electronics

Article

An Enhancement for IEEE 802.11 STA Power Saving and Access Point Memory Management Mechanism

Vishal Bhargava [1,*] and Nallanthighal Raghava [2]

[1] Computer Engineering Department, Delhi Technological University, Delhi 11042, India
[2] Department of Electronics & Communication Engineering, Delhi Technological University, Delhi 110042, India
* Correspondence: nsraghava@gmail.com

Abstract: Wi-Fi researchers are trying hard to extend battery life by optimizing 802.11 power save. The rising number of Wi-Fi devices and IoT devices and daily demands have reduced Station (STA) device power consumption. Better memory management at the Access Point (AP) side is also needed, so that AP can store maximum data to deliver sleepy STA devices. There are three main contributions of this study. The first one focuses on a power-saving mechanism scheme with an adaptive change to Listen Interval (LI) based on the battery status of station devices. The second contribution aims to examine better memory management for the AP buffer to store packets that will in the future deliver power-saving STA when awake. The third contribution, under the implementation of the proposed method, includes Wi-Fi corner cases covered as Beacon frames missed via STA, the keep-alive factor, and the upper-layer time taken to care for and ensure the delivery of unicast/multicast/broadcast data. The proposed approach introduced 802.11 protocols to share battery status, a protocol to announce proposed features via AP, and a protocol to change LI at runtime. Simulation results show that the proposed scheme performs better than 802.11 power saving in terms of power usage at the STA and access point memory management.

Keywords: 802.11 protocol; access point; buffer management; constrained device; internet of things; power save; Wi-Fi

Citation: Bhargava, V.; Raghava, N. An Enhancement for IEEE 802.11 STA Power Saving and Access Point Memory Management Mechanism. *Electronics* **2022**, *11*, 3914. https://doi.org/10.3390/electronics11233914

Academic Editors: Juan-Carlos Cano and Nurul I. Sarkar

Received: 5 October 2022
Accepted: 14 November 2022
Published: 26 November 2022

1. Introduction

With the increased demand for Wi-Fi in today's scenario [1] and the increased number of smartphones/IoT devices, devices have become very power-savvy [2]. For Wi-Fi devices, it is essential to maintain power consumption while maintaining connections with access points. When battery downfall happens, the user experience may deteriorate when Wi-Fi becomes disconnected [3] or the early battery finishes. Wireless Local Area Network (WLAN), or Wi-Fi, was created to connect mobile devices such as laptops to the network wirelessly. Power save management was defined as a critical component in the early specifications for the efficient use of such mobile devices. It was even stated in the IEEE 802.11-1999 specification, which only allowed for a 2 Mbps connection rate. Up to that point, portable batteries, including dry and button batteries but, excluding huge rechargeable laptop batteries, were not widely used. As a result, power-saving management in systems where battery power is a concern is still in its infancy, and engineers confront numerous challenges in implementing it. To facilitate reading, the frequently used acronyms and notations are summarized in Table 1.

The IEEE 802.11 specification introduced a WLAN power saver to conserve energy [4]. It saves power by switching from active mode to power save mode. IEEE 802.11 standards have introduced many power-saving mechanisms, allowing Wi-Fi devices to reduce their power consumption. Access Point (AP) is generally connected via an external power device that cannot go into Power Save (PS) mode. However, a battery-operated Station (STA) needs to go into PS mode to increase the battery life and allow for more extended performance. Section 3 discusses the power-save mechanism in more detail. This battery life extension can be significant for low-powered devices such as smartphones, Wi-Fi sensors, or IoT devices. Access Point buffer management [5] and Scheduling of Sleep/Wake [6] stations is also performed in 802.11 power saving. The STA device needs to send a frame periodically to AP to remain connected. If no significant traffic goes between them, STA sends a keep-alive packet periodically, so AP knows STA is there and will not kick off a particular STA from its connected STA list. The layer above the data link layer and its protocol also cause link failure [7], so the keep-alive timer decides that the upper layer timeout should not happen. The Listen Interval (LI) plays an important role when the device goes into power-save and how much memory buffer is assigned via AP for the station. However, these parameters are static and do not solve the need of the hour. The LI has been configured during the association phase, and its value remains unchanged until the station does not disconnect. In practice, STA put the LI value as 1, in order to wake up on every beacon, and AP assigned static memory to the connected STA. In [8,9], the authors try to change the LI according to the load on AP, unfortunately without prioritizing STA, so no practical framework architecture is covered. In our paper, more simple and functional designs are preferred to consider IoT device use cases. IoT devices transfer less data but always want to connect with low power spent. Using the slightly modified framework, we can make it fully compatible with the existing 802.11 standards devices.

Table 1. Acronyms and notations.

Acronym and Notations	Definition
AP	Access Point
STA	Station
LI	Listen Interval
PM	Power Management
PS	Power Save
PSM	Power Saving Mode
QoS	Quality of Service
NAV	Network Allocation Vector
VHT	Very High Throughput
TXOP	Transmission Opportunity
TWT	Target wake time
OS	Operating System

This paper takes a more practical approach to solve the above challenges; the presented work aims to reduce embedded Wi-Fi device power consumption and improve access point memory management. The contributions of this paper are listed as follows:

1. Adaptive Listen Interval: An adaptive listen-interval-based buffer management scheme for the Wi-Fi device has been proposed. In the proposed architecture, dynamic LI used is based on the STA device's battery status and on the basis of the defined policy under architecture. Regarding the base of dynamic LI, AP allocates buffers for the STA device. For this purpose, in this paper, we have introduced 802.11 protocols and policies to accept LI adaption requests. The keep-alive factor is considered under implementation.
2. Proposed LI change, protocol implementation, and activity table updates are software-driven, so updates at the existing STA and AP framework are covered under the proposed architecture.
3. We conducted an extensive performance evaluation. The results of STA power consumption have been compared with standard 802.11 PSM in different situations. AP buffer memory improvement over conventional static AP memory results is also discussed in the performance evaluation section.

The rest of the paper is organized as follows: Section 2 reviews relevant work/methods related to wireless power-save, software-based Wi-Fi device changes, and discusses amendment architectures. Section 3 focuses on IEEE power save standards. Section 4 discusses the design and architecture framework of the proposed methods, STA and AP side. The proposed protocol changes and their algorithm implementation are discussed in Sections 5 and 6, respectively. Results and discussion are presented in Section 7. At last, Section 8 concludes the proposed work.

2. Related Work

Several researchers work in the Wi-Fi domain, and they are multi-dimensional. There have been many articles welcoming/appreciating the new methods to improve the 802.11 Power Saving Mode (PSM) standard, but power is something which always has scope for improvement. In Wireless Sensor Networks (WSNs), the connections between sensor nodes are closely related to their wireless transmission power [10]. The author in [11] revealed power-save use cases and applications in different practicalities. IEEE Wi-Fi standards try to improve the Wi-Fi power-saving mechanism. With standards, hardware and software are continuously evolving to save power. Furthermore, 802.11ax standard introduces a Target Wake Time (TWT) mechanism to reduce power consumption, and later, specific to IoT devices' power saving, IEEE introduced the 802.11ba standard. The 802.11ba [12] Wake-Up Radio (WUR) mode is a low-power operating mode where only the Rx chain is active, but it needs special hardware, which always awakens even in a sleepy state. Both 802.11ba [13] and 802.11ax [14] power savings suffer from the clock drift effect, which significantly degrades their performance in the case of low traffic. Sampleless Wi-Fi [15] creates new constellation diversity via changes done under the receiver part of the Radio Frequency (RF) chain. Hardware changes are costlier, and it is tough to change old devices with the latest hardware.

Researchers use a software-driven approach for wireless solutions [16]. Jose et al. defined a software-based framework for wireless devices [17]. The proposed solution can work on access points, and AP can provide innovative functionalities such as load balancing and Quality of Service (QoS) facility. The problem with implementation is that no real access points software has been taken to reflect the changes discussed in the paper, and backward compatibility is not considered at all. The article by [18] also uses the RTS/CTS frame with the existing power save mode. Authors claim that STAs can also sleep during the Network Allocation Vector (NAV) period. However, the presented approach is only applicable for low data traffic and RTS/CTS frame addition, and an extra overhead of frames in the network, which a part of is not considered under the paper when authors conduct performance evaluations with existing methods. Saeed and Kolberg presented a context-aware listen interval in a more software-driven way, using a machine learning approach [19]. The paper shows the experimental result, revealing that the presented way can save 75% energy compared to the 802.11 standard power-saving mechanisms. However, the presented way is more theoretical in comparison to practical implementation. There is no protocol implementation discussed in the paper.

Tae et al. [20] propose an intelligent reinforcement learning-based power-saving mechanism. The discussed machine learning algorithm dynamically changes the listen intervals of stations, which helps optimize STA's energy consumption. This paper efficiently manages the trade-off between energy consumption and transmission delay. Unfortunately, the article missed the 802.11 implementation phase, AP side changes, and input data consideration. In another paper, the author proposed an energy-conserving model [21] for Wi-Fi STAs competing for power save mode and connected with the same access point. A Harmonious Power Saving Mechanism (HPSM) handles PSM traffic contention problems, and STA prioritizes according to link resource consumption. According to the model, higher priority STA has more chance to deal with AP, and STA does not waste energy to obtain the environment under Carrier Sense Multiple Access/Collision Avoidance (CSMA/CA). Again, the presented model does not cover practical details, and QoS and protocol part are not considered under the implementation phase.

