Towards Understanding and Enhancing Association and Long Sleep in Low-Power WiFi IoT Systems

Vikram K. Ramanna*†, Jaykumar Sheth*, Simon Liu*, and Behnam Dezfouli*
*Internet of Things Research Lab, Computer Science and Engineering, Santa Clara University, USA
†Infineon Technologies, San Jose, USA
{vramanna, jsheth, sliu3, bdezfouli}@scu.edu

Abstract—Enhancing the energy efficiency of WiFi IoT stations introduces unique challenges compared to 802.15.4 and BLE. The four essential operations performed to ensure connectivity between stations and the access point in a WiFi network are *association*, *periodic beacon reception*, *maintaining association*, and *station wake up*. Understanding and enhancing these operations are essential for building energy-efficient and dependable IoT systems. However, it is unclear how the software and hardware configuration of station and access point, concurrent traffic, power management, and security protocols affect the reliability and energy efficiency of these operations. In this paper, first, we present a thorough analysis of the association cost of WPA2 and WPA3 and mitigate the effect of key computation on association overhead. Second, we prove that increasing listen interval to reduce beacon reception wake-up duration may negatively impact energy efficiency. We identify the primary causes of this problem subject to link quality estimation algorithm and beacon delay. Third, we show that maintaining association by relying on access-point-based polling is not reliable. In particular, we confirm the wake-up delay of low-power stations is highly affected by factors such as channel utilization and beacon listen interval. We also confirm that key renewal aggravates the chance of disassociation.

Index Terms—Empirical Evaluation, Energy Efficiency, 802.11, WPA2, WPA3, Interference, Delay, Reliability.

1 Introduction

A WiFi network (a.k.a., 802.11 network) is composed of two components: *stations* and *Access Point (AP)*. Stations connect to the AP to communicate with other stations and the Internet. WiFi is an appealing technology for IoT connectivity [1]–[4] considering multiple reasons: First, large-scale deployment of WiFi APs provides a ready infrastructure for IoT connectivity in license-free bands. Second, the high data rate of this standard, compared to low-power technologies such as 802.15.4 and Bluetooth Low Energy (BLE), facilitates the development of applications such as medical monitoring, video streaming, process control, and robotics [5]–[7]. Third, the energy consumption of WiFi stations has been significantly reduced during recent years. Specifically, existing studies show the higher energy efficiency of WiFi's physical layer compared to BLE and LTE [3], [4].

Improving the energy efficiency of Internet of Things (IoT) systems is important from two perspectives: First, the impact of the energy consumption of these stations on global Information and Communications Technology (ICT) energy footprint is increasing [8]. Second, many of these stations (e.g., smart locks, security cameras) run on battery or energy harvesting mechanisms. Thereby, resource-constrained stations need to put their transceiver into sleep mode aggressively to reduce energy consumption whenever no communication is taking place.

Regardless of the application type, all WiFi IoT systems need to perform the following operations to ensure AP-station connectivity: *association*, *periodic reception of beacon packets*, *maintaining association*, and *station wake up*. Association refers to the process of exchanging station and AP information (e.g., supported data rates) and establishing keys for secure communication. Once associated, a low-power IoT station needs to periodically wake up and receive beacon packets sent by the AP. Beacon reception serves three primary purposes: first, the AP uses beacon packets to inform the stations about their buffered packets, second, the stations can measure their link quality to the AP, and third, stations sync up their clock with the AP. Maintaining association is performed by both sides—stations and the AP. If a station misses a particular number of beacons, it may initiate the roaming process or retry to reassociate with the AP. On the other hand, if the AP does not hear from a station for a particular time duration, it may try to poll the station or simply disassociates the station. The periodic wake up of stations and the need to maintain association may cause unreliable or delayed station wake-up. Despite the importance of these operations, existing studies primarily focus on the effects of traffic on energy efficiency [1], [9], [10], or evaluate performance at a high level [11], [12], and unfortunately, much less attention has been paid to the impact of the above operations on system dependability and energy efficiency.

