Enhanced Power Saving Mode for Low-Latency Communication in Multi-Hop 802.11 Networks

V. Vukadinovic[a,*], I. Glaropoulos[a], S. Mangold[a]

[a] *Disney Research Zurich, Stampfenbachstrasse 48, 8006 Zurich, Switzerland*

Abstract

The Future Internet of Things (IoT) will connect billions of battery-powered radio-enabled devices. Some of them may need to communicate with each other and with Internet gateways (border routers) over multi-hop links. While most IoT scenarios assume that for this purpose devices use energy-efficient IEEE 802.15.4 radios, there are use cases where IEEE 802.11 is preferred despite its potentially higher energy consumption. We extend the IEEE 802.11 Power Saving Mode (PSM), which allows WLAN devices to enter a low-power doze state to save energy, with a traffic announcement scheme that facilitates multi-hop communication. The scheme propagates traffic announcements along multi-hop paths to ensure that all intermediate nodes remain awake to receive and forward the pending data frames with minimum latency. Our simulation results show that the proposed Multi-Hop PSM (MH-PSM) improves both end-to-end delay and doze time compared to the standard PSM; therefore, it may optimize WLAN to meet the networking requirements of IoT devices. MH-PSM is practical and software-implementable since it does not require changes to the parts of the IEEE 802.11 medium access control that are typically implemented on-chip. We implemented MH-PSM as a part of a WLAN driver for Contiki OS, which is an operating system for resource-constrained IoT devices, and we demonstrated its efficiency experimentally.

Keywords: Internet of Things, IEEE 802.11, power saving, ad hoc networks, multi-hop networks

1. Introduction

Nowadays almost every desktop computer, laptop, tablet, and smartphone is connected to the Internet. The emergence of the Internet of Things (IoT) will provide global IP connectivity to a broader variety of devices, such as entertainment electronics, wearable sport gadgets, home appliances, and industrial

*Corresponding author

Email addresses: vvuk@disneyresearch.com (V. Vukadinovic), stefan@disneyresearch.com (S. Mangold)

Figure 1: Application scenario in which smart radio-enabled toys communicate with decorative lightning (© Disney).

sensors. Some of these devices are portable, battery-powered, and need to connect wirelessly to surrounding devices and Internet gateways. The wireless communication may significantly contribute to their overall battery consumption, especially in the case of constrained embedded devices. Therefore, minimizing the energy consumption of wireless interfaces and networking protocols is one of the prerequisites for the IoT.

The IEEE 802.15.4 standard [1] defines a low-power physical layer, which is the foundation of ZigBee, a candidate wireless technology for the IoT. Despite being energy-efficient, ZigBee might not be the best wireless technology for all IoT devices and applications. Wireless LAN, which is based on IEEE 802.11 standard [2], dominates the consumer electronics market and any IoT device that needs to connect to smartphones, tablets, TVs, set-top boxes, game consoles, and toys would benefit from WLAN connectivity. Also, some sensors that operate at high sampling rates, such as those used in seismic monitoring and imaging, may generate large amounts of data that cannot be transmitted using ZigBee due to its limited throughput, but can easily be transmitted by WLAN.

The energy consumption of WLAN is relatively high compared to ZigBee and may quickly drain the battery of a device. Long battery recharge/replacement

cycles are preferred for cost and convenience reasons. For example, a survey has shown that 51% of electronic toy consumers are concerned about the battery replacement costs [3]. The major sources of unnecessary energy consumption in WLAN are packet collisions, overhearing, control packet overhead, and idle listening. Among those, idle listening is particularly wasteful since it consumes energy even when there is no traffic in the network — the radio must perform idle listening continuously in order to detect arriving packets. The energy consumption of idle listening in WLAN is comparable to that of packet transmission and reception [4]. To alleviate the problem, the 802.11 standard [2] specifies a Power-Saving Mode (PSM) that allows an idle 802.11 station to transition to a low-power doze state by switching off its radio transceiver. The role of 802.11 PSM is similar to that of Radio Duty Cycling (RDC) in 802.15.4. There are some notable differences: RDC typically operates below MAC, directly on top of the 802.15.4 PHY layer. It may include information from the MAC layer, in which case MAC and RDC are cross-optimized as in [5], but it can also be isolated from MAC. With RDC, a radio can be switched on and then rapidly switched off after a few milliseconds if no activity is detected on the channel. The 802.11 PSM is part of the MAC layer management entity. The intervals in which PSM alternates between doze and awake states are typically measured in tens and hundreds of milliseconds: All 802.11 stations wake up synchronously at the beginning of a beacon interval, listen for traffic announcements from other stations that have data packets destined to them, and announce their own data packets (if any) destined to other stations. If a station does not receive any traffic announcements and it does not have any buffered packets that need to be transmitted in the current beacon interval, it returns to the doze state.

The IEEE 802.11 standard specifies the details of PSM for the infrastructure/BSS mode (Basic Service Set with an access point) and the ad hoc/IBSS mode (Independent Basic Service Set without an access point). Since it has been originally designed for single-hop communication in the infrastructure mode (from the access point to a station and vice versa), the PSM performs poorly in the ad hoc mode, especially in multi-hop networks [6, 7, 8]. When a data frame is forwarded over multiple hops, the PSM may significantly increase the delivery delay because only the next-hop station is notified about the pending frame via traffic announcements, while the stations on subsequent hops may remain in the doze state. Therefore, in each beacon interval the frame is forwarded over a single hop and has to be buffered before being forwarded further. Depending on the number of hops, the end-to-end delay may be long enough to affect time-sensitive applications. Another problem of PSM is that a station is occasionally forced to stay awake even though it has no frames to transmit or receive. The reason to stay awake is to respond to probe requests of devices that are actively scanning the medium when attempting to discover and join networks. For example, if there are two stations in an 802.11 ad hoc network, at least one of them would have to remain awake at any time, which limits the sleep time to at most 50%. Hence, the 802.11 PSM is not suitable for low-energy low-latency multi-hop communication, which is a common requirement for the IoT.

In this paper, we address the problem of increased frame delays due to PSM in multi-hop ad hoc networks. We propose a mechanism that wakes up downstream stations so that data frames can be forwarded over multiple hops in a single beacon interval. This is achieved by instructing each station along the path to forward the traffic announcement to its downstream neighbor. The proposed mechanism significantly reduces the end-to-end delay, especially for bursty traffic where intermediate stations may move to the doze state between two consecutive traffic bursts. We also question the 802.11 standard requirement for a station to stay awake to respond to probe requests. We describe a mechanism that enables actively scanning stations to discover an ad hoc network in which no station is required to stay awake for the entire duration of the beacon interval. The proposed mechanisms enhances the standard PSM to what we call Multi-Hop PSM (MH-PSM). MH-PSM does not prevents stations to interoperate with those that employ standard PSM since it does not alter the state machine, the frame formats, and other important protocol mechanisms. MH-PSM is also software-implementable: It does not require modifications to the parts of the 802.11 MAC protocol that are usually implemented on-chip, such as the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) medium access protocol. We implemented MH-PSM as a part of our WLAN driver for the Contiki operating system [9]. The paper provides a practical demonstration that, with few simple modifications, WLAN ad hoc mode may become a compelling technology for some IoT applications. A concise version of this paper was published in [10].

The rest of the paper is organized as follows: Section 2 provides an overview of the standard 802.11 PSM. In Section 3, we describe MH-PSM and discuss deployment and standard compatibility issues. The performance of MH-PSM is evaluated in Section 4 using simulations, and in Section 5, we describe our testbed implementation of MH-PSM and experimental results. Section 6 gives a brief overview of related work. Finally, Section 7 concludes the paper.

