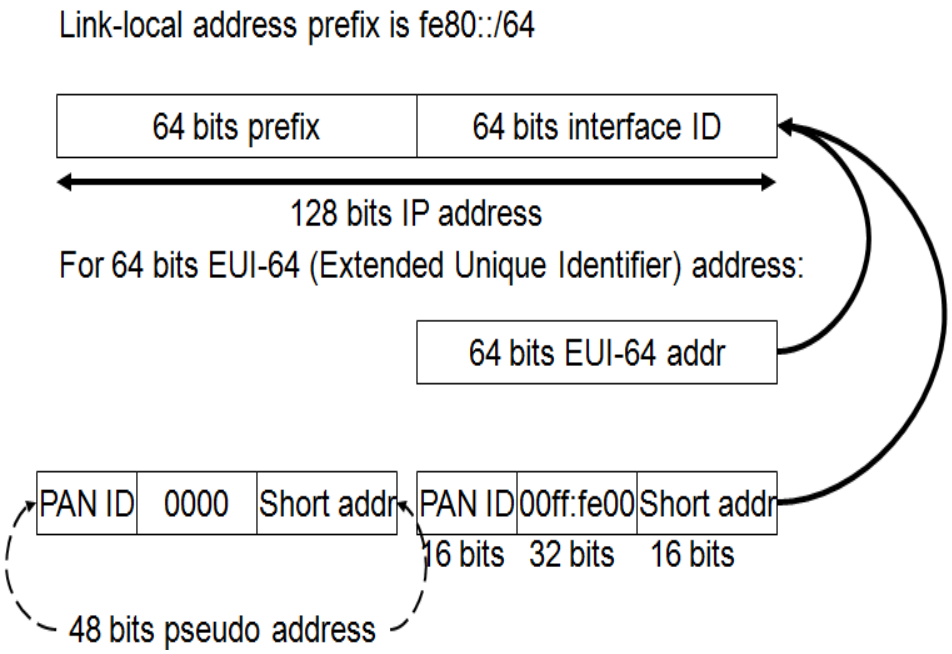


Figure 19: 6LoWPAN addressing, based on the description in RFC4944 [9].

7 Open Problems

There are other issues for 6LoWPAN. These include enabling 6LoWPAN in a dynamic environment where sensor nodes are moving, and where neighbour discovery is required. Table management is also a challenge for 6LoWPAN running on limited resource sensor nodes. Another open problem is providing a secure method for forwarding IPv6 packets in 6LoWPAN networks.

8 Conclusion

We surveyed the commonly used protocols for wireless sensor networks. The assumption is that sensor nodes are in a fixed location, with strong restrictions on the energy use by wireless transceivers typically running on 1.5 V batteries. We found that Berkeley Medium Access Control (B-MAC), defined by Polaster et al in 2004, is widely used, is energy efficient and with a low memory footprint in both RAM and ROM. A comprehensive survey by Langendoen and Meier [2010] showed that a protocol called WiseMAC (Wireless Sensor MAC) has the best energy-latency tradeoff among five channel polling protocols (including B-MAC) for very low data rates (i.e. less than 1 message received per second).

For routing, Woo et al [2003] showed that minimum transmission (MT) routing with a window mean with exponentially weighted moving average (WMEWMA) estimator performed best among six routing algorithms. These algorithms were tested in a 50-node network with eight foot spacing and a neighbour table size of 30.

Our survey concludes with a review of the 6LoWPAN (IPv6 over Low power Wireless Personal Area Networks) protocol. 6LoWPAN uses the IEEE 802.15.4 packet size of 127 bytes for sensor nodes while preserving other IPv6 attributes. An adaptation layer in edge router handles the fragmentation necessary to split the 1280 byte IPv6 packets into smaller 6LoWPAN packets, and vice versa. An experimental evaluation of four sensor networks [Hui and Culler, 2010] with an average of 139 nodes using 6LoWPAN on 16-bit sensor nodes with 10 KB of RAM showed an average packet delivery success rate of 99.94% with a 0.4% duty cycle.

References

[1] `http://www.snyders.us/ipv6-security.htm`. accessed May 20, 2011.

[2] Giuseppe Anastasi, Marco Conti, Mario Di Francesco, and Andrea Passarella. Energy conservation in wireless sensor networks: A survey. *Ad Hoc Networks*, 7(3):537–568, 2009.

[3] John-Paul Arp and Bradford G. Nickerson. End-to-end acknowledgement for data collection in wireless sensor networks. In *Proc. of Communication Networks and Services Research (CNSR-2010), Montreal.*, pages 93–101.

[4] Amre El-Hoiydi and Jean-Dominique Decotignie. Wisemac: An ultra low power mac protocol for multi-hop wireless sensor networks. In *ALGOSENSORS*, pages 18–31, 2004.

[5] Gertjan P. Halkes and Koen Langendoen. Crankshaft: An energy-efficient mac-protocol for dense wireless sensor networks. In *EWSN (European Conference on Wireless Sensor Networks)*, pages 228–244, 2007.

[6] Ed. J. Hui and P. Thubert. Compression Format for IPv6 Datagrams in Low Power and Lossy Networks. Technical report, IETF, February 2011. draft-ietf-6lowpan-hc-15, updates 4944 (if aapproved).

[7] Philo Juang, Hidekazu Oki, Yong Wang, Margaret Martonosi, Li-Shiuan Peh, and Daniel Rubenstein. Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with zebranet. In *Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 96–107, 2002.

[8] Koen Langendoen and Andreas Meier. Analyzing mac protocols for low data-rate applications. *TOSN (Transactions On Sensor Networks)*, 7(2), 2010.

[9] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler. Transmission of IPv6 Packets over IEEE 802.15.4 Networks. Technical report, IETF, September 2007. request for comments RFC4944.

[10] Joseph Polastre, Jason L. Hill, and David E. Culler. Versatile low power media access for wireless sensor networks. In *SenSys–The ACM Conference on Embedded Networked Sensor Systems*, pages 95–107, 2004.

[11] Proceedings of IEEE. *IPv6 in Low-Power Wireless Networks*, September 2010.

[12] Tijs van Dam and Koen Langendoen. An adaptive energy-efficient mac protocol for wireless sensor networks. In *SenSys–The ACM Conference on Embedded Networked Sensor Systems*, pages 171–180, 2003.

[13] Alec Woo and David Culler. *Evaluation of Efficient Link Reliability Estimators for Low-Power Wireless Networks. Technical Report UCB//CSD-03-1270.* U.C. Berkeley Computer Science Division, September 2003.

[14] Alec Woo, Terence Tong, and David E. Culler. Taming the underlying challenges of reliable multihop routing in sensor networks. In *SenSys–The ACM Conference on Embedded Networked Sensor Systems*, pages 14–27, 2003.

[15] Wei Ye, John S. Heidemann, and Deborah Estrin. An energy-efficient mac protocol for wireless sensor networks. In *INFOCOM–International Conference on Computer Communications*, 2002.

[16] Wei Ye, Fabio Silva, and John S. Heidemann. Ultra-low duty cycle mac with scheduled channel polling. In *SenSys (The ACM Conference on Embedded Networked Sensor Systems)*, pages 321–334, 2006.

[17] Lohith Y.S. 6lowpan: Wireless internet of things. Presentation at Department of ECE, Indian Institue of Science, Bangalore, November 2010.