In this paper, we present an empirical study of association, beacon reception, maintaining association, and station wake up delay, with focus on parameters including energy efficiency, reliability, and delay. We also propose methods to improve the performance of these operations. Specifically, the contributions of this paper are as follows.

Association Overhead. The process of associating a station with an AP impose a non-negligible delay and energy consumption burden on the station. Through empirical evaluations, first, we show that instead of exhaustive channel sweeping, using specific AP probing can reduce association

overhead by about 40%. However, using this method causes a higher number of association failures, especially when the key generation duration increases or channel utilization escalates. Second, we reveal the high overhead of IP assignment even when a static IP is used. Specifically, our results confirm the higher IP assignment overhead of ThreadX with NetXDuo stack compared to FreeRTOS with LwIP stack. Third, we compare WPA2 and WPA3 and confirm their relative performance depends on the processor and firmware implementation. Fourth, to reduce the overhead of WPA2 key generation, we offload the hashing function to the application processor. We then compare the performance of WPA2 with key offloading against WPA3 with key caching.

Beacon Reception. To reduce the overhead of beacon reception, a straightforward approach is to increase the station's listen interval. However, we substantiate that employing this method may considerably increase the overhead of beacon reception in terms of awake duration, which translates to higher energy consumption. We identify three reasons for this behavior: First, link quality estimation algorithms (implemented in station's firmware) exhibit various levels of sensitivity to beacon loss. Second, we signify that the Transmit Opportunity (TxOP) reserved by voice and video flows can cause considerable beacon transmission delay. Third, as beacon loss increases, the awake time of the station in advance of beacon reception may increase.

Maintaining Association. To avoid the overhead of reassociation, it is essential to maintain association with minimum overhead. This is particularly challenging because APs maintain a per-station inactivity timer to ensure the station is alive and within the communication range. Although APs, by default, poll inactive stations periodically, we prove the unreliability of this method. In particular, this may lead to what we call a 'disassociation-unaware station', which means the station is unaware of its disassociation by the AP. We also reveal key renewal as a time-critical operation that may result in disassociation. We present solutions to remedy these problems.

Wake-up Delay. Ensuring short, reliable wake-up delays prevents cases such as polling failure and key renewal, and it is also important for applications where the station must be woken up by user or cloud applications. We consider the two widely-used power-saving methods of WiFi, namely Power Save Mode (PSM) and Automatic Power Save Delivery (APSD), and quantify wake up delay. Our main findings are as follows. First, uplink traffic has a higher effect on prolonging wake up duration. Second, although APSD is accepted as a more efficient method to reduce AP-station delay, the wake up delay of this method is higher than that of PSM.

Segregating the overall system operation into the above fundamental components allows for extending the presented studies and methods to a wide range of applications, hardware, and software platforms. This segregation also allows us to tackle the challenges of holistic system evaluation considering the large number of combinations of configuration parameters.

The rest of this paper is organized as follows: Section 2 overviews the association process and power-saving mechanisms. Section 3 presents our research methodology. Association overhead is studied in Section 4. In Section 5, we study the impact of listen interval on energy efficiency. Mechanisms for preventing reassociation are presented in Section 6. A quantitative study of wake up delay is given in Section 7. We overview the related work in Section 8 and conclude the paper in Section 9.

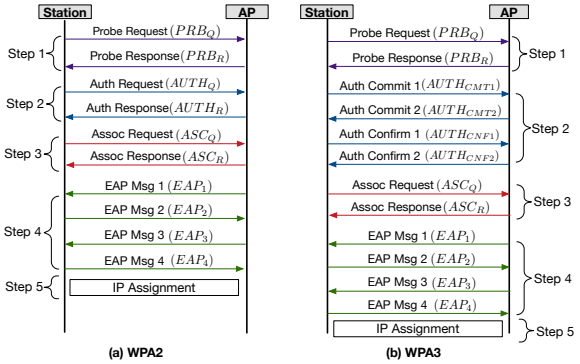

Fig. 1: Association process performed by (a) WPA2 (b) WPA3.