2. Power-saving mode for 802.11 ad hoc networks

In the standard 802.11 PSM for ad hoc/IBSS networks, time is divided into periods called beacon intervals. Each station wakes up at the beginning of a beacon interval and starts a back-off procedure in an attempt to transmit a beacon. If a station receives a beacon from another station before its back-off timer expires, it cancels the pending beacon transmission. The Timing Synchronization Function (TSF) uses the time-stamped beacons to synchronize clocks among stations, to ensure that all stations wake up at the same time. Following the beacon exchange, each station stays awake for an ATIM window interval as shown in Fig. 2. During the ATIM window, stations announce pending data frames to their neighbors using unicast Announcement Traffic Indication Messages (ATIMs). ATIMs are sent using the 802.11 Distributed Coordination Function (DCF) that operates with a CSMA/CA channel access procedure. A station that receives an ATIM should respond with an ACK. Successful exchange of ATIM-ACK packets between two stations implies that they can now

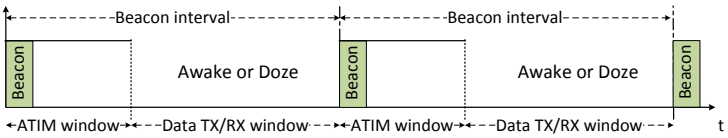

Figure 2: From [10]. The 802.11 PSM divides time into beacon intervals.

exchange any pending data frames and thus both should stay awake until the next beacon interval. Stations that neither send nor receive any ATIM frame during the ATIM window will move to the doze state for the rest of the beacon interval. After the end of an ATIM window, all stations that remain awake will follow the normal DCF procedure to transmit and receive data frames.

The described PSM protocol has many drawbacks: It uses DCF, which may waste scarce battery resources and bandwidth due to frame collisions and increase the frame delay due to back-offs. A station that has pending data frames must estimate if the receiving station is using PSM. ATIMs should be sent only to stations that are using PSM. Stations that are not in PSM will not respond with an ACK, which will trigger undesirable re-transmissions. The standard however does not specify how to estimate if a station is using PSM or not. When a station successfully transmits or receives an ATIM frame during an ATIM window, it must stay awake during the entire rest of the beacon interval. At low loads, this approach results in a higher energy consumption than necessary. Another shortcoming is that all stations in an IBSS must use the same fixed ATIM window size, which is set at the time when the IBSS is created, as well as identical beacon intervals. Since the ATIM window size critically affects the throughput and energy consumption, the fixed ATIM window does not perform well in all situations, as shown in [11]. Some of these drawbacks have been addressed in previous work, which are discussed in Section 6. This paper, however, addresses the problem of end-to-end delay on multi-hop paths, as described in the following.

Consider a typical multi-hop scenario where station A needs to send a single-frame message to station D using intermediate stations B and C as relays (Fig. 3). In the first beacon interval, station A announces the data frame to station B using an ATIM. Station B acknowledges the ATIM an remains awake so that it can receive the data in the period that follows the ATIM window. Assume that station C has not received any traffic announcements and, therefore, it enters the doze state. Since station B is not able to forward the frame to station C in the current beacon interval, it has to wait for the start of the next beacon interval to send an ATIM to station C. Following a successful ATIM-ACK exchange, the frame is forwarded to C. Station D will receive the frame in the third beacon interval. The resulting increased end-to-end delay may considerably affect applications with strict latency constraints, which is undesirable. Therefore, enabling PSM in multi-hop ad hoc networks must be combined with effective mechanisms for mitigating its effect on the resulting packet delays.

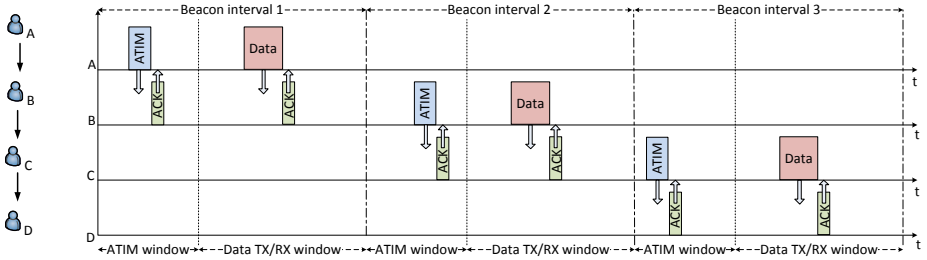

Figure 3: From [10]. Multi-hop forwarding in standard 802.11 PSM may cause a delay of several beacon intervals.

3. Enhanced 802.11 PSM for Multi-Hop Communication

In the scenario described above, the data frame sent by A must be buffered at B before it is relayed to C in the following beacon interval. This could be avoided if there was a way for B to, upon receiving the ATIM from A, send an early ATIM to C and D to inform them about the pending data frame at A. This is what our low-latency multi-hop PSM (MH-PSM) aims to achieve.

Before introducing MH-PSM, we describe the format of ATIM frames. An ATIM frame contains a MAC header, whose structure is common to all management frames as shown in Fig. 4. The frame body of an ATIM is empty. The header includes three address fields: Address 1 contains the MAC address of the ATIM receiver. Address 2 contains the MAC address of the ATIM sender. Address 3 may contain different information depending on the type of the management frame and network mode (BSS, IBSS, or mesh). The Address 3 field of ATIM frames contains the BSSID (BSS identifier) of the IBSS. In case of *group-addressed* (i.e., broadcast) ATIMs, the BSSID is used to verify that the frame originated from a station in the IBSS of which the receiving station is a member. In case of individually addressed (i.e., unicast) ATIMs, the BSSID is not used at the receiver [2].

3.1. Proposed Extension: Multi-Hop PSM (MH-PSM)

We propose that, in order to inform all stations along the path to D about the pending data frame, the station A writes the MAC address of D into the Address 3 field of the ATIM frame sent to B. Methods to resolve the MAC address of D from its IP address are discussed later in this section. Upon

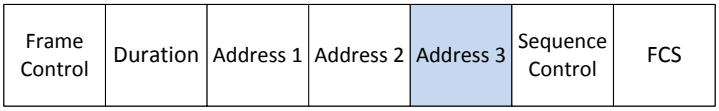

Figure 4: From [10]. Structure of the ATIM frame. The Address 3 field can be used for the MAC address of the end destination.

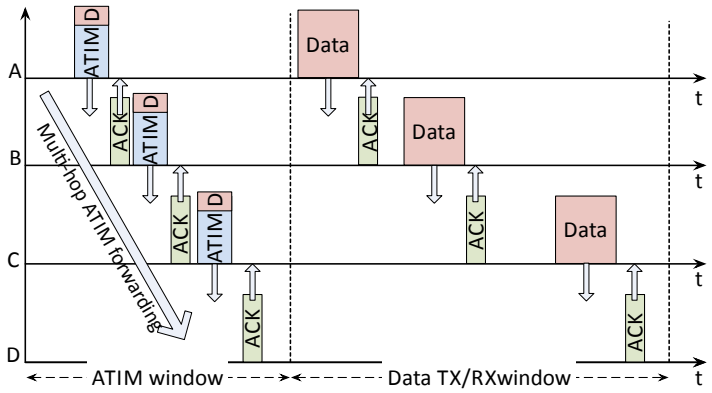

Figure 5: From [10]. The proposed multi-hop forwarding mechanism allows data frames to be forwarded end-to-end in a single beacon interval.

receiving the ATIM, B inspects the Address 3 field to derive the final destination of the data frame announced by that ATIM. It retrieves the MAC address of D from the Address 3 field, resolves it to the IP address of D, and consults the routing table to find out that C is the next hop on the path to D. Then B creates an ATIM frame for C with the MAC address of D inside the Address 3 field. When C receives the ATIM from B, it uses the same procedure to create an ATIM for D. In this way, a chain of ATIM transmissions is created along the multi-hop path to wake up all relays and the destination of the data frame. Following the end of the ATIM window, the data frame can be forwarded end-to-end in the current beacon interval since all stations on the path are in the awake state. The procedure is illustrated in Fig. 5. The ATIM chain may not reach the end destination: It may terminate at the end of the ATIM window or upon reaching a station that cannot resolve the MAC address of the destination. In that case, the data frame will be forwarded as far as the furthest station that has received the ATIM. Nevertheless, MH-PSM may significantly decrease the end-to-end delay in lightly-loaded multi-hop networks because, unlike with the standard PSM, data frames are forwarded over multiple hops in a single beacon interval.