The authors in [22] propose a Named Data Networking (NDN)-enabled PSM model for Wi-Fi devices, where STA can predict data arrival precisely and go into light or deep sleep mode accordingly. This method is based on a cost function that considers priority and packet latency and can reduce idle listening time. Simulations show methods effective in improving average power consumption and average transmission delay. Two power modes are discussed here, light and deep sleep modes. However, the paper does not reveal much difference between them, and the Which RF chain part is close under which mode. One drawback of the method is that STA can only predict unicast packets. Still, the implementation phase does not consider broadcast/multicast packets, which are obvious packets under the Wi-Fi domain. Some research used low-power wireless technology with Wi-Fi, whether BLU-FI or ZigBee attached power save [23]. However, the range of Wi-Fi is different from low-power technology, and these technologies cannot send high data. Receiver design [24,25] or context-sensitive framework development [26] is also a way to save energy. Power consumption is also challenging, especially for battery-operated devices [27]. The offloading of features is performed to overcome this challenge in a small manner. A study by [28] presented two algorithms specially designed for wireless ad hoc networks. One is the Select_Node algorithm to determine how many nodes want to participate in data transmission based on the battery status. The second one is the Total_Minimum_Power algorithm, which calculates the total power consumption in a wireless ad hoc network and finds the minimum energy required to transmit data from one node to another. These two algorithms help to minimize the consumption of considerable battery power and increase the overall lifetime of the network. Unfortunately, the presented method only applies to the ad-hoc mode, not to the infra mode, which is widely used in the Wi-Fi world.

Fonseca et al. [29] presented a resource management framework for 802.11 wireless AP. The results show a significant performance improvement in throughput and an improved working model under interference. The given model does not consider the power save of STAs and memory management for access point memory. The motivation for our work is to introduce a new adaptive power-save feature and a more practical solution to manage fairness in an IoT battery-operated device environment. The proposed paper covered the point with a focus on Wi-Fi protocols and every corner case covered under the work.

3. Power Management in IEEE 802.11

According to IEEE 802.11 specifications, any device that can connect with Wi-Fi AP can be called an STA device. A Wi-Fi station can be in two modes. The first one is the active mode, where a device is always awake, and AP immediately transmits frames to the client. The second mode is the PS mode, where a client is in a dozen state. In this mode, AP buffers the data destinated for the client. In the power-save mechanism, STA radio cannot transmit/receive, and the radio can go into a dozen states from an active state. How a Wi-Fi station switches between active to PS mode will be discussed. In this process, if an STA wants to move into power-save mode, it notifies the access point via any data frame, setting the Power Management (PM) bit to 1 (shown in Figure 1), and AP starts the buffering of data for STA. The PM bit is part of the frame control filed within the Medium Access Control (MAC) header of the 802.11 frames. The legacy PS-Poll or NULL data frame retrieves data from the AP side. In legacy mode, a control frame, i.e., PS-Poll frame STA, is sendt to recover every single data frame, as shown in Figure 2.

Next, how does the STA know the AP is buffering data for it? The STA utilizes the beacon interval of the access point to know whether the AP is holding any data packets. The AP indicates buffered frames using the Beacon frames' Traffic Indication Map (TIM) element. The LI plays an important role when the device goes into power save mode and how much memory buffer is assigned via AP for the station. LI negotiates during the association phase, and STA lets AP know how many beacons STA will go into power-save mode. LI is communicated in units of the beacon interval. For example, in (Figure 3), for the association request frame, the listen interval is set to 1 by the STA. If STA never wants to go into power-save mode, the LI value is set at 0.

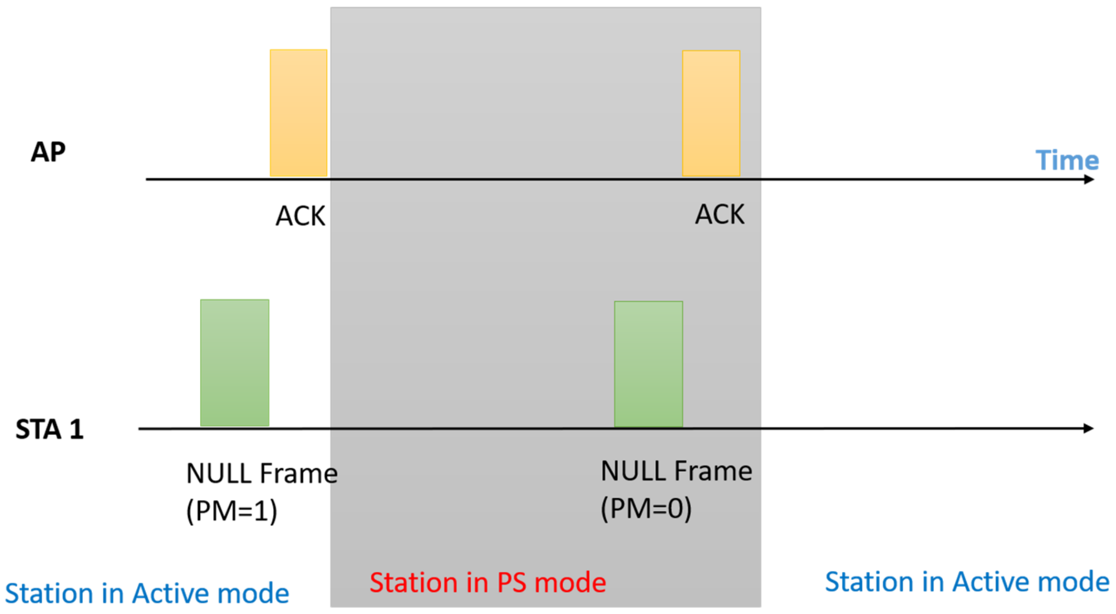
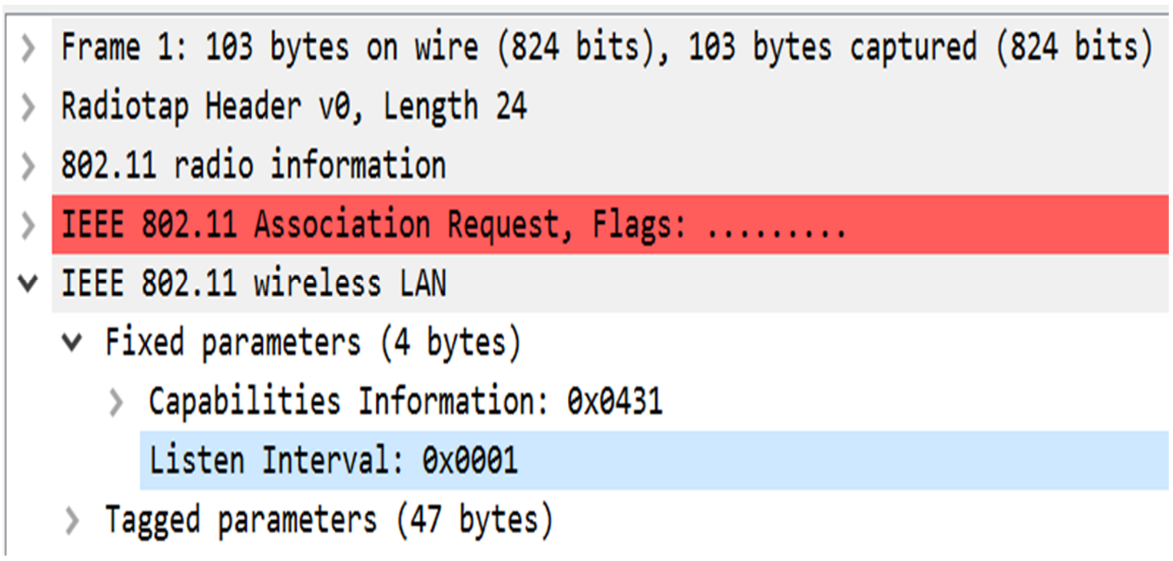


Figure 1. Power management transition.



```
> Frame 1: 103 bytes on wire (824 bits), 103 bytes captured (824 bits)
> Radiotap Header v0, Length 24
> 802.11 radio information
> IEEE 802.11 Association Request, Flags: .........
v IEEE 802.11 wireless LAN
  v Fixed parameters (4 bytes)
    > Capabilities Information: 0x0431
      Listen Interval: 0x0001
  > Tagged parameters (47 bytes)
```

Figure 2. “Listen Interval” under the Association Request frame.