2 Background

In this section, we provide an overview of the association process as well as the mechanisms that enable stations to leverage power saving via switching off their wireless transceiver.

2.1 Association Process

The association process is composed of the following steps, as demonstrated in Figure 1. *Step 1*: The station scans for nearby APs by sending probe requests on all the channels. Among the APs whose Service Set Identifier (SSID) matches with that programmed in the station, the compatible AP with the highest signal strength is selected. *Step 2*: The station authenticates with the AP. With WPA2, this primarily means the two sides share information such as their MAC addresses to generate Pairwise Master Key (PMK). With WPA3, the station and AP share information (e.g., calculated scalar and point) that allow them to perform the Simultaneous Authentication of Equals (SAE) handshake, a method that has been originally designed for 802.11s mesh networks. The SAE handshake, which is similar to Diffie–Hellman (DH) key exchange with authentication, is based on a zero-knowledge proof algorithm known as Dragonfly. *Step 3*: The station sends an association request to the AP and awaits an association response. The primary result of this step is assigning an Association ID (AID) to the station. *Step 4*: A 4-way key exchange happens to generate Pairwise Transition Keys (PTK) and Group Temporal Key (GTK), which are used for unicast and broadcast communication, respectively. *Step 5*: If no static IP has been configured, the station requests for IP address using DHCP.

2.2 Energy Saving Mechanisms

The 802.11 standard offers various power-saving methods. The Power Save Mode (PSM) enables the stations to wake up periodically at each Beacon Interval (BI). The BI value used by commercial APs is 102.4 ms. Before transitioning

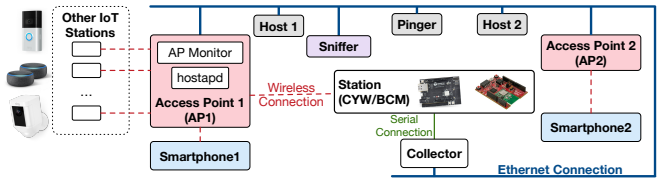

Fig. 2: The testbed used in this work. As we will explain in the subsequent sections, various experiments may employ subsets of this testbed's components.

into the sleep mode, the station informs the AP about its decision via a successful packet exchange where the power management bit (in the MAC header) is set. Every BI instance, the AP broadcasts a beacon message which, in addition to other information, conveys to the stations if the AP has buffered packets for the stations. This is performed by setting the AID bit of the station in the Traffic Indication Map (TIM) field. In this case, the station stays awake, sends a PS-Poll message to the AP, and waits for downlink packet transmission.

With PSM, stations need to retrieve downlink packets every BI. Since the delay of this mechanism may not be acceptable for interactive applications, the Automatic Power Save Delivery (APSD) mechanism has been introduced. With APSD, a station can poll the AP anytime by sending a Null packet. The APSD mechanism, which is part of the 802.11e amendment (and is available in 802.11n/ac/ax), also introduces the concept of Access Category (AC) to differentiate the priority of voice, video, best-effort, and background traffic types. From left to right, the priority of channel access reduces.

To deliver multicast and broadcast packets such as Address Resolution Protocol (ARP), the AP needs to ensure all the stations are awake. These packets are delivered after the transmission of a beacon packet including the Delivery Traffic Indication Message (DTIM) element. The DTIM period is configured as a multiple of beacon interval in the range of 1 to 255.

3 Methodology

Figure 2 shows the testbed used in this paper. We explain the components and operation of this testbed as follows.