3.2. Address 3 Resolution

The sending station A needs to write the MAC address of the destination D into the Address 3 field of the ATIM sent to B. Therefore, A needs to resolve the MAC address of D from its IP address. Since the paper targets Internet of Things (IoT) and smart toy communication scenarios, we assume that IPv6 is used. The IPv6 protocol suite includes the Neighbor Discovery (ND) protocol [12], which provides address resolution, next-hop determination, and duplicate address detection. Address resolution enables stations to determine MAC addresses of their neighbors given only their IP addresses. The neighbor

solicitation messages, which are used for address resolution, are sent via multicast. The ND protocol is not designed with multi-hop ad hoc networks in mind. A node in such network is able to broadcast to other nodes within its radio range, but the communication is non-transitive. Therefore, a wireless ad hoc network is a Non-Broadcast Multi-Access (NBMA) structure with generally no network-wide multicast capabilities. The network solicitation messages are not forwarded in an IBSS. Hence, station A is only able to resolve MAC addresses of its immediate neighbors, but not of D, which is multiple hops away. There are several proposals to extend the capabilities of the ND protocol to multi-hop ad hoc networks [13] and 6LoWPAN (IPv6 over Low power Wireless Personal Area Networks [14]) in particular [15]. These proposals include mechanisms for multi-hop Duplicate Address Detection (DAD), which allows a station to check the uniqueness of an IP address in an n-hop neighborhood. The multi-hop DAD can also be used for multi-hop address resolution: Station A may initiate multi-hop DAD for the IP address of D. Upon receiving a DAD request, D will respond with a DAD confirmation message that contains its MAC address. It this way, A can resolve the MAC address of D based on its IP address. Each station maintains a cache of resolved addresses, which limits the need for network-wide multi-hop address resolution.

3.3. Backward-Compatibility

Backward-compatibility with the standard PSM for IBSS is ensured since MH-PSM does not violate neither the frame formats nor the protocol operations. Stations that implement standard PSM will not check the Address 3 field of received ATIMs and, therefore, the chain of ATIMs will terminate at such stations. This diminishes some of the delay improvements, but otherwise does not prevent or impair communication.

To better understand how standard PSM and MH-PSM may coexist, consider a scenario where station A sends data to station E using B, C, and D as intermediate relays. Assume that all the stations except station B use MH-PSM. When B receives an ATIM from A (with E's MAC address in the Address 3 field), it will not immediately create an ATIM for C: Instead, it will wait for the data packet to arrive and then, at the beginning of the next beacon interval, it will create an ATIM for C. This ATIM will not contain E's address in its Address 3 field because B runs standard PSM. Once it receives the ATIM, the MH-PSM-enabled station C (and all subsequent downstream stations) will fall back to the standard PSM.

3.4. Support for Network-Wide Broadcasts

As pointed out earlier in this section, in case of broadcast ATIMs, the Address 3 field must contain the BSSID, which is used by the receiver to verify that the frame originated from a station in the IBSS of which the receiver is a member. Therefore, it cannot be used to store the MAC address of the final destination. Broadcast ATIMs are mostly used to announce *link-local* broadcasts (e.g., Node Solicitation messages of the Node Discovery protocol). Hence, they

do not need to be forwarded over multiple hops since the announced broadcast is aimed at stations that are one hop away from the sender. However, when *network-wide* broadcasts (e.g., Route Request messages of the AODV routing protocol) are announced, the broadcast ATIMs should be forwarded over multiple hops to ensure that all stations in the network remain awake following the end of the ATIM window. To support such broadcasts in MH-PSM, there should be a field in the ATIM header (other than Address 3 field) that a sender could use to declare if the broadcast ATIM is announcing a link-local or a network-wide broadcast. A possible solution would be to amend the 802.11 standard to include a new management frame subtype Multihop ATIM (analogous to Multihop Action frames [2]), which would contain an additional field in the header for this purpose.

3.5. Sleep On Beacon Transmission (SoBT)

We propose an additional mechanism to increase the doze time of idle stations. The 802.11 standard mandates that a station that wins the back-off at the beginning of a beacon interval and subsequently transmits a beacon, should remain awake until the end of the beacon interval. This is considered necessary to ensure that the IBSS to which that station belongs can be discovered by the devices that employ active scanning. Most portable battery-powered devices, such as smartphones and tablets, use active instead of passive scanning to conserve energy. Unlike with passive scanning, when the scanning device spends substantial time listening for incoming beacons, with active scanning the device may wake up for a short period of time, transmit Probe Requests on different channels, and return to the doze state. At least one station in the IBSS must reply with a Probe Response containing the SSID of the IBSS, which can then be added to the list of known SSIDs. If all stations in an IBSS move to the doze state, there is no one to reply to the Probe Requests and, therefore, the IBSS might be invisible to the devices that employ active scanning. The probability that a station transmits a beacon increases as the number of neighbors decreases. Hence, in a small and/or sparse IBSS network, a station might win beacon back-offs in many consecutive beacon intervals and is forced to stay awake even though there is no traffic in the network. In the following, we discuss how this requirement of the 802.11 standard can be relaxed in practice.

A device that employs active scanning should repeat the following procedure for each channel to be scanned (see [2] for a full description):

(a) Wait until the ProbeDelay time has expired or a PHYRxStart.indication has been received.

(b) Perform the basic DCF access procedure and send Probe Request to the broadcast destination address.

(c) Clear and start a ProbeTimer.

(d) If PHY-CCA.indication (busy) has not been detected before the ProbeTimer reaches MinChannelTime, then scan the next channel, else when ProbeTimer reaches MaxChannelTime, process all received Probe Responses.

Hence, ProbeDelay is the delay prior to transmitting a Probe Request on a new channel, MinChannelTime and MaxChannelTime are, respectively, the minimum and the maximum amount of time spent on that channel after the Probe Request transmission. Assuming that PHYRxStart.indication has not been received in a), the device spends at least the time interval $T_{min} = ProbeDelay + t_{TX}(ProbeRequest) + MinChannelTime$ on each channel. If the device receives a beacon during the interval T_{min}, the SSID advertised in the beacon will be added to the list of discovered SSIDs. Since WLAN channels (2.4 GHz band) are overlapping, the minimum time that the scanning device spends listening on a channel is even longer: While being tuned to channel i, a WLAN device is able to receive transmissions on channels $[i-k, i+k]$, where typically $k = 2$ or $k = 3$. Therefore, assuming that neighboring channels are scanned consecutively, the minimum total time during which the scanning device is able to receive transmissions on channel i is $T^{tot}_{min} = T_{min} \times (2k + 1)$ for $k + 1 \leq i \leq 13 - k$. For channels that are at the edges of the 2.4 GHz band (e.g., $i = 1$ and $i = 13$), $T^{tot}_{min} = T_{min} \times (k + 1)$. Instead of staying awake for the entire beacon interval to respond to Probe Requests, an idle station in PSM mode could wake up periodically and transmit what we call *intra-beacons*. As long as the intra-beacon interval T_{IB} is shorter than T^{tot}_{min} the scanning device will be able to receive an intra-beacon and discover the IBSS. Hence, unlike regular beacons that are sent at the beginning of each beacon interval (if the station wins the back-off), intra-beacons are transmitted during the beacon interval. The station generates intra-beacons only in beacon intervals in which it i) wins the beacon back-off and subsequently transmits a regular beacon and ii) does not receive or send any ATIM frames during the ATIM window. If the beacon interval is shorter than T^{tot}_{min} there is no need to transmit intra-beacons because regular beacons are frequent enough to be received by the scanning device.

We investigated the feasibility of the proposed SoBT scheme for cases where scanning devices run Android and Apple iOS, which today cover more than 90% of the current smartphone market, according to [16]. An Android phone may either perform soft-scanning or hard-scanning depending on whether its Wi-Fi driver implements active scanning or not. If active scanning is not supported by the driver, the phone performs so-called soft-scanning, which is implemented in /net/mac80211/scan.c of the Android kernel. Otherwise, the hard-scanning is performed. With soft-scanning, an Android phone always waits for ProbeDelay before it sends a Probe Request on a new channel regardless if PHYRxStart.indication has been received or not. This is a departure from the step a) of the standard scanning procedure.