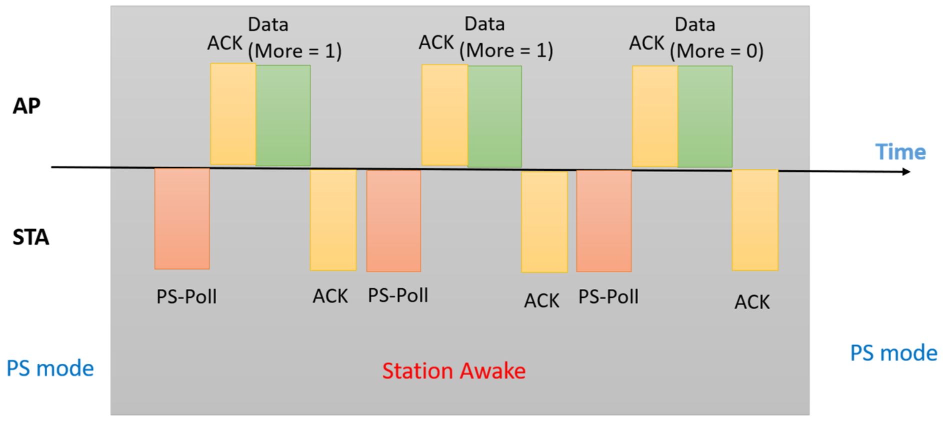


Figure 3. Legacy power management.

An AP may use the Listen Interval information to determine the lifetime of frames it buffers for an STA. However, there is no way to change LI during the connection. STA needs to disconnect if it wants to connect with AP with another LI value. AP also defines MAX_LISTEN_INTERVAL, the maximum value of the listen interval that AP supports. The device becomes inactive (in power save mode) before accessing the regular frames from Access Point. The station sleeps for that listen interval, and Access Point accumulates the packets until STA does not wake. The energy is also wasted when STA awakes in every Delivery Traffic Indication Message (DTIM) Beacon, even when no packet is received. A long LI allows STA to sleep for a long time, which causes more memory on the AP side. This paper tries to optimize the AP buffer to overcome this problem and change LI on runtime according to the requirement. This paper proposes a novel approach to managing the power-saving scheme for low-battery station devices. Moreover, while looking for power-saving ways in this research, the proposed method will also cover the optimization of AP buffer for STAs operating in power-save mode and keep-alive time management. Before connecting to an infrastructure WLAN, an STA must associate with an AP of the WLAN by sending an Association Request (AR). When STA wants to go into power save mode, it sets its PM bit in its MAC header to let AP know it is active no more. In this case, Access Point will buffer all the information for the client device.

Furthermore, 802.11e specification introduced Unscheduled Automatic Power-Save Delivery (UAPSD), which replaced legacy PS-Poll frames with trigger frames (Figure 4). When STA lives in PS mode, it sends a trigger frame to retrieve all frames from AP and again goes into PS mode. In addition, 802.11ac specification introduced Very High Throughput (VHT) Transmission Opportunity (TXOP) power save; in this mechanism, when any STA finds another STA with a TXOP, it will put the radio into low power. With the 802.11ax (Wi-Fi 6) standard, target wake time (TWT) was introduced, in which AP manages activity in the network. AP manages the overall network to reduce contention between STAs and STA awake time [30]. Furthermore, 802.11ax supported STA negotiating TWT time with TWT capable AP, and according to that, they can go into PS mode (Figure 5). The TWT mechanism was implemented and revealed through simulation via the authors in [31], which show that the TWT reduces the sensor's power consumption. The article by [32] discusses TWT's efficiency and problem. TWT and other IEEE 802.11 mechanisms do not set any STA's priority bases on power save. No mechanism can tell how AP will make decisions based on STA power requirements. The presented work in this paper takes a more practical approach and gives weight to the STA power condition under the power saving mechanism.

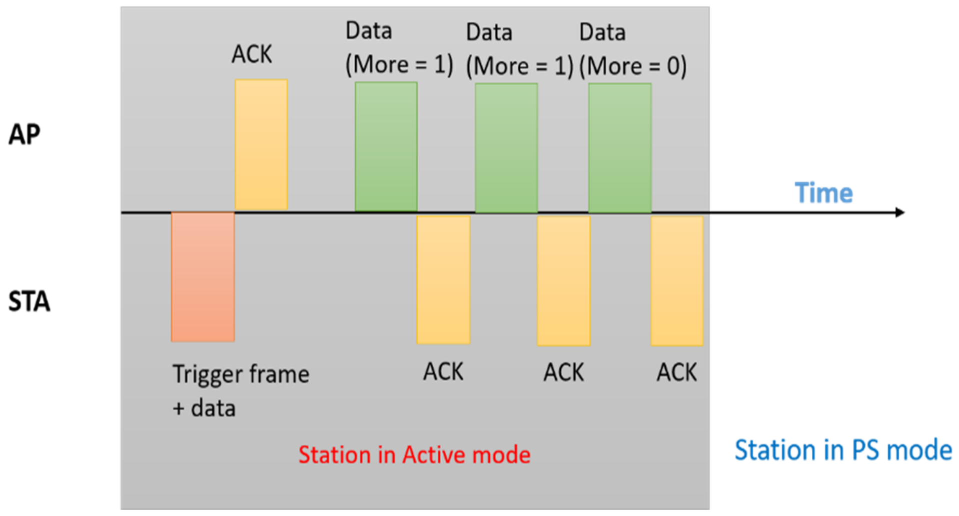


Figure 4. 802.11e PS Frame Exchange.

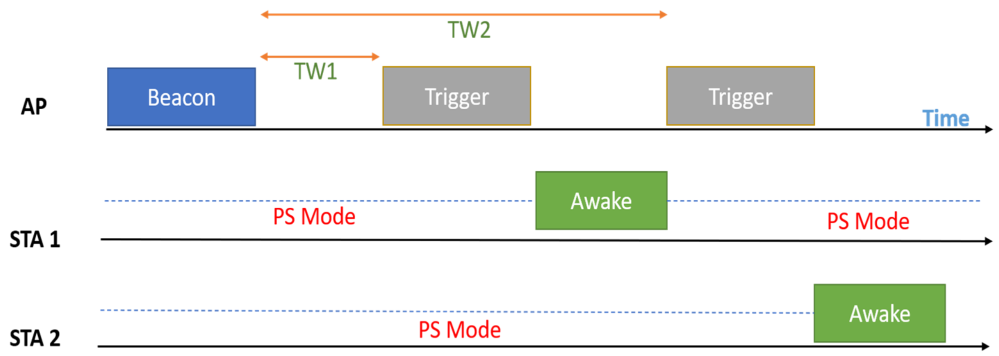


Figure 5. 802.11ax TWT Power Save Mechanism.

4. Design and Architecture of the Proposed Framework

The problem can be cast as a framework/protocol limitation aiming to maximize station battery usage and AP memory under Wi-Fi use cases. Moreover, we intend to propose, without requiring modifications, the standard IEEE 802.11 end devices hardware. We suggest software changes on both sides (STA and AP) of the Wi-Fi device. The first subsection describes access point changes, and the second discusses station side changes. Moreover, these changes are entirely software-driven and do not need any hardware modification. To implement this feature, a user needs to change the Wi-Fi driver and firmware of the device and deploy it on the device side.

4.1. AP Framework Description

This framework can classify frames based on the frame type and subtype. Different elements play their role according to the requirement. An AP can have many parts, but we discussed and showed the aspects according to our proposed method. This proposal aims to improve battery consumption, especially when the battery dies. It adaptively and dynamically controls AP performance concerning the AP's buffer and keep-alive time network characteristics.

Figure 6 shows the proposed framework for the access point. With the proposed method, AP also supports legacy STA devices, which means STA can connect to this AP, even though it does not support dynamic LI. A particular bit is introduced in the beacon to give information about this feature, whether it is supported via AP. Our proposal allows the access point to evaluate whether the clients have the right to obtain more buffers. AP framework elements are discussed as follows:

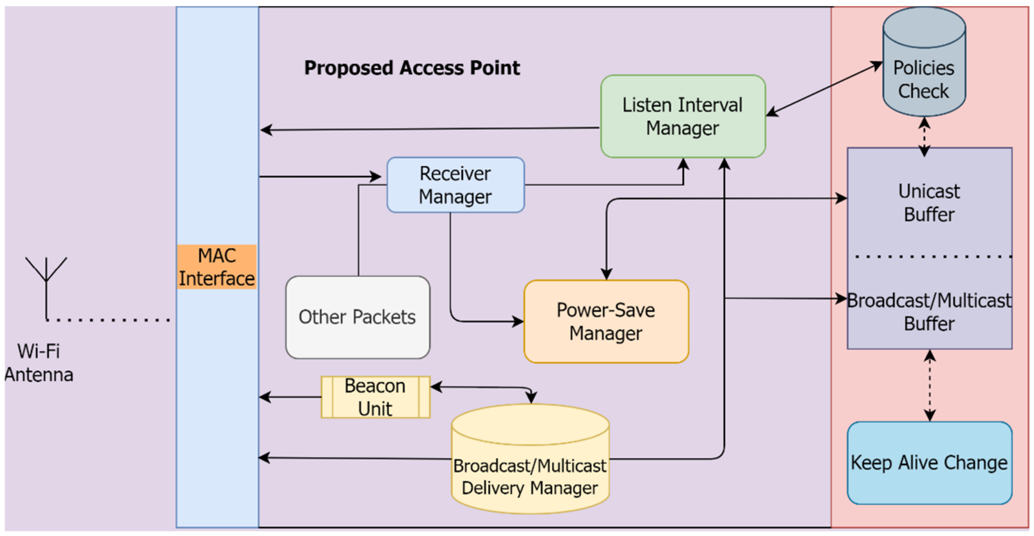


Figure 6. Proposed access point Framework.