3.0.1 IoT stations

We use multiple types of stations in this paper. However, the main two low-power WiFi transceivers used in our studies are: CYW43907 [13], and BCM4343W [14]. CYW43907 (*henceforth as the CYW station*) includes two ARM-Cortex R4 processors. BCM4343W (*henceforth as the BCM station*) includes two ARM-Cortex M3 processors. In both systems, one core is dedicated to the *wireless subsystem* and the other core comprises the *application subsystem*. These two transceivers are used by various low-power IoT development boards as well as products such as Amazon Tap, LG Urbane Watch, and Samsung Gear S2. The operating systems supported by these boards are FreeRTOS [15], [16] and ThreadX [17], [18] and the networking stacks are LwIP [19] (for FreeRTOS) and NetXDuo [20] (for ThreadX). Unless otherwise mentioned, ThreadX and NetXDuo are, respectively, the operating system and network stack used by these stations.

3.0.2 AP

Unless otherwise mentioned, the transceiver used by the APs is Atheros AR9462. AP functionality is handled by `hostapd`, which is a user-space daemon used by most commercial APs. We use the 802.11n standard as the communication protocol between stations and AP. The APs operate in the 2.4 GHz band. The APs and stations support PSM and APSD.

3.0.3 Channel utilization

We access driver's counters to measure the channel utilization consumed by Downlink (DL)/Uplink (UL) and interference traffic. Since we use Atheros transceiver, we monitor `AR_CCCNT` register value, which stores the time elapsed since the start time of the transceiver, and `AR_RCCNT` register value, which stores the activity duration sensed on the channel. We refer to the value of the first and second registers as T and B, and compute channel utilization during an interval $t1$ to $t2$ as $(B_{t2} - B_{t1})/(T_{t2} - T_{t1})$.

3.0.4 Controlled concurrent channel access scenarios

To measure the effect of concurrent traffic, channel access, and link unreliability, we use the following cases: presence of DL/UL traffic, and presence of interference. In the DL/UL case, AP1 is associated with the CYW/BCM station and Smartphone1. AP1 and Smartphone1 continuously exchange DL and/or UL traffic. In the interference case, AP1 is associated only with CYW/BCM; also, AP2 and Smartphone2 (associated with it) are operating in the vicinity (about three meters) and exchange DL and UL traffic. AP1 and AP2 operate on the same channel. Host1 and Host2 are used to generate DL traffic from AP1 and AP2 to Smartphone1 and Smartphone2, respectively. The Pinger is used for generating ping packets towards the station under test. The distance between stations and their associated AP is about two meters.

Figures 3 (a) and (b) characterize CYW-to-AP and AP-to-CYW station link quality by demonstrating the average number of retransmissions per packet. We ran each experiment 10 times for 1000 packets sent per round. In other words, for each round of 1000 packets, we compute packet retransmission rate by dividing the total number of retransmissions by 1000. We observe that DL, UL, and interference almost equally affect station-to-AP link. In contrast, for AP-to-station communication, the effect of DL traffic is almost negligible because it causes packet transmission delay instead of packet loss. Note that the maximum number of MAC layer retransmissions per data packet is 7 with most wireless transceivers.

3.0.5 Power measurement

The energy measurement tool is EMPIOT [21]. This platform's basic sampling rate is 500 Ksps, which are then averaged and streamed to the control software at 1 Ksps. Its maximum accuracy error is 4% compared to the existing high-end commercial products.

4 Association Overhead

In this section, we empirically evaluate the overhead of association process and present methods to alleviate it.

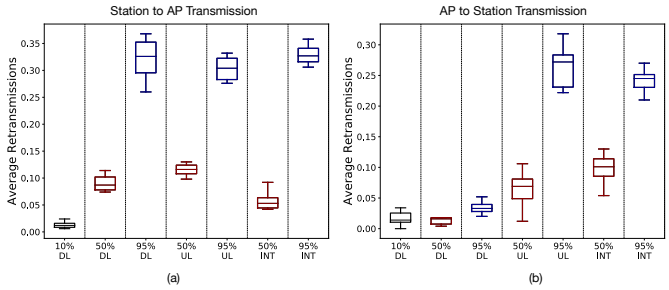

Fig. 3: Average number of retransmissions per data packet for (a) station-to-AP transmission, and (b) AP-to-station transmission. X-axis values refer to the level of channel utilization by Downlink (DL), Uplink (UL), and interference.