The ProbeDelay and MinChannelTime are 30 ms each. Therefore, an Android phone spends $T_{min} = 60$ ms on each channel, assuming that the time to transmit a Probe Request $t_{TX}(ProbeRequest)$ is negligible. The described soft-scanning procedure is however rarely used because Wi-Fi drivers of most Android phones implement hard-scanning. We examined the hard-scanning procedure of a Samsung Galaxy S3 phone, which is equipped with a Broadcom BCM4334 Wi-Fi chip. The Broadcom's *bcmdhd* driver for Android does not wait for ProbeDelay before sending a Probe Request on a new channel (step

a) of the standard scanning procedure). When the phone is not connected to any WLAN network, the driver scans for new networks by sending two consecutive Probe Requests and waiting $MinChannelTime = 40$ ms for Probe Responses on each channel. Therefore, the Samsung Galaxy S3 phone (and any other Android phone whose hard-scanning is performed by the *bcmdhd* driver) spends $T_{min} = 40$ ms on each channel, assuming that the time to transmit two consecutive Probe Request is negligible. Apple's iPhone 5 is equipped with the same Broadcom BCM4334 Wi-Fi chip as Samsung Galaxy S3, but the details of the active scanning procedure are not readily available since the driver source code is not available. We performed a set of measurements, which indicate that, when iPhone 5 is not connected to any WLAN network, it scans for new networks by sending one Probe Request and waiting $MinChannelTime = 20$ ms for Probe Responses on each channel. Therefore, $T_{min} = 20$ ms, assuming that $t_{TX}(ProbeRequest)$ is negligible.

Our tests with both Samsung Galaxy S3 and iPhone 5 have shown that, during active scanning on channel i, the phones are able to receive beacons on channels $[i - 2,\ i + 2]$. Therefore, in an IBSS that uses channel $i = 1$, it is sufficient to transmit intra-beacons every $T_{IB} \leq 3T_{min}$. Hence $T_{IB} \leq 120$ ms and $T_{IB} \leq 60$ ms for Samsung Galaxy S3 and iPhone 5, respectively. In an IBSS that uses channel $3 \leq i \leq 11$, it is sufficient to transmit intra-beacons every $T_{IB} \leq 5T_{min}$. Hence $T_{IB} \leq 200$ ms and $T_{IB} \leq 100$ ms for Samsung Galaxy S3 and iPhone 5, respectively. We implemented the SoBT scheme in MH-PSM as described in Section 5. To test the scheme, we created an IBSS containing a single dozing station whose inter-beaconing period T_{IB} was set according to the determined values. Our tests confirmed that a single scanning round is sufficient to receive an intra-beacon and, therefore, discover the IBSS without forcing the station to stay awake and continuously listen for Probe Requests. [1]

4. Simulation Results

We extensively evaluated MH-PSM and compared its performance to standard PSM using simulation. The performance is measured in terms of end-to-end delay, doze time ratio, ATIM overhead, and packet delivery ratio, as defined below. The simulation setup and the results are described in the following.

End-to-End Delay is the average time required to forward a data frame from a source to its destination over multiple hops. It is averaged over all successfully delivered data frames.

Doze Time Ratio is the percentage of beacon intervals in which a station enters the doze state and it is closely related with the energy consumption. It is averaged over all stations that participate in traffic forwarding.

[1] In case of iPhone 5, the reception of an intra-beacon caused the SSID of the IBSS to be added to the list of known SSIDs. In case of Samsung Galaxy S3, we were able to confirm the reception of an intra-beacon, but the SSID was not added to the list because Android's Wi-Fi Manager (as of version 4.2.2) does not support IBSS mode.

Table 1: Default simulation parameters.

Parameter	Value
Channel model	unit disk
IEEE 802.11 PHY mode	11 Mb/s (802.11b)
MAC buffer size	100 frames
Beacon interval	200 ms
ATIM window	20 ms
Data frame payload size	500 Bytes
Traffic model	Poisson(λ), λ=2.5, 5, 10 frames/s

ATIM Overhead is the average number of ATIM frames sent per one successfully delivered data frame. The relative ATIM overhead of MH-PSM is the ratio of ATIM overheads obtained with MH-PSM and standard PSM.

Packet Delivery Ratio is the percentage of data frames that are successfully delivered to the end destination. A station may drop a data frame if it exceeds the maximum number of retransmissions.

4.1. Simulation Setup

We implemented and evaluated MH-PSM in Jemula802 [17], which is a Java-based 802.11 protocol simulator. We consider a network of static regularly spaced 802.11 stations that are 50 m apart from each other. We assume a simple unit disk radio propagation model. We varied the radio range from 50 m to 150 m to influence the number of hops between source-destination pairs. The beacon interval and ATIM window size are 200 ms and 20 ms, respectively, unless stated otherwise. The data traffic is Poisson (exponentially distributed frame interarrival times) with fixed frame sizes of 500 B. The number of traffic flows and mean frame interarrival time are varied to control the load in the network. We ensured that the duration of each simulation run is sufficient to make the variations in time-moving averages insignificant. The main simulation parameters are summarized in Table 1.

4.2. Performance Results

Consider first the simple single-flow scenario shown in Fig. 6, where the station on the far left is sending data frames to the station on the far right over multiple hops. The radio transmission range is set to 50 m, 100 m, and 150 m in different simulation runs to create paths with two, three, and six hops,

Figure 6: Simulated network topology with a single flow. The transmission range is set to 50 m, 100 m, and 150 m in different simulation runs to produce paths with two, three, and six hops, respectively.

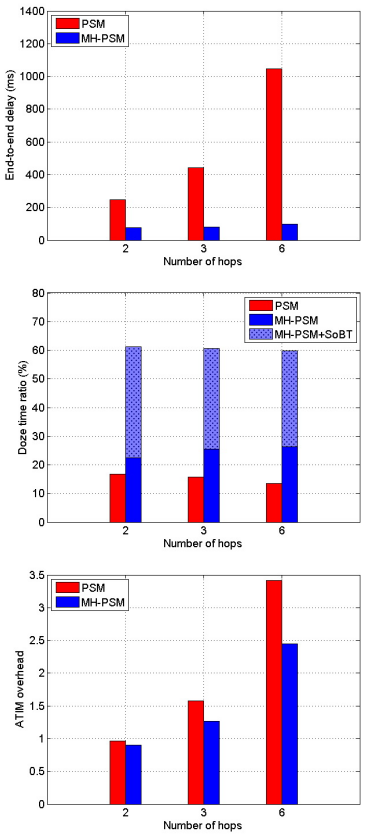

Figure 7: End-to-end delay, doze time ratio, and ATIM overhead for different number of hops.

respectively. On average, the sender is generating one frame every 200 ms ($\lambda = 5$ frames/s).

The results for the average end-to-end frame delay are shown in Fig. 7 (top). As expected, the delay increases with the number of hops. For the standard PSM it takes almost N beacon intervals to forward a frame over N hops. It may happen that a frame is forwarded over multiple hops in a single beacon interval: if its next-hop neighbor is awake, a station may immediately forward the frame to it, without waiting for the next beacon interval to send a traffic announcement. In a lightly loaded network, however, it is likely that the next-hop station is in the doze state, and therefore, the data frame has to be buffered. The results show that the delay is significantly shorter for MH-PSM. Although it slightly increases with the number of hops (due to processing in intermediate stations and increasing probability of collisions/retransmissions caused by hidden stations) the average delay is well below 200 ms, which is the duration of the beacon interval. As the number of hops increases from two to six, the percentage of frames that are forwarded end-to-end within a single beacon interval

marginally deceases from 100% to 99%, whereas for standard PSM drops from 92% down to zero.