Receiver Manager: The receiver-manager handles all the good packets received via AP. For example, if the frame is PM = 1, it will redirect to the power-save manager, and if the frame is related to changing into LI, it will activate the Listen Interval Manager. Other packets go on the default path.

Listen Interval Manager: This module decided whether AP should accept a change of LI request from STA. With consideration of policies, accept or reject request decisions are taken.

The Listen Interval Manager considers the following policies in the decision:

1. LI should be like this; all STAs should awake for Broadcast (BC)/Multicast (MC) frame notification.
2. MAX_LISTEN_INTERVAL should be defined in a way; Transmission Control Protocol (TCP) timeout or upper-layer timeout should not happen.
3. The requested LI should not be greater than MAX_LISTEN_INTERVAL.
4. The memory buffer should be enough to increase LI.

Power-save Manager: This module handles power-save management requests and stores upcoming frames to buffer.

Beacon Unit: Beaconing with supported feature enabled is performed via this unit.

Broadcast/Multicast Delivery Manager: This module sets DTIM into a beacon to deliver a BC/MC packet to the station. It deals with broadcast/multicast buffers, and whether a BC/MC packet needs to buffer or from the buffer when it needs to deliver. Here, the Broadcast/Multicast Delivery Manager makes sure at any point, and STA should wake together, so that BC/MC packets can provide to them. Dynamically, DTIM values change according to the LI values of STA and the buffer.

4.2. STA Framework Description

Figure 7 shows the proposed changes for the station. The "battery status utility" application is created on the Operating System (OS) level. This utility is responsible for reading the battery status, triggering the Wi-Fi connection, and triggering the power save operation, apart from the application layer proposed framework same as the standard Linux OS wireless device framework. The management packet (e.g., Association frame) is handled via wpa_supplicant and passed through to the driver. In contrast, data packets (power save packets) go through a standard netlink Socket (SOCK) to the device.

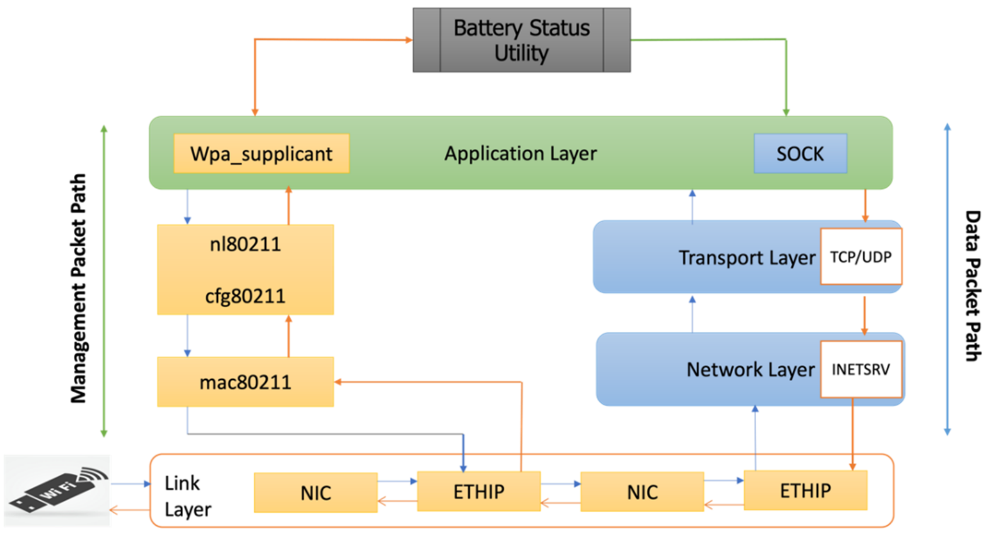


Figure 7. Proposed Station Framework.

The OS tool is used to read battery status. For example, the upower command-line tool extracts information related to the power source (batteries) on Linux. Example command:

upower -I $(upower -e | grep BAT) | grep—color = never -E “state|to\ full|to\ empty|percentag”

Furthermore, 802.11 has three different types of packets: Management, control, and data packets. Association request (a management packet) and other data packets are in the interest of this paper. In Figure 7, the management path (management packet flow) and data paths (data packet flow) are shown.

Management path: When the connection is triggered, battery status is provided to wpa_supplicant via battery status utility. Messages sent via nl80211 are in the kernel handled in the cfg80211 module and finally reach the device, as shown in Figure 7.

Data path: Power-save related and low battery data received from the utility are passed to the relevant socket, through the TCP/IP stack, then passed to the server, which implements the IP protocol (INTETSRV), and finally reaches the device. The device transfers packets to the intended access point.

5. Protocol Implementation

Protocol implementation aims to demonstrate the proposed method’s effectiveness and fit it with the 802.11 standards. Two Atheros ar9271 USB boards are taken to experiment, develop, and validate the presented protocols. The existing source code [33] needs to be modified according to the proposed protocol. One USB board acts as STA, and the other acts as an AP. The lab setup is shown in Figure 8.

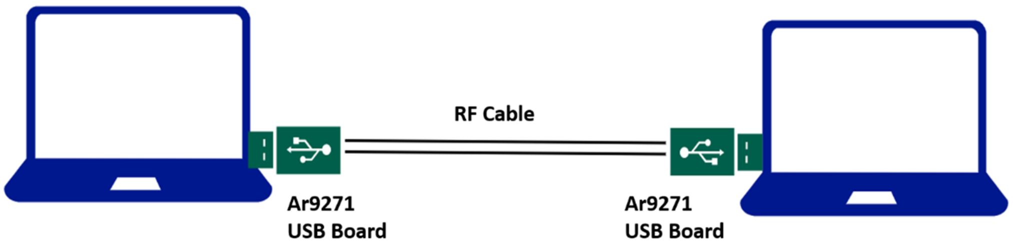


Figure 8. Lab Setup.

Hostapd is used to configure AP, and wpa_supplicant is used to configure STA. For example, the following command is used to up AP and install hostapd:

```
$ apt-get update
$ apt-get install firmware-atheros
$ apt-get install iw
$ apt-get install hostapd
$ hostapd ~/hostapd.conf//To configure AP
$ apt-get install isc-dhcp-server
```

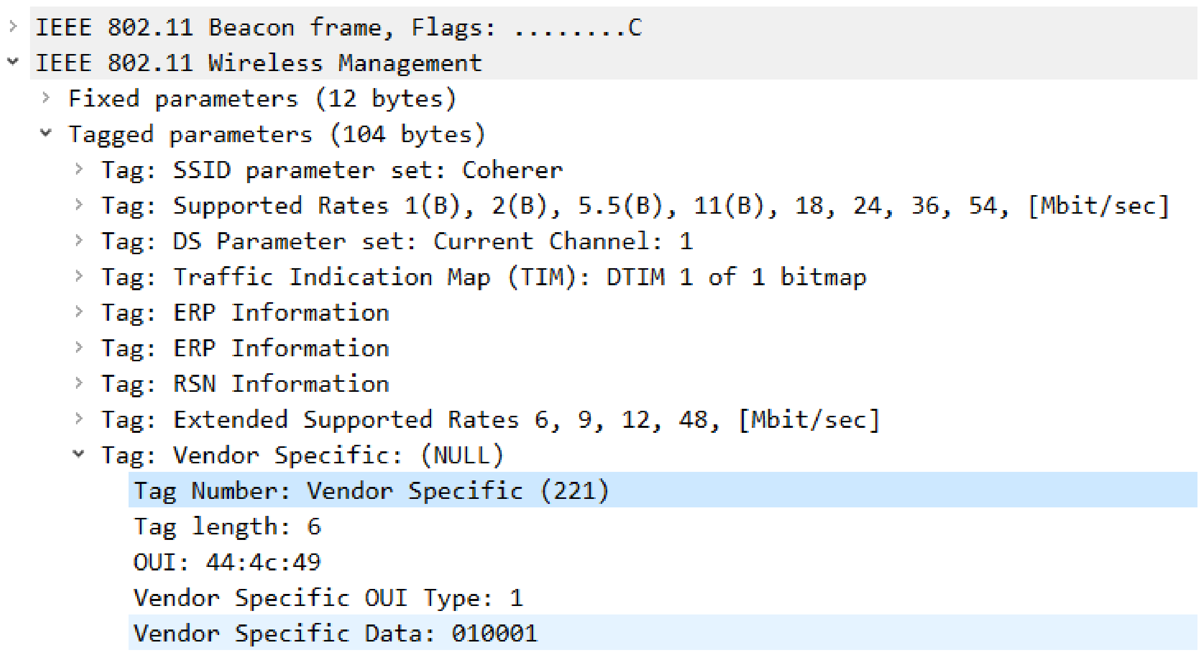
And station can connect to AP (SSID—LI_AP) with the following command:

```
$ wpa_passphrase LI_R_AP >>/etc/wpa_supplicant/wlan0_sta.conf
```

In the 802.11 specifications, a vendor-specific element can be used for specific information defined for a particular purpose. The vendor element will be used here to customize the packet. Mainly four frames have been amended or introduced to implement the proposed feature at the STA and AP sides.