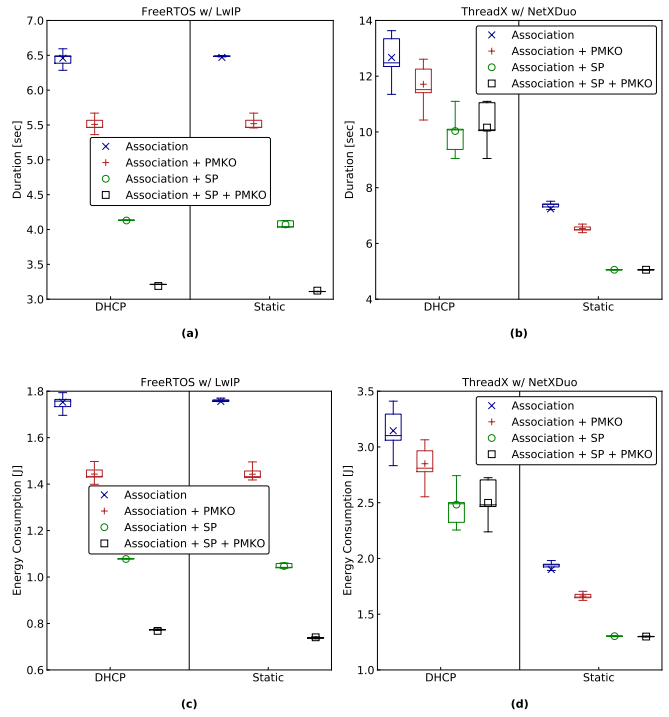

Fig. 4: The duration ((a) and (b)) and energy consumption ((c) and (d)) of various association methods using the CYW station. The PMK offloading method is referred to as PMKO, and the specific probing method is denoted as SP.

4.1 Effect of Probing, Key Generation, Operating System, and Network Stack on Association

In this section, we study the impact of various parameters on association. Channel utilization level in all the experiments is less than 10%. The station used in these experiments is CYW. With regard to the testbed demonstrated in Figure 2, only the CYW station and AP1 are the active components of these experiments. We use the WPA2 protocol in this section. Also, it is worth noting that WPA2 is still the most widely used authentication protocol on IoT stations, including Raspberry Pi (RPi) (3B+ and 4), Ring Camera, Nest Camera, and Amazon Echo. Figure 4 presents the results.

4.1.1 Probing

The probing process requires the station to send multiple probe packets on each channel, and wait long enough (dwell time) for the reception of responses. This process increases the energy consumption of the station and elongates the association process.

Specific Probing (SP). In this method, we reduce association overhead by sending a probe request to a particular AP on a channel known to the station. We implement SP by using driver interfaces that allow us to program the channel and SSID of a particular AP in the station's firmware. When using SP, the probe request is still a broadcast packet, however, the packet is sent on the programmed channel only and includes the SSID of the target AP. As Figure 4 shows, for FreeRTOS, the SP method reduces the duration and energy consumption by 39.66% and 43.58% on average, respectively, compared to a regular association. These values are 21.73% and 21.67% for ThreadX.

4.1.2 PMK Computation

During the 4-way handshake of WPA2, a shared secret key called PMK is generated. Using SHA-1 algorithm, PMK is derived from Pre-Shared Key (PSK), SSID, and SSID length using the Password-Based Key Derivation Function 2 (PBKDF2) hashing algorithm as follows: `PBKDF2(HMAC-SHA1, PSK, SSID, 4096, 256)`. Here, a 256-bit PMK is generated from 4096 iterations of the hashing method. This function is implemented in the wireless firmware, and therefore, it is run by the transceiver's processor. However, PMK generation is a process-intensive operation and increases the energy consumption and duration of association.