The average doze time ratio is shown in Fig. 7 (middle). Even without SoBT, MH-PSM significantly increases the energy efficiency by allowing the stations to move to the doze state more often than standard PSM. The reason for this is that MH-PSM prevents excessive buffering of frames in intermediate stations (the number of frames in the station's queue is lower), which effectively decreases the traffic load and the probability of collisions/retransmissions. This shows that MH-PSM provides both shorter delay and lower energy consumption, which is a major improvement over standard PSM whose parametric adjustments/optimizations may only trade shorter delay for higher energy consumption and vice versa. When combined with SoBT, the doze time of MH-PSM surges to 60%. Hence, allowing stations to sleep after beacon transmissions is very effective in reducing idle listening.[2] Standard PSM forces a station to stay awake if it transmits a beacon, which can be a major source of energy waste, especially in sparse and in networks with many hidden terminals. Consider our two-hop scenario (A→B→C): If all the stations could hear each others transmissions, the probability that an arbitrary station transmits a beacon would be 1/3. However, since A and C are hidden from each other, they may both transmit beacons if their backoff timers expire before the backoff timer of B. Moreover, if beacon transmissions from A and C overlap in time, they will collide at B and, therefore, B will also transmit a beacon. Hence, without SoBT, there is a high probability that a station has to remain awake because it transmitted a beacon.

In Fig. 7 (bottom), we show the ATIM overhead for both PSM schemes. While the overhead for MH-PSM is comparable to that of standard PSM for the path with two hops, it is almost 30% lower in the six-hop case. To understand the reasons for this, consider a five-hop path from station A to station E via B, C, and D, as shown in Fig. 8. Assume that one frame is buffered at station A and one at station C. In the best-case scenario, it will take four beacon intervals and six ATIMs to deliver both frames to the destination under standard PSM. With MH-PSM however, it will only one beacon interval and four ATIMs to achieve the same because it creates a wave of ATIMs that flushes all buffered frames to the destination, as shown in Fig. 9. There are however scenarios where the ATIM overhead of MH-PSM is higher than that of the standard PSM even for paths with many hops. In standard PSM, a station sends a single ATIM to its neighbor to announce all data frames that it intends to forward to this neighbor, regardless of their end destinations. In MH-PSM, the station may send multiple ATIMs with different Address 3 fields to the neighbor if the pending data frames have different end destinations. For example, consider two flows whose eight-hop paths contain a common subset or relays, as shown in Fig. 10. In MH-PSM, the common relays may need to forward two ATIMs with different Address 3 fields to their next-hop neighbors in the same ATIM window. This is not the case in

[2]Note however, that during SoBT intervals, stations may need to wake up and transmit intra-beacons, which diminishes some of the energy saving gain.

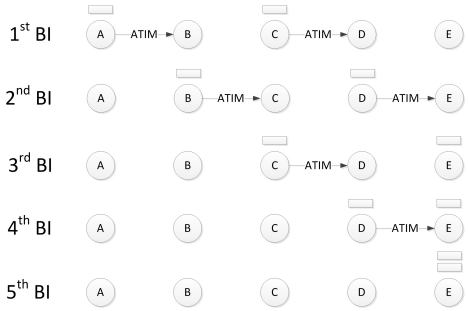

Figure 8: From [10]. Standard PSM requires four Beacon Intervals (BIs) and six ATIMs to deliver the frames buffered at A and C.

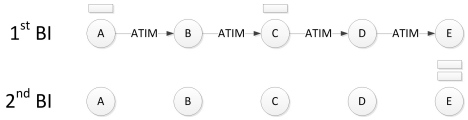

Figure 9: From [10]. MH-PSM requires only one Beacon Interval (BI) and five ATIMs to deliver the frames buffered at A and C.

standard PSM, where only one ATIM is sent. The results in Fig. 11 show that the ATIM overhead of MH-PSM is 20% higher. MH-PSM outperforms standard PSM in all other respects: The end-to-end delay is close to tenfold shorter, the doze time ratio is slightly higher, and the packet delivery ratio is significantly improved. Therefore, the relative ATIM overhead of MH-PSM had no bearing to the key performance metrics.

We next investigate the impact of beacon interval on the performance of PSM and MH-PSM. The results presented so far assume a beacon interval of 200 ms. We changed the interval to 100 ms and 400 ms and repeated the simulations for the basic scenario shown in Fig. 6 with the transmission range of 50 m (i.e. six hops). The average frame interarrival time is 200 ms regardless of the beacon interval. The results are summarized in Table 2. As expected, the frame delay for standard PSM increases linearly with the beacon interval because the time that frames stay buffered in the intermediate nodes is proportional to the beacon interval. The delay for MH-PSM also increases, but remains

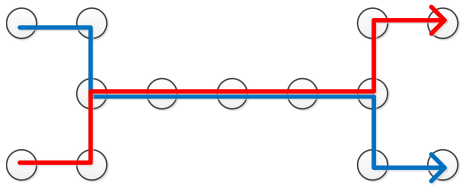

Figure 10: An example of two flows whose paths partially overlap.

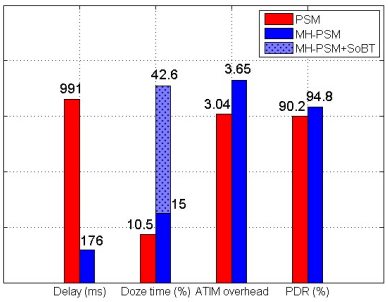

Figure 11: Performance of standard PSM and MH-PSM for the scenario with two flows whose paths partially overlap (Fig. 10).

much shorter than for standard PSM. The increase is due to the fact that MH-PSM does not guarantee that all frames will be delivered end-to-end in a single beacon interval: Some frames may be buffered along the path as in the case of standard PSM. The doze time decreases for both schemes because the number of idle (with no data traffic) beacon intervals decreases as they become longer. Another observation is that the packet delivery ratio of standard PSM decreases significantly for the longer beacon interval (from close to 100% for 100 ms to 93% for 400 ms), while for MH-PSM it remains close to 100%: For standard PSM, the number of buffered frames along the path increases with the duration of the beacon interval, which effectively increases the traffic load in the network and the probability of collisions. With MH-PSM, most frames are delivered end-to-end without buffering in intermediate nodes.

In the last two simulation scenarios, the average frame Interarrival Time (IAT) and the number of flows in the network are varied to evaluate the impact of traffic load on the performance of MH-PSM. The beacon interval is 200 ms regardless of the traffic load.

The results presented so far assume an average frame IAT of 200 ms. We changed the average IAT to 100 ms and 400 ms and repeated the simulations for the basic scenario shown in Fig. 6 with the transmission range of 50 m (i.e. six hops). The results are summarized in Table 3. The frame delay decreases for shorter IATs. This is because the doze time decreases with the traffic load, meaning that frames are more likely to be forwarded without buffering. In

Table 2: Performance of standard PSM and MH-PSM for different beacon intervals. Frame interarrival times is 200 ms. Frames are forwarded over six hops.

BI (ms)	Delay (ms)		Doze time (%)		ATIM overhead	
	PSM	MH-PSM	PSM	MH-PSM	PSM	MH-PSM
100	532	51	22	31+41	4.55	3.02
200	1047	99	13	26+33	3.42	2.45
400	2044	179	8	23+30	2.35	1.45

Table 3: Performance of standard PSM and MH-PSM for different frame interarrival times. Beacon interval is 200 ms. Frames are forwarded over six hops.

IAT (ms)	Delay (ms)		Doze time (%)		ATIM overhead	
	PSM	MH-PSM	PSM	MH-PSM	PSM	MH-PSM
100	1010	82	9	23+30	2.36	1.49
200	1047	99	13	26+33	3.42	2.45
400	1094	102	22	31+40	4.65	3.55

case of MH-PSM, nearly 100% of frames are transmitted end-to-end within a single beacon interval regardless of the IAT. The differences in frame delays for different IATs are due to the initial hold-up at the sender: For average IAT of 400 ms (twice the beacon interval), it is more likely that the sender is in the doze state when a frame is passed to the 802.11 MAC; therefore, the frame has to wait for the next beacon interval to be transmitted. For average IAT of 100 ms (two frames per beacon interval), many frames are transmitted immediately since the sender is likely to be awake due to earlier packer arrivals. The ATIM overhead also decreases for shorter IATs because it becomes unnecessary to send ATIMs for some of the frames: Only one ATIM is sent per hop per beacon interval regardless of the number of frames that need to be transmitted in that beacon interval.