First is the Beacon frame amendment on the AP side. A vendor element (Information Element 0xDD) has been added under the beacon frame, showing AP supports the proposed dynamic LI feature.

The sniffer snapshot (Figure 9) clearly shows that the vendor-specific data is 0x1, which means the proposed dynamic LI feature is supported via AP. AP does not support this feature if this element is not present or the information is 0x0.

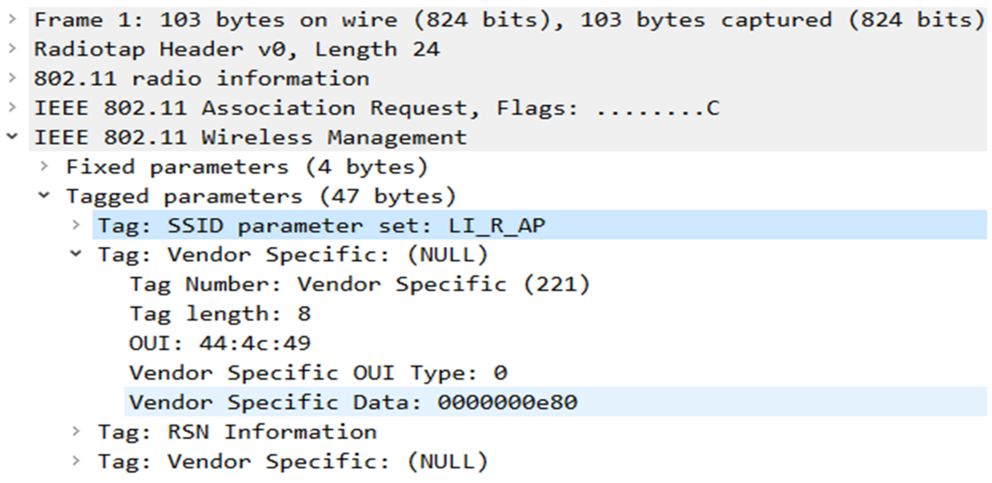


```
> IEEE 802.11 Beacon frame, Flags: ........C
v IEEE 802.11 Wireless Management
  > Fixed parameters (12 bytes)
  v Tagged parameters (104 bytes)
    > Tag: SSID parameter set: Coherer
    > Tag: Supported Rates 1(B), 2(B), 5.5(B), 11(B), 18, 24, 36, 54, [Mbit/sec]
    > Tag: DS Parameter set: Current Channel: 1
    > Tag: Traffic Indication Map (TIM): DTIM 1 of 1 bitmap
    > Tag: ERP Information
    > Tag: ERP Information
    > Tag: RSN Information
    > Tag: Extended Supported Rates 6, 9, 12, 48, [Mbit/sec]
    v Tag: Vendor Specific: (NULL)
        Tag Number: Vendor Specific (221)
        Tag length: 6
        OUI: 44:4c:49
        Vendor Specific OUI Type: 1
        Vendor Specific Data: 010001
```

Figure 9. Beacon sniffer capture with LI feature-enable.

The second is Association Request on the STA side. In the association request, STA sent LI. If STA never wants to go into power-save mode, value 0 is used by STA. If an STA connects with LI 0, this feature is disabled via AP internally for that STA. Battery-operated devices should not choose 0 as the default value. The LI value is expressed in units of beacon intervals. AP uses this LI information in determining the lifetime of frames that it buffers for an STA.

Here a vendor-specific element (ID = 0xDD) is used to send battery status information to AP under the association request frame (Figure 10). With this information, STA announced to AP "it supports the proposed dynamic LI feature," and AP can predict STA's future requirements. If the station has enough battery at the time of connection and after connection, "Is STA taking unfair advantage via showing low battery?" is shown.



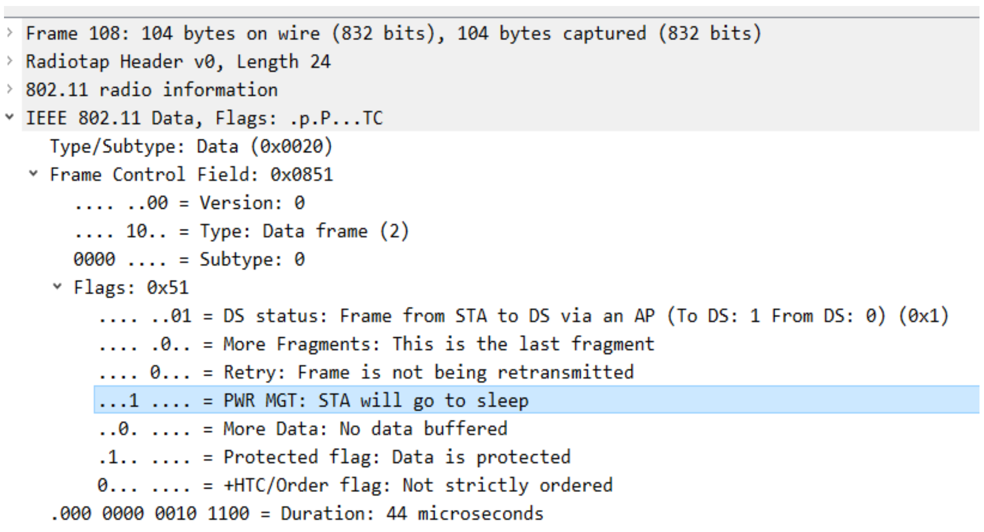
```
> Frame 1: 103 bytes on wire (824 bits), 103 bytes captured (824 bits)
> Radiotap Header v0, Length 24
> 802.11 radio information
> IEEE 802.11 Association Request, Flags: ........C
v IEEE 802.11 Wireless Management
  > Fixed parameters (4 bytes)
  v Tagged parameters (47 bytes)
    > Tag: SSID parameter set: LI_R_AP
    v Tag: Vendor Specific: (NULL)
        Tag Number: Vendor Specific (221)
        Tag length: 8
        OUI: 44:4c:49
        Vendor Specific OUI Type: 0
        Vendor Specific Data: 0000000e80
    > Tag: RSN Information
    > Tag: Vendor Specific: (NULL)
```

Figure 10. Association request sniffer capture with battery status.

Formula to send battery status under association request = (total battery * battery percentage)/100

In Figure 10, STA announces it supports the LI feature, and its current battery is 3712 mAh (0x0E80).

The third packet is a new data packet (Figure 11) to indicate the power save status. An STA can go into sleep mode via a set PM bit into any data packet. Usually, a NULL data frame is used, indicating that power management is enabled to the AP. A particular introduced data packet is used via STA in the proposed mechanism to indicate PSM mode. When STA goes into PSM mode, it constantly updates its battery status to AP. This frame also helps AP to precisely buffer its management algorithm.



```
› Frame 108: 104 bytes on wire (832 bits), 104 bytes captured (832 bits)
› Radiotap Header v0, Length 24
› 802.11 radio information
˅ IEEE 802.11 Data, Flags: .p.P...TC
    Type/Subtype: Data (0x0020)
  ˅ Frame Control Field: 0x0851
      .... ..00 = Version: 0
      .... 10.. = Type: Data frame (2)
      0000 .... = Subtype: 0
    ˅ Flags: 0x51
        .... ..01 = DS status: Frame from STA to DS via an AP (To DS: 1 From DS: 0) (0x1)
        .... .0.. = More Fragments: This is the last fragment
        .... 0... = Retry: Frame is not being retransmitted
        ...1 .... = PWR MGT: STA will go to sleep
        ..0. .... = More Data: No data buffered
        .1.. .... = Protected flag: Data is protected
        0... .... = +HTC/Order flag: Not strictly ordered
    .000 0000 0010 1100 = Duration: 44 microseconds
```

Figure 11. Data packet with PM enables sniffer capture with battery status.

The fourth is data packets that support dynamic listen interval changes (Figure 11). A particular data packet is sent via STA to change LI according to its battery status. As shown in Figure 7, the battery status utility continuously monitors the battery's health. This packet is sent via STA to change the LI value when it is going down from the threshold value (discussed in the algorithm implementation section). It depends on whether AP it accepts or rejects the change to the LI request. If rejected, AP can suggest a new value. If it agrees to the LI change request, it does not need to send the last two frames (Figure 12). These changes have been done on both sides (STA and AP). Rejection can be based on the policies defined under Section 4.1.