PMK Offloading (PMKO). In this method, we offload the calculation of PMK to the application processor. It is worth noting that considering the complexity of 802.11 compared to standards such as 802.15.4, a separate processor is used in the wireless subsystem of WiFi stations. For example, both CYW and BCM stations include two processors: one for wireless connectivity, and one for running application threads. Therefore, the PMKO method can leverage the application processor, which is usually at least as powerful as the wireless subsystem's processor. We implemented PMKO on the CYW station by computing PMK in the application processor and then passing it to the transceiver.[1] As soon as the station boots up, the application processor computes PMK and makes it available to be used during the association process.

The results in Figure 4 demonstrate the effect of PMKO on association overhead. On average, using PMKO with FreeRTOS reduces delay and energy consumption by 18.87% and 23.38%, respectively. These values are 4.01% and 5.39% for ThreadX. Although ThreadX shows higher association overhead compared to FreeRTOS, the difference is not caused by the WPA2 method. We observed that the higher overhead is caused by IP assignment processing and the packet exchanges made to inform the AP (and other nodes) about IP assignment. Therefore, the IP assignment overhead of ThreadX overshadows the benefits achieved by using PMKO.

1. The implementation is available at the following link: https://github.com/SIOTLAB/Low-Power-WiFi-Association.git

4.1.3 IP Assignment

Although using static IP allocation prevents the need to communicate with a DHCP server, the results in Figure 4 show the considerable impact of static IP allocation when using ThreadX w/ NetXDuo, compared to a marginal effect with FreeRTOS w/ LwIP. With NetXDuo, the association process includes the transmission of Gratuitous ARP packets after IP assignment. The station broadcasts these ARP packets to announce its IP to the entire network. This extra step is not performed by LwIP stack.

4.2 WPA2 vs WPA3

In this section, we compare the overhead of WPA2 and WPA3 while varying channel utilization level by controlling the amount of UL and DL traffic exchanged between AP1 and Smartphone1 (§3). The AC of this traffic is voice. Also, considering the performance benefits of SP (§4.1.1), we use this method for all the experiments. Compared to the experiments demonstrated in the previous section, and in order to make the observations independent of the operating system and protocol stack used, in this section we merely focus on the first four stages of association process, as demonstrated in Figure 1. Therefore, the overhead of IP assignment has been excluded from all the results of this section.

4.2.1 WPA2 vs WPA3

Similar to WPA2, a PMK must be calculated during Step 2 of WPA3 association. However, this calculation is different than that of WPA2; this is because the main vulnerability of WPA2 is its heavy reliance on using PSK to compute PMK. In contrast, WPA3 uses zero-knowledge proof for PMK generation to ensure no elements of the PSK are exchanged between the station and AP. This calculation introduces a heavy processing load due to hashing and DH key generation. Figure 5 compares the overhead of WPA2 and WPA3. We first discuss the default operation of the two algorithms, demonstrated simply as WPA2 and WPA3 in the legends of this figure. With CYW, the duration and energy consumption of WPA3 are lower than that of WPA2, however, the overhead of WPA3 is higher than WPA2 for BCM station. Since we use a similar firmware and driver on the CYW and BCM stations, their main difference is the processor used in their wireless subsystems. CYW includes a Cortex-R4 processor which implements the ARMv7-R architecture with an eight-stage pipeline and includes data and instruction caches. BCM's wireless subsystem includes a Cortex-M3 processor, which implements the ARMv7-M architecture with a three-stage pipeline and without cache memory. We conjecture that the performance differences are caused by the wireless subsystem's processor type. To verify this, we chose additional off-the-shelf stations that employ Cortex-M3 (CYW43364 [22]) and Cortex-R4 (CYW43455 [23], RPi) in their wireless subsystem. Since the default firmware and driver of the RPi do not support WPA3, we complied a Full MAC (F-MAC) Linux driver and flashed a new firmware to this station to enable WPA3 association. Therefore, the driver and firmware used for RPi's CYW43455 are different than those used by CYW43455. Figure 6 presents the PMK calculation and total association duration of these