We next consider the scenarios with multiple (i.e. 2, 4, and 8) intersecting flows in a grid topology shown in Fig.12. The transmission range is 50 m; therefore, frames are forwarded over six hops. The results in Table 4 show that the performance deteriorates with the number of flows. Transmissions of intersecting nodes are especially prone to collisions because they are surrounded by four active/forwarding stations that do not hear each other's transmissions (hidden stations). The impact of collisions on the performances of standard PSM and MH-PSM is somewhat different: While the frame delay for standard PSM remains unaffected by the number of flows, the delay for MH-PSM increases considerably (yet still remains relatively low). The reason is that collisions in intersecting nodes may disrupt the cut-through forwarding of data frames in MH-PSM. In the single-flow scenario, 100% of frames are forwarded end-to-end in a single beacon interval. For the eight-flow scenario, this percentage drops to

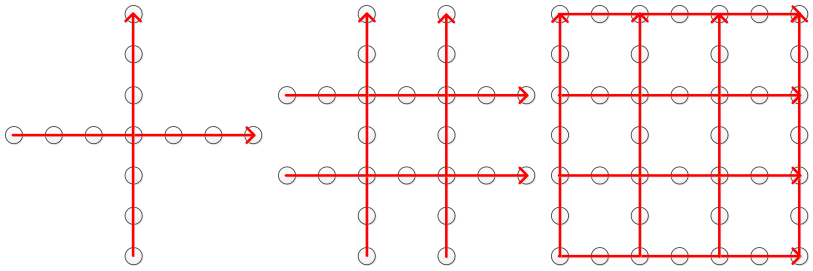

Figure 12: Simulated network topology with 2, 4, and 8 simetric flows.

Table 4: Performance of standard PSM and MH-PSM for different numbers of flows. The transmission range is 50 m – frames are forwarded over six hops.

Num. flows	Delay (ms)		Doze time (%)		PDR (%)	
	PSM	MH-PSM	PSM	MH-PSM	PSM	MH-PSM
1	1047	99	14	26+33	99	100
2	1048	171	14	18+28	83	93
4	1055	253	14	15+28	81	90
8	1063	285	12	14+20	77	86

63%. The additional hold-up in intersecting nodes does not affect the frame delay in standard PSM so significantly because most frames are anyway forwarded only one hop per beacon interval.

5. Experimental Results

MH-PSM is software-implementable: The parsing and generation of ATIM frames are not time-critical operations that have to be implemented on-chip. This enables the integration of MH-PSM into an 802.11 device driver without modifications of the lower-level MAC operations. Our experimental MH-PSM implementation is described in the following.

5.1. MH-PSM Implementation

We implemented the proposed MH-PSM as a part of our WLAN driver module for Contiki [9], an open source operating system for the Internet of Things. The used hardware platform consists of an Arduino Due board (Cortex-M3 MCU, 96 KB of SRAM) connected via USB interface to an 802.11n transceiver based on Atheros AR9001U-2NX chipset [18], as shown in Fig. 13. The AR9001U-2NX chipset contains an AR9170 MAC/baseband and an AR9104 (dual-band 2×2) radio chip. Atheros has released the firmware of AR9170 as open source, which enables us to write the Contiki WLAN driver. The open source firmware provides a direct access to the lower-MAC program that runs on the AR9170 chip, which greatly simplifies driver debugging. The used Contiki driver is partially based on the otus driver [19], a depreciated Linux driver for Atheros devices (replaced by the carl9170 driver as of kernel version 2.6). It is fully integrated with the Contiki's uIP protocol stack for TCP/IP and supports standard PSM and MH-PSM in ad hoc mode. Our goal was not only to validate MH-PSM, but to build a flexible open-source platform for experimentation with 802.11 MAC Layer Management Entity (MLME) algorithms (power saving, beaconing, time synchronization, scanning, association, authentication) for future IoT-ready WLAN-enabled embedded devices.[3] While there is abundance of

[3]Lower-MAC functions with strict timing constraints, such as DCF, are implemented on the AR9170 chip.

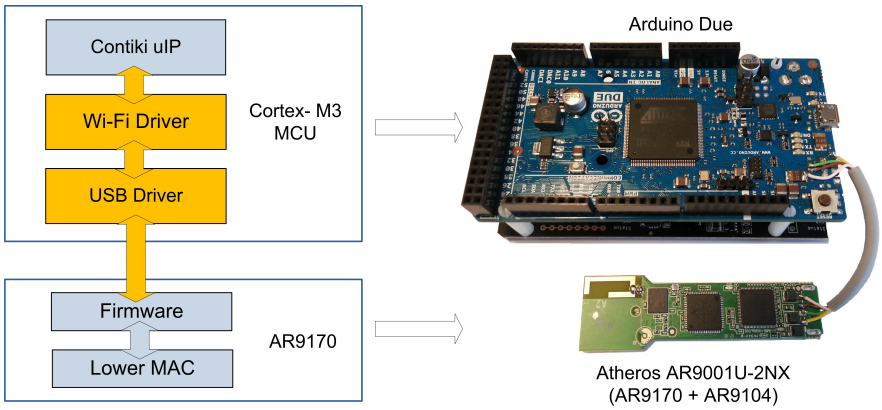

Figure 13: HW/SW platform for MH-PSM evaluation consists of an Arduino Due board and an Atheros 802.11n transceiver. We ported Contiki OS to the Arduino Due and implemented WLAN and USB drivers for the Atheros transceiver.

low-power WLAN modules for embedded devices (e.g., Roving Networks RN-131C, Gainspan GS2100M, Texas Instruments CC3000, Broadcom BCM4390, etc.) and WLAN enabled development boards for IoT applications (WiSmart EC32Lxx, RTX41xx, Spark Core, Flyport WLAN, etc.) they are not suitable for experimentation with 802.11 MAC layer management algorithms: Typically, their proprietary network stack implementations are provided as binary firmwares and only high-level communication and configuration APIs are disclosed. Moreover, their stack implementations often do not support IBSS mode and/or IPv6.

5.2. Experimental Setup

Our experimental setup consists of up to seven WLAN nodes lined up 10 m apart from each other[4] in a quiet alley at the back of an office building compound, as shown in Fig. 14. The experiments were run overnight, when the interference from the neighboring access points was negligible: WLAN spectrum monitors picked up only control and management traffic from other networks at those hours. The transmit power of the nodes was reduced to the minimum of 0 dBm, which resulted in the transmission range of roughly 30 m. Therefore, nodes that were up to three hops away could still observe and decode each others transmissions. Hence, there were fewer hidden terminals in the experimental setup than in the simulation setup. The goal was not to exactly replicate the simulation results, but to compare MH-PSM and standard PSM in the described experimental setup.

[4]We could not place the nodes further apart due to space constraints.

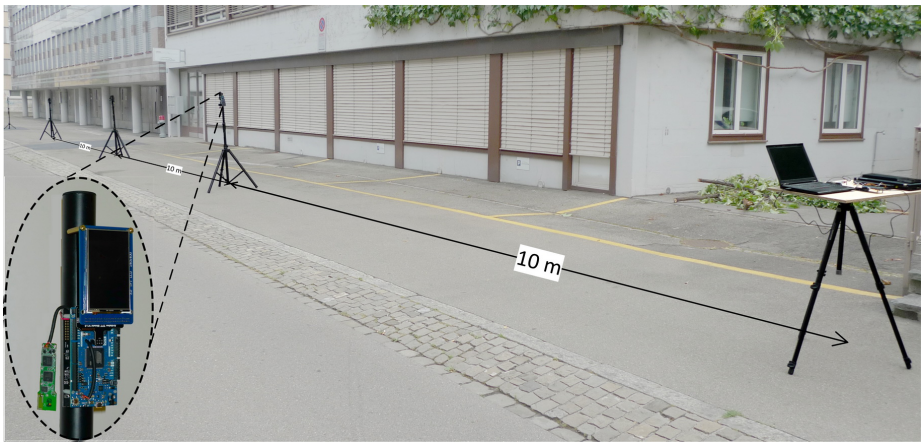

Figure 14: Experimental setup: Seven WLAN nodes lined up 10 m apart from each other. The experiments were run overnight to minimize the interference from surrounding access points. An LCD screen is connected to each node to monitor the status.