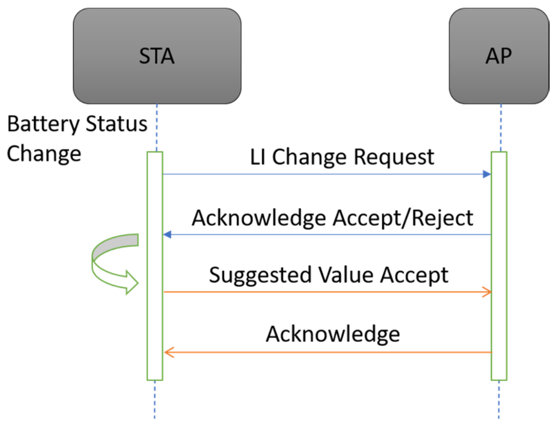


Figure 12. LI Change Sequence Diagram.

STA can change the LI value again (to the previous value) using the same packet if the STA device charges and the battery level increases.

6. Algorithm Implementation

This section presents the data structure and mechanism used to implement the discussed STA and AP functionality. The data delivery flow chart is shown in Figure 13. The framework of our proposed solution for STA is shown in Algorithm 1 (dynamic battery changes algorithm), which contains several procedures to implement functionality discussed in the framework and protocol implementation phase.

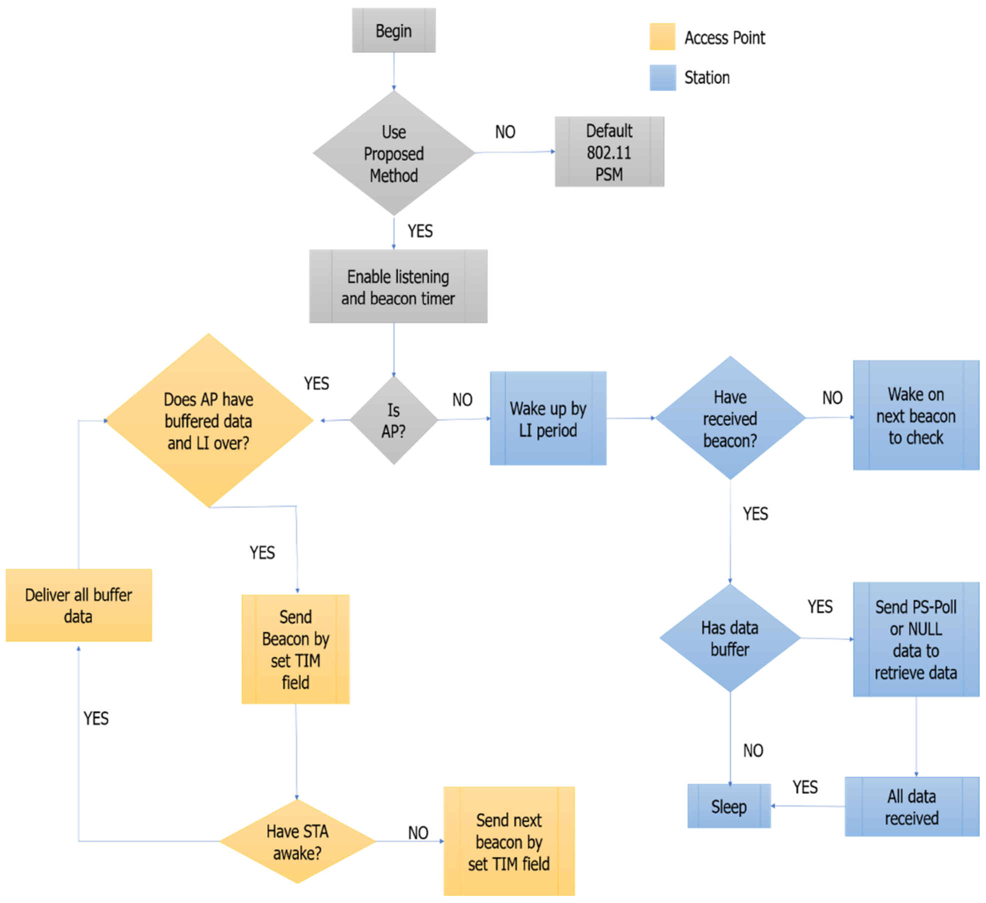


Figure 13. Data Delivery Flow Chart.

As the system's central unit, AP stores the packets for stations that have gone into power save mode. Previously, buffer allocation for all stations was also fixed when the listen interval and the absolute power save mechanism were static. It is a waste of memory. However, in the proposed method, since the listen interval can be dynamic, AP needs to adapt to the change (Algorithm 2) and change the buffer limit allocated for each station. The proposed approach uses the listen interval, maximum data rates supported via STA, and the Number of Special Streams (NSS) to define how much maximum buffer should be allocated to the station. Previously, static allocation methods such as storing buffer data in arrays could be used. However, now such practices would be expensive, and rather than these, we suggest holding the buffered data using a heap data structure with linked lists. One of the reasons for adopting this methodology is that now the AP does not know how much the listen interval will be, and it can change according to the needs of the STA. Hence, a linked list is a better choice as, unlike arrays, the size of a linked list is not predefined, allowing the linked list to increase or decrease in size as the program runs. Another reason for using the heap data structure is that the AP uses queues to send packets in the FIFO mechanism. Now, to support the Wi-Fi Multimedia (WMM) standard and QoS, queues should also need to store the data as per the priority of the packets. The heap is a'data structure used to store such queues according to their priority in the form of binary trees. The algorithm uses max heap data structure, which means the root node must be the greatest among the keys present at all its children. The exact property must be recursively true for all sub-trees in that binary tree. Hence, whenever AP needs to dequeue the buffered packet, it will send the root node buffered data, which will have the highest priority. Of course, the ageing technique is used to overcome the starvation of the deque frame.

WMM prioritizes traffic according to four Access Categories (AC): voice (AC_VO), video (AC_VI), best-effort (AC_BE), and background (AC_BK). Following this and in the order of packet received, the Rx queue will assign weights to each buffered packet, and this weight will be further used to decide the packet's position in the queue.

For different access category packets, different weights will be assigned in the order as (MAX)AC_VO → AC_VI → AC_BE → AC_BK(MIN). For same-category packets, weights will be assigned as per the FIFO mechanism. Therefore, the first packet received will have more weight than the packet received afterwards.

A new packet is attached at the tail, first per algorithm and then moving the packet up the order per the access category. Once it reaches the same access category packet or the root, stop iteration and enqueue the packet.

The discussed algorithm also ensures data delivery even when the beacon misses via STA (shown in Figure 13). If STA missed the beacon due to noise, congestion, or some reason, STA was still awake for the next beacon, and AP put the delivery status into the next beacon.

Algorithm 1: STA Dynamic Battery Change

```
Input:    batteryStatus            //current status of battery in percent
          powerState               //current power state, battery or charging
          defaultlisteninterval    //default listen interval
          gMaxLI                   //max listen interval supported by AP
          //assign variable to default values or with zero.
          newlistenInterval ←0     //variable to store new listen interval
Output:   newlistenInterval        //Latest LI value
1         if powerState == DeviceBatteryOn then
2         if (BatteryStatus < 5%) then
3         newlistenInterval ← z
4             else if (BatteryStatus < 10%) then
5                 newlistenInterval ←y
6             else if (BatteryStatus < 15%) then
7                 newlistenInterval ←x
8             else
9                 newlistenInterval←defaultlisteninterval
10            end if
11        else               //powerState == DeviceCharging
12        newlistenInterval←defaultlisteninterval
13        end if
14
15        if newlistenInterval > gMaxLI then
16        newlistenInterval←gMaxLI
17        end if
18
19        //send a frame to update the new listeninterval to AP
20        updatelistenintervaltoAP(newlistenInterval)
21
22        return newlistenInterval
End Algorithm
```

Algorithm 2: AP Dynamic Buffer Change

```
Input:      gMaxLI                          //max listen interval supported by AP
            ReqLIbySTA                      //requested listen interval by STA
            defaultSTABackupBuffSize        //default STA buffer size
            //assign variable to default values or with zero.
            STABackupBuffSize ←0            //variable to store allocated buffer
Output:     STABackupBuffSize               //Allocated buffer size to station
1           if ReqLIbySTA > gMaxLI then
2           STABackupBuffSize←0
3           return 0          //reject the ReqLIbySTA to STA.
4           end if
5
6           if ReqLIbySTA >=z    then
7                 STABackupBuffSize ←(defaultSTABackupBuffSize*z)
8           or else if ReqLIbySTA >=y    then
9                 STABackupBuffSize ←(defaultSTABackupBuffSize*y)
10          or else if ReqLIbySTA >=x    then
11                STABackupBuffSize ←(defaultSTABackupBuffSize*x)
12          or else
13                STABackupBuffSize←defaultSTABackupBuffSize
14          end if
15
16          return STABackupBuffSize
End Algorithm
```

7. Performance Evaluation

This section reports the power consumption results with the time of the proposed and conventional methods. The proposed listen adapter adaption decision can be taken via OS, depending on the type of application/requirement. For example, if some background traffic is running, OS can choose a high LI value for more power saving, and if any video/voice traffic is running, OS can go for a low LI value. There can be a difference between YouTube videos and Zoom meetings that OS can differentiate easily. In the IoT device environment, it is mostly STA transfer packets. STA is well aware of its data requirements to manage LI accordingly. Here, our goal is to show that "battery status" can be used to choose an adaptive LI value. A synthesis of the analyzed papers is summarized in other subsections.