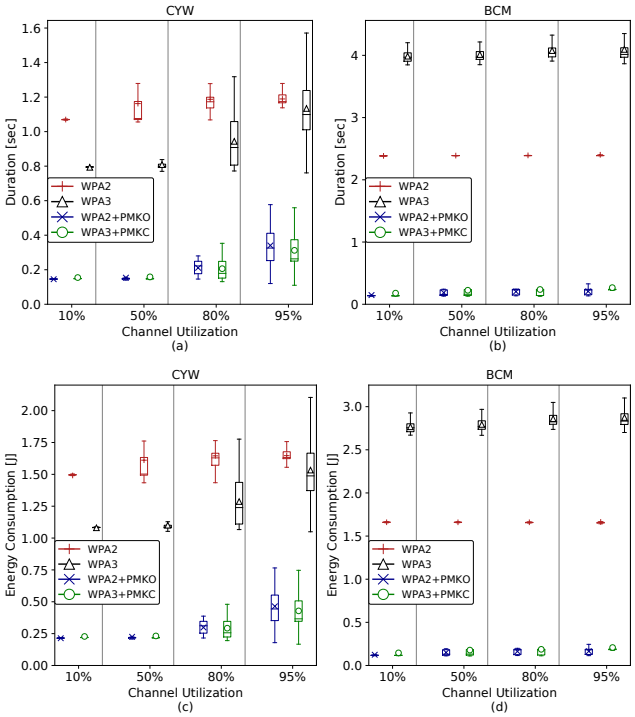

Fig. 5: The duration ((a) and (b)) and energy consumption ((c) and (d)) of WPA2 and WPA3 association using CYW and BCM stations. Channel utilization is generated by controlling the amount of UL and DL traffic between AP1 and Smartphone1. All the association scenarios use SP.

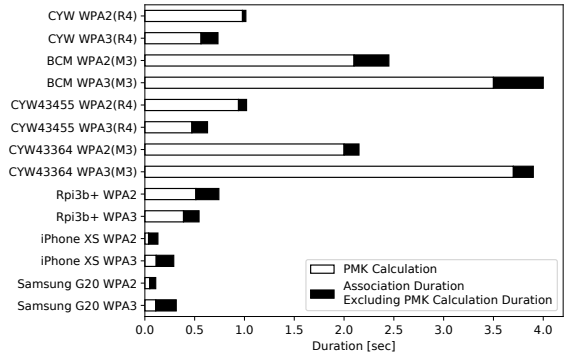

Fig. 6: PMK computation and total association duration for stations that use Cortex-M3 (BCM4343 [14], CYW43364 [22]) and Cortex-R4 (CYW43907 [13], CYW43455 [23], RPi3B+) in their wireless subsystem.

stations. These results were collected when the channel utilization was no more than 10%. To compute PMK calculation duration, we used the Sniffer (cf. Figure 2) and measured the time interval between the reception of probe response packet (PRB_R) and the transmission of first authentication message ($AUTH_{CMT1}$). The results confirm that the ratio of WPA3 to WPA2 PMK calculation duration is less than one for Cortex-R4 stations; whereas, the ratio is greater than one for Cortex-M3 stations. The results also reveal the longer duration of WPA3 compared to WPA2 for Apple iPhone XS and Samsung Galaxy phones; however, we can cannot confirm the processor and driver type of these stations.