5.3. Performance Results

We first consider scenarios with two, three, and six hops that are created by placing, respectively, three, four, and seven nodes in a line. The beacon interval and ATIM window size were set to 200 ms and 20 ms. We used the same Poisson traffic generator with fixed packet sizes as in the simulations. The sender (rightmost node in Fig. 14) generated on average one frame every 200 ms ($\lambda = 5$ frames per second). For each scenario we performed multiple (typically three to five) runs with 300 packets each. The results were then averaged over those runs.

The average end-to-end delay, doze time ratio, and ATIM overhead are shown in Fig. 15. The results follow the pattern seen in the simulation (Fig. 7). For the standard PSM it takes almost N beacon intervals to forward a frame over N hops, while for the MH-PSM the delay is much shorter. The absolute delays are somewhat higher then in the simulations because packet generation and processing/parsing in intermediate nodes takes some time. Our hardware/software platform can be further optimized for this task. The doze time ratio is still significantly higher with MH-PSM than with standard PSM. The absolute values are higher than in the simulations (Fig. 7) because there are fewer hidden terminals in our experimental setup (e.g., in two-hop and three-hop scenarios, all nodes are able to hear each other). This reduces the number of beacon transmissions, hence, the number of beacon intervals in which nodes must stay awake. The gain of SoBT is lower, but still significant. The ATIM overhead of MH-PSM is comparable to that of standard PSM in the two-hop and almost 30% lower in the six-hop case, as predicted by the simulations. The absolute values are lower than in simulation because fewer ATIMs had to be retransmitted due to collisions (there were fewer hidden terminals in the experiments than in the simulations).

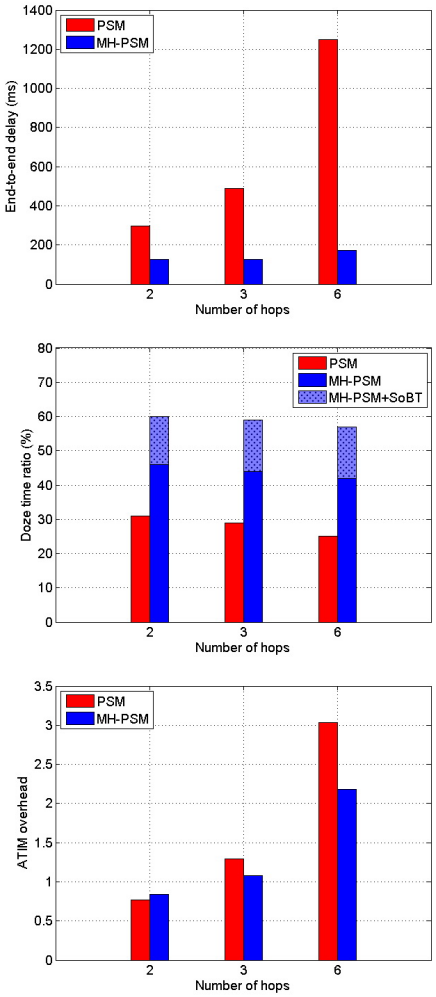

Figure 15: End-to-end delay, doze time ratio, and ATIM overhead for different number of hops.

Table 5: Performance of standard PSM and MH-PSM for different beacon intervals. Frame interarrival times is 200 ms. Frames are forwarded over six hops.

BI (ms)	Delay (ms)		Doze time (%)		ATIM overhead	
	PSM	MH-PSM	PSM	MH-PSM	PSM	MH-PSM
100	586	125	40	52+17	4.11	2.78
200	1249	173	25	42+15	3.03	2.18
400	2234	281	19	41+15	2.12	1.23

We also measured the performance of PSM and MH-PSM for different beacon intervals. The results presented so far assume the beacon interval of 200 ms. We changed the interval to 100 ms and 400 ms and repeated the measurements for the six-hop scenario. The average frame interarrival time is 200 ms regardless of the beacon interval. The results are summarized in Table 5. All our observations based on the simulation results (Table 5 apply to measurement results too: While the frame delay increases linearly with the beacon interval for the standard PSM, it increases moderately and remains comparably short for the MH-PSM. Even without SoBT, MH-PSM outperforms standard PSM in terms of the doze time ratio. As expected, the doze time decreases with the length of beacon intervals because fewer intervals are idle. The improvement in terms of ATIM overhead is also significant.

Finally, we varied the average frame Interarrival Time (IAT) $1/\lambda$ to evaluate the impact of traffic load in the six-hop scenario. The beacon interval is 200 ms regardless of the IAT. The results are summarized in Table 6. With MH-PSM, close to 80% of frames are transmitted end-to-end within a single beacon interval regardless of the traffic load (0% with standard PSM). The slight increase in the delay for longer IATs is due to the initial hold-up at the sender, as discussed in Section 4.2. As expected, the doze time increases when the traffic load decreases (i.e., for longer IATs). There is a mismatch between experimental and simulation results (in terms of absolute values, due to different network setups), but trends, relative performance, and conclusions are the same: MH-PSM significantly outperforms standard PSM in all scenarios considered in this paper.

Table 6: Performance of standard PSM and MH-PSM for different frame interarrival times. Beacon interval is 200 ms. Frames are forwarded over six hops.

IAT (ms)	Delay (ms)		Doze time (%)		ATIM overhead	
	PSM	MH-PSM	PSM	MH-PSM	PSM	MH-PSM
100	1226	166	17	37+14	1.89	1.32
200	1249	173	25	42+15	3.03	2.18
400	1285	185	40	52+17	4.11	3.03

6. Related Work

The IEEE 802.11ah proposal [20] defines a low power medium access method that optimizes standard 802.11 PSM for battery-powered devices used in smart metering and machine-to-machine communication. However, the optimization focuses on BSS (infrastructure) networks where PSM-enabled stations communicate with an access point. A number of solutions have been proposed to optimize the PSM in IBSS (ad-hoc) networks. Some of them focus on minimizing the duration of idle listening by introducing mechanisms for early transition to the doze state [6, 7, 8]. For example, in [6], the explicit announcement of the number of pending frames in ATIMs is proposed in order to allow the receiving station to move to the doze state after it receives the last frame, instead of waiting for the end of the beacon interval. In [21], the authors propose a scheme where ATIMs contain information about the nature of the intended traffic, so stations can differentiate between broadcast and multi-cast traffic; in the later case they can immediately transit to doze state if they are not members of the multicast group. In various approaches, the early transition to the doze state is combined with the dynamic adjustment of the ATIM window duration, depending on the traffic conditions in the IBSS [22]. In [7] the authors propose an algorithm for a station to dynamically adjust the remaining ATIM window duration as a response to ATIM receptions in order to transit to sleep earlier in case of low network traffic. To further decrease the energy used for idle listening, [23] proposes a scheme where transmitting stations announce their intention of sending ATIM frames in a short time period at the beginning of the beacon interval. Stations that do not send or receive any announcements do not have to stay awake for the entire ATIM window. Considering a similar low-traffic scenario, [24] proposes a scheme where the absence of traffic is declared by transmitting a delayed beacon, so that stations can skip idle listening during the ATIM window. In [25, 26], the authors propose a topology-aware power-saving algorithm based on the overhearing of the ATIM frames transmitted by the neighbors. By extracting the source addresses from the received ATIM acknowledgments, a station can defer from transmitting ATIMs to stations known to remain awake after the expiration of the ATIM window. This scheme can efficiently decrease the required ATIM window size in a fully-connected IEEE 802.11 mesh network, but it is less effective in multi-hop IBSS network topologies.