7.1. Power Consumption Compare

First, we measured the power consumption concerning the changes in the LI. The same hardware on an Ubuntu-based Linux machine is used for conducting the simulation. Two modes have been chosen to perform the test—one is normal web browsing and the other is an idle test.

Script-based web browsing is performed for the first test. Data interchange will be uniform for all the web-browsing tests. The script performs browsing for the combination of voice, video, normal web-browsing, iperf [34], TCP throughput, and (File Transfer Protocol) FTP. There is one main machine connected to the ethernet switch with an AP. Iperf, a host FTP server, and a video server are run on this machine. User Datagram Protocol (UDP) and TCP are packets transferred from STA to AP and vice versa. The FTP server receives the files uploaded via the station, and the file is 10 Gb in size. Iperf is used to run TCP/UDP traffic, so the worst-case scenario and maximum throughput can be tested. The video server is used to run 4 K videos on STA. This traffic pattern represents daily use cases such as watching videos, email checking, file uploading/downloading, and web browsing on the wireless medium.

The results presented in Figures 14 and 15 show a higher LI value can save more power in every scenario. The test duration for the web-based test is 60 min, and for the idle test, it is 30 min. The test is performed with LI values of 1, 2, 4, and 8.

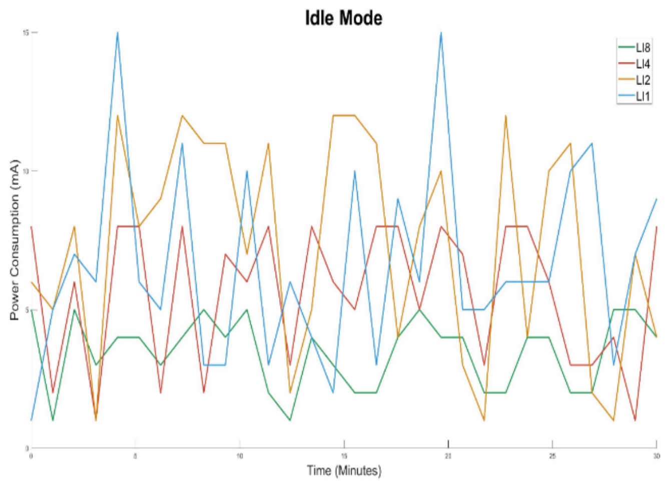


Figure 14. Station Power Consumption in idle mode.

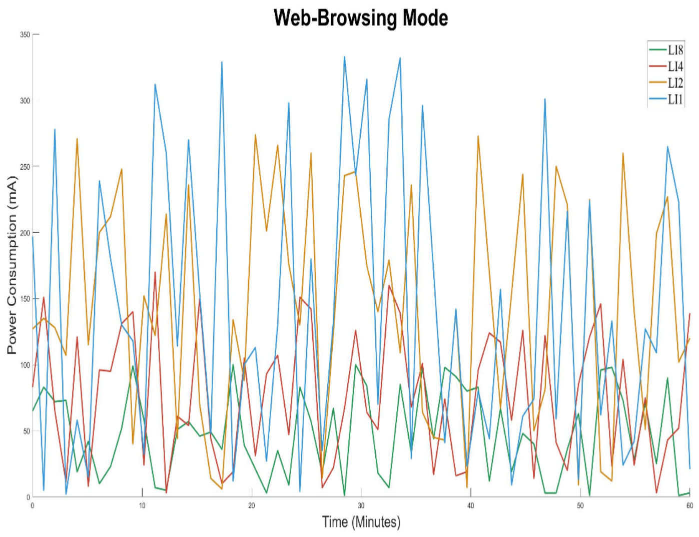


Figure 15. Station Power Consumption in web-browsing mode.

An Asus AP is taken to connect the station device (configuration shown in Table 2). A Wi-Fi power consumption reading is taken in millimetres, and the same is used to plot the graph (shown in Figures 14 and 15).

Table 2. Test Access Point Configuration.

Frequency	Mode	Channel	SSID	Bandwidth
5.0 GHz	11ac	36	Test-LI	80 MHz

The system is put into the idle state to perform the idle test, and there is no intentional traffic running on it, and every time, a Wi-Fi connection is to be maintained by STA. There is no such disconnection happening during the test.

7.2. AP Buffer Compare

Conventionally, static allocation is done for the STA buffer, which takes 100% of the memory. The proposed access point buffer management algorithm (using system simulation) runs for eight station devices to demonstrate the efficiency of the proposed access point framework. The results are shown in the graph (Figure 16).

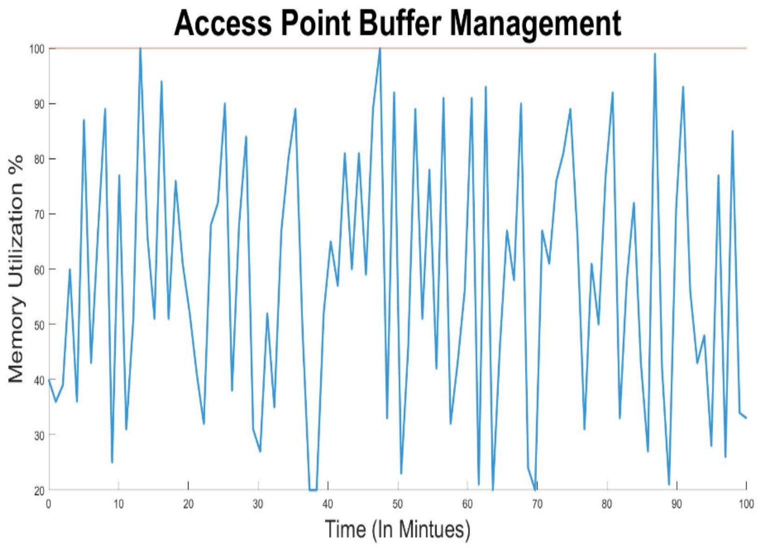


Figure 16. Access Point buffer memory utilization with time.

At the start of the tests, all software-based 8-stations connected to the access point and random Wi-Fi data and sleep tests are performed one hundred times to simulate the test. All the wireless entities present in the model strictly followed the CSMA/CA concept and proposed policies. A memory manager unit calculates memory usage during tests, and LI changes accordingly.

7.3. Summary

As discussed, our proposed solution can save significant power on the STA side. This paper only shows power consumption for the 802.11n mode with web browsing and idle state. In the wireless network and the Internet of Things world, each Wi-Fi supported 802.11 modes (b, g, n, ac & ax), different bandwidths, and different rates tested for millions of devices. Usually, IoT devices do not require a bulk amount of data transfer, and they send a small amount of data periodically.

Therefore, a large-scale proposed model can save significant power. With protocol implementation, the following corner cases are also covered:

- Beacon missed via STA
- Keep-alive time taken care of
- Upper layer timeout taken care of
- Delivery of unicast/multicast/broadcast data ensured.

Moreover, the proposed method saves a good amount of memory at the access point's lower layer side, considerably saving the processor time, which can be used for other tasks. For example, the ARP table can store at the lower layer for fast ARP resolution.

8. Conclusions and Future Works

We have shown how the proposed method can help stations to choose LI values based on the available battery and save reasonable power consumption. The performance results show that battery-operated devices can run for a long time with a long LI value, and AP can also optimize the buffer memory. Moreover, the framework achieved the practical workability of 802.11 use cases. The proposed protocol implements an adaptive LI mechanism between the station and AP with consideration of the 802.11 layers. Furthermore, no hardware change is necessary, making it easy to implement. The discussed method is new, and to our knowledge, no wireless instrument (available in the market) allows for changing dynamic STA's LI and AP buffers. Hence, system-based simulation is described in Section 7.2. Two Atheros ar9271 USB boards are taken to demonstrate a physical experiment, and the proposed protocol development and validation are added in Section 7.1.

Although the suggested approach significantly reduces Wi-Fi power consumption, it still has some points which must be addressed in the future.

1. The proposed method is physically tested in a system simulator using two Wi-Fi boards, but not in real environments. The proposed method can experiment with mobile devices, wireless sensors, notebooks, etc. Hence, real-time implementation needs to be done by Wi-Fi chip vendors.
2. How will AP deal with the situation where STA takes unfair advantage of the proposed method?
3. QoS data pattern can be analyzed with the proposed method.
4. Currently, IEEE specification does not provide any specific protocol which can change LI. A universal protocol and command can be introduced in the future, providing universal behaviour for all Wi-Fi vendors.
5. It would be interesting to cover all the failures and corner cases in the future.