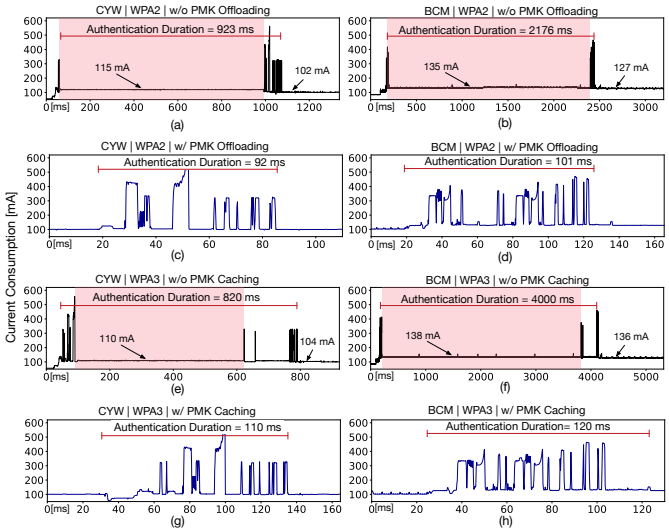

Fig. 7: Energy consumption of CYW and BCM stations during the association process using WPA2 and WPA3. The shaded parts refer to PMK calculation.

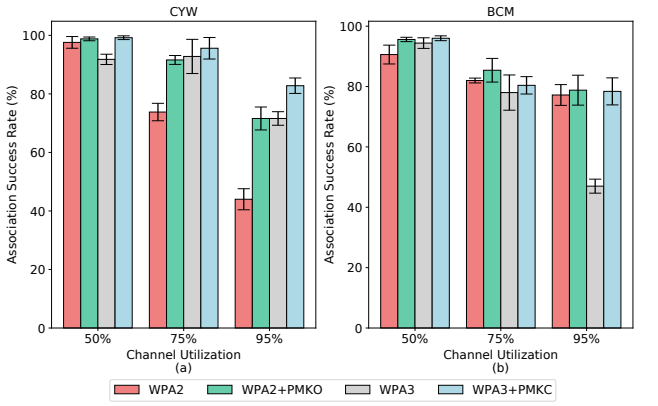

Fig. 8: Association success rate versus channel utilization level for WPA2 and WPA3. These results exclude the failures caused by missing probe request and response packets.

4.2.2 PMK Caching for WPA3

Although PMK offloading is not possible with WPA3, we can expedite the association process by using PMK Caching (PMKC), which is part of the standard. If the station and AP possess PMK caches, they can skip SAE and proceed with the 4-way handshake. With regard to Figure 1, using PMKC reduces the number of packets exchanged during Step 2 to two packets. Specifically, the station uses the previously-computed PMK by sending an authentication message with the authentication algorithm set to open. The AP then checks the validity of PMKID, which is a unique key identifier maintained by the AP on a per-station basis. If the PMKID is valid, the AP responds with an authentication commit response packet and the association process proceeds with association request. Figure 7 presents the power consumption trace of CYW and BCM stations and compares how PMKO and PMKC affect WPA2 and WPA3, respectively. These results were collected when the channel utilization was no more than 10%. As the results show, both PMKO and PMKC eliminate the long flat part of the graph where PMK computation is occurring. For example, on the CYW station, using PMKC reduces the total association duration of WPA3 from 820 ms to 110 ms, and on the BCM station, the drop is from 4000 ms to 120 ms. These results also show the higher power consumption during key calculation. Referring back to Figure 5, the duration and energy consumption of WPA2 and WPA3 are very similar when PMKO and PMKC are used. This is because, in addition to excluding the heavy processing load of key calculation, both WPA2 and WPA3 need to exchange eight packets to perform association.

4.3 Channel Utilization

The results in Figure 5 confirm that as the channel utilization escalates, the duration and energy consumption of association process increase. This is because intensifying channel utilization increases packet collision rate, the number of retransmissions, and packet exchange duration.

To quantify how packet loss and delay affect association reliability, we measured the association success rate of WPA2 and WPA3 in the presence of various levels of channel utilization. Figure 8 presents these results. A successful association requires completion of the four steps explained in §2.1, and it is important to note that the success probability of these steps vary. Probe request is not followed by layer-2