The problem of energy and latency optimization in multi-hop 802.11 IBSS networks has also been addressed in several works. In [27], the authors describe a scheme that creates a wave of RTS/CTS frames to reserve radio resources along the route for latency-optimized multi-hop communication. The scheme, however, only minimizes the latency due to channel contention and not the latency caused by PSM-enabled stations on the route. In [8], the latency is reduced by organizing the IBSS network hierarchically, so that always-awake master stations create a backbone that relays the multi-hop traffic between PSM-enabled slave stations. A low-latency routing algorithm that forwards packets via stations that are known to be awake in the current beacon interval is presented in [28]. Our solution operates at the MAC layer and, therefore, it

is independent of routing. A history-based prediction mechanism by which a station infers if it needs to stay awake to forward incoming packets is described in [29]. In case of sporadic packet bursts and short-lived flows, wrong prediction may cause the station to stay awake for no reason. In [30], the authors propose a fast flooding algorithm that propagates ATIMs and allows broadcast packets to travel multiple hops in a single beacon interval. The algorithm is applicable to broadcast transmissions only.

7. Conclusions

The Internet of Things will connect not only Zigbee-enabled devices, such as industrial sensors, but also consumer electronics that typically uses Wi-Fi for network connectivity. The power saving mechanisms of the IEEE 802.11 MAC have to be optimized to enable low-cost battery-powered devices to connect to each other ad hoc, without infrastructure support. In this paper, we proposed MH-PSM, an extension of the standard IEEE 802.11 PSM that enables low-latency ad hoc communication over multiple hops. MH-PSM also increases the doze time ratio of the devices compared to the standard PSM to further extend their battery lifetime. MH-PSM is software implementable since it does not require changes to the lower MAC. It is also backward-compatible with the standard PSM, which guarantees interoperability with legacy devices. We implementing the scheme on an embedded open source platform and demonstrated its effectiveness using both simulations and experiments. We are planning to investigate the interaction of MH-PSM with upper layer protocols (i.e., RPL/loadNG routing in particular) to further optimize the IoT communication protocol stack.

8. Acknowledgement

This work was partially funded by the European Union Seventh Framework Programme (FP7-ICT/2007-2013) under grant agreement number 288879 (Calipso; see http://www.ict-calipso.eu/).

[1] IEEE standard for Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks. Specific requirements. Part 15.4: Wireless Medium Access Control and Physical Layer Specifications for Low-Rate Wireless Personal Area Networks (2006).

[2] IEEE standard for Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks. Specific requirements. Part 11: Wireless LAN Medium Access Control and Physical Layer specifications (2012).

[3] Consumer perception of electronic toys (2007).

[4] X. Zhang, K. G. Shin, E-mili: Energy-minimizing idle listening in wireless networks, Mobile Computing, IEEE Transactions on 11 (9) (2012) 1441–1454.

[5] A. Dunkels, The ContikiMAC radio duty cycling protocol, Tech. Rep. T2011:13, Swedish Institute of Computer Science (Dec. 2011).
URL `http://dunkels.com/adam/dunkels11contikimac.pdf`

[6] D.-Y. Kim, C.-H. Choi, Adaptive power management for IEEE 802.11-based ad hoc networks, in: Proc. 5th World Wireless Congress, San Francisco, USA, 2004.

[7] E.-S. Jung, N. H. Vaidya, Improving IEEE 802.11 power saving mechanism, Wireless Networks 14 (3) (2008) 375–391.

[8] S. Yongsheng, T. A. Gulliver, An energy-efficient MAC protocol for ad hoc networks, Wireless Sensor Network 1 (5) (2009) 407–416.

[9] Contiki OS, http://www.contiki-os.org/. [Apr-2013].

[10] I. Glaropoulus, S. Mangold, V. Vukadinovic, Enhanced IEEE 802.11 Power Saving for Multi-Hop Toy-to-Toy Communication, in: IEEE Internet of Things (iThings), Beijing, P.R.China, 2013.

[11] H. Woesner, J.-P. Ebert, M. Schlager, A. Wolisz, Power-saving mechanisms in emerging standards for wireless LANs: the MAC level perspective, IEEE Personal Comm. 5 (3) (1998) 40–48.

[12] T. Nartem, E. Nordmark, W. Simpson, H. Soliman, Neighbor discovery for IP version 6 (IPv6), RFC 4861 (Sept. 2007).

[13] M. Grajzer, Nd++ – an extended IPv6 Neighbor Discovery protocol for enhanced duplicate address detection to support stateless address auto-configuration in IPv6 mobile ad-hoc networks, Internet-Draft (March 2011).

[14] Ipv6 over low power wireless personal area networks, ietf working group on, http://datatracker.ietf.org/wg/6lowpan/charter/. [Apr-2013].

[15] Z. Shelby, S. Chaktabarti, E. Nordmark, Neighbor discovery optimization for Low Power and Lossy Networks (6LoWPAN), Internet-Draft (Aug. 2012).

[16] IDC's Worldwide Quarterly Mobile Phone Tracker, http://www.idc.com/tracker/showproductinfo.jsp?prod_id=37. [June-2013].

[17] S. Mangold, Jemula802, https://github.com/schmist/Jemula802. [Apr-2013].

[18] Atheros AR9001U, http://wikidevi.com/files/Atheros/specsheets/AR9001U.pdf. [Apr-2013].

[19] Otus, http://linuxwireless.org/en/users/Drivers/otus. [Apr-2013].

[20] S. Aust, R. Prasad, I. G. M. M. Niemegeers, IEEE 802.11ah: Advantages in standards and further challenges for sub 1 GHz Wi-Fi, in: Communications (ICC), 2012 IEEE International Conference on, 2012, pp. 6885–6889.

[21] A. K. Sharma, A. Gupta, A. Misra, Optimized power saving mechanism for wireless ad hoc networks, in: Proc. 1st Int. Conf. Recent Advances in Inf. Tech. (RAIT), Dhanbad, India, 2012.

[22] E.-S. Jung, N. Vaidya, An energy efficient MAC protocol for wireless LANs, in: Proc. IEEE Infocom, New York, USA, 2002.

[23] N. Rajangopalan, C. Mala, Modified power save model for better energy efficiency and reduced packet latency, American Journal of Engineering and Applied Sciences 5 (3) (2012) 237–242.

[24] J.-M. Choi, Y.-B. Ko, J.-H. Kim, Enhanced power saving scheme for IEEE 802.11 DCF based wireless networks, in: Personal Wireless Comm., Vol. 2775 of Lecture Notes in Computer Science, 2003, pp. 835–840.

[25] W. Akkari, A. Belghith, A. Ben Mnaouer, Enhancing power saving mechanisms for ad hoc networks using neighborhood information, in: Proc. Int. Wireless Comm. and Mobile Comp. Conf. (IWCMC), Crete, Greece, 2008, pp. 794–800.

[26] A. Belghith, W. Akkari, Neighborhood aware power saving mechanisms for ad hoc networks, in: Proc. IEEE Conf. Local Computer Networks (LCN), IEEE, Montreal, Canada, 2008.

[27] G. Hiertz, J. Habetha, E. Weiss, S. Mangold, A cut-through switching technology for IEEE 802.11, in: Proc. IEEE Circuits and Systems Symp. on Emerging Technologies, Shanghai, China, 2004.

[28] R.-H. Hwang, C.-Y. Wang, C.-J. Wu, G.-N. Chen, A novel efficient power-saving MAC protocol for multi-hop MANETs, Int. Journal of Comm. Systems 26 (1) (2013) 34–55.

[29] N. Rajagopalan, C. Mala, Modified power save model for better energy efficiency and reduced packet latency, Am. J. Eng. Applied Sci. 5 (3) (2012) 237–242.

[30] M.-H. Seo, H.-J. Yoon, J.-S. Ma, Fast flooding in Power Save Mode of IEEE 802.11 DCF based mobile ad hoc networks, in: NETWORKING 2004, Vol. 3042 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2004, pp. 1464–1469.