Author Contributions: Conceptualization, V.B.; methodology, V.B. and N.R.; investigation, V.B.; writing—original draft preparation, V.B.; writing—review and editing, N.R.; research administration, V.B and N.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Wi-Fi Alliance. The Economic Value of Wi-Fi®: A Global View (2021–2025). September 2021. Available online: https://www.wi-fi.org/file/detail-global-economic-value-of-wi-fi-2021-2025 (accessed on 5 September 2022).
2. Friedman, R.; Kogan, A.; Krivolapov, Y. On Power and Throughput Trade-offs of Wi-Fi and Bluetooth in Smartphones. *IEEE Trans. Mob. Comput.* **2013**, *12*, 1363–1376. [CrossRef]
3. Seneviratne, S.; Seneviratne, A.; Mohapatra, P.; Tournoux, P.-U. Characterizing Wi-Fi connection and its impact on mobile users: Practical insights. In Proceedings of the 8th ACM International Workshop on Wireless Network Testbeds, Experimental Evaluation & Characterization, Miami, FL, USA, 30 September 2013; pp. 81–88.
4. *IEEE 802.11-2007*; 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 (MAC) and Physical Layer (PHY) Specifications. IEEE: New York, NY, USA, 2007.
5. Zhu, Y.; Lu, H.; Leung, V.C.M. Access Point Buffer Management for Power Saving in IEEE 802.11 WLANs. *IEEE Trans. Netw. Serv. Manag.* **2012**, *9*, 473–486. [CrossRef]
6. Valerrian Pasca, S.T.; Srividya, V.; Premkumar, K. Energy efficient sleep/wake scheduling of stations in wireless networks. In Proceedings of the 2013 International Conference on Communication and Signal Processing, Melmaruvathur, India, 3–5 April 2013; pp. 382–386.
7. Jain, N.; Payal, A.; Jain, A. Analysis of link failures and recoveries on 6to4 tunneling network with different routing protocol. *J. Intell. Manuf.* **2021**. [CrossRef]
8. Xie, Y.; Sun, X.; Chen, X.; Jing, Z. An adaptive PSM mechanism in WLAN based on traffic awareness. In Proceedings of the 10th IEEE International Conference on Networking Sensing and Control (ICNSC), Evry, France, 10–12 April 2013; pp. 568–573.
9. Li, Y.; Zhang, X.; Yeung, K.L. DLI: A dynamic listen interval scheme for infrastructure-based IEEE 802.11 WLANs. In Proceedings of the 2015 IEEE 26th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), Hong Kong, China, 30 August–2 September 2015; pp. 1206–1210. [CrossRef]
10. Fu, X.; Pace, P.; Aloi, G.; Li, W.; Fortino, G. Cascade Failures Analysis of Internet of Things Under Global/Local Routing Mode. *IEEE Sens. J.* **2022**, *22*, 1705–1719. [CrossRef]
11. Lopez-Aguilera, E.; Demirkol, I.; Garcia-Villegas, E.; Paradells, J. IEEE 802.11-Enabled Wake-Up Radio: Use Cases and Applications. *Sensors* **2020**, *20*, 66. [CrossRef] [PubMed]
12. LAN MAN Standards Committee of the IEEE Computer Society. IEEE 802.11 Standard—Wireless LAN Medium Access Control and Physical Layer Specifications. June 2007. Available online: https://ieeexplore.ieee.org/document/4248378 (accessed on 10 September 2022).
13. Deng, D.-J.; Lien, S.-Y.; Lin, C.-C.; Gan, M.; Chen, H.-C. IEEE 802.11ba wake-up radio: Performance evaluation and practical design. *IEEE Access* **2020**, *8*, 141547–141557. [CrossRef]

14. Khorov, E.; Kiryanov, A.; Lyakhov, A.; Bianchi, G. A tutorial on IEEE 802.11ax high efficiency WLAN. *IEEE Common. Surveys Tuts.* **2019**, *21*, 197–216. [CrossRef]
15. Wang, W.; Chen, Y.; Wang, L.; Zhang, Q. Sampleless Wi-Fi: Bringing Low Power to Wi-Fi Communications. *IEEE/ACM Trans. Netw.* **2017**, *25*, 1663–1672. [CrossRef]
16. Software-Defined Networking for Wi-Fi. White Paper. Available online: https://docplayer.net/9190422-Software-defined-networking-for-wi-fi-white-paper.html (accessed on 10 September 2022).
17. Saldana, J.; Munilla, R.; Eryigit, S.; Topal, O.; Ruiz-Mas, J.; Fernández-Navajas, J.; Sequeira, L. Unsticking the Wi-Fi Client: Smarter Decisions Using a Software Defined Wireless Solution. *IEEE Access* **2018**, *6*, 30917–30931. [CrossRef]
18. Omori, K.; Tanigawa, Y.; Tode, H. Power-saving for wireless stations using RTS/CTS handshake and burst transmission in wireless LANs. In Proceedings of the 2017 14th IEEE Annual Consumer Communications & Networking Conference (CCNC), Las Vegas, NV, USA, 8–11 January 2017; pp. 708–711.
19. Saeed, A.; Kolberg, M. Towards Optimizing WLANs Power Saving: Context-Aware Listen Interval. *IEEE Access* **2021**, *9*, 141513–141523. [CrossRef]
20. Lim, T.H.; Rhee, S.H. An Adaptive Power Management Scheme for WLANs using Reinforcement Learning. In Proceedings of the 2019 International Conference on Information and Communication Technology Convergence (ICTC), Jeju, Korea, 16–18 October 2019; pp. 412–415.
21. Liu, D.; Wang, H.; Zhou, G.; Mao, W.; Li, B. Arbitrating Traffic Contention for Power Saving with Multiple PSM Clients. *IEEE Trans. Wirel. Commun.* **2016**, *15*, 7030–7043. [CrossRef]
22. Wu, F.; Yang, W.; Ren, J.; Lyu, F.; Yang, P.; Zhang, Y.; Shen, X. Named Data Networking Enabled Power Saving Mode Design for WLAN. *IEEE Trans. Veh. Technol.* **2020**, *69*, 901–913. [CrossRef]
23. Jin, T.; Noubir, V.; Sheng, B. WiZi-Cloud: Application-transparent Dual Zigbee-WiFi Radios for Low Power Internet Access. *Proc. Infocom.* **2011**. [CrossRef]
24. Yomo, H.; Kondo, Y.; Miyamoto, N.; Tang, S.; Iwai, M.; Ito, T. Receiver design for realizing on-demand Wi-Fi wake-up using WLAN signals. In Proceedings of the 2012 IEEE Global Communications Conference (GLOBECOM), Anaheim, CA, USA, 3–7 December 2012; pp. 5206–5211. [CrossRef]
25. Kondo, Y.; Yomo, H.; Tang, S.; Iwai, M.; Tanaka, T.; Tsutsui, H.; Obana, S. Wake-up Radio using IEEE 802.11 Frame Length Modulation for Radio-On-Demand Wireless LAN. In Proceedings of the 2011 IEEE 22nd International Symposium on Personal, Indoor and Mobile Radio Communications, Toronto, ON, Canada, 11–14 September 2011.
26. Zhang, Y.; Song, Z.; Tian, Y.; Wang, W. A Runtime Framework for Context-Sensitive Device-to-Device Communication. In Proceedings of the 2017 IEEE 86th Vehicular Technology Conference (VTC-Fall), Toronto, ON, Canada, 24–27 September 2017; pp. 1–5.
27. Balasubramanian, N.; Balasubramanian, A.; Venkataramani, A. Energy consumption in mobile phones: A measurement study and implications for network applications. In Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement, Chicago, IL, USA, 4–6 November 2009.
28. Gupta, P.; Saxena, P.; Ramani, A.K.; Mittal, R. Optimized use of battery power in wireless Ad hoc networks. In Proceedings of the 2010 The 12th International Conference on Advanced Communication Technology (ICACT), Gangwon, Korea, 7–10 February 2010; pp. 1093–1097.
29. Fonseca, M.S.P.; Munaretto, A.; Mendes, C. A resource management framework for 802.11 wireless access networks. *Wirel. Netw* **2015**, *21*, 1891–1898. [CrossRef]
30. Afaqui, M.S.; Villegas, E.; Aguilera, E. IEEE 802.11 ax: Challenges and requirements for future high efficiency WiF. *IEEE Wirel. Commun.* **2016**, *24*, 130–137. [CrossRef]
31. Santi, S.; Tian, L.; Khorov, E.; Famaey, J. Accurate Energy Modeling and Characterization of IEEE 802.11ah RAW and TWT. *Sensors* **2019**, *19*, 2614. [CrossRef]
32. Bankov, D.; Khorov, E.; Lyakhov, A.; Stepanova, E. Clock drift impact on target wake time in IEEE 802.11ax/ah network. In Proceedings of the 2018 Engineering and Telecommunication (EnT-MIPT), Moscow, Russia, 15–16 November 2018; pp. 1–6.
33. Firmware Source Code for the Qualcomm Atheros AR9271 USB 802.11n NIC. Available online: https://github.com/qca/open-ath9k-htc-firmware (accessed on 10 September 2022).
34. NLANR. Iperf Measuring TCP and UDP Bandwidth Performance. 2005. Available online: https://users.informatik.haw-hamburg.de/~{}schulz/pub/Rechnernetze/tools/iperf/iperfdocs_1.7.0.html (accessed on 10 September 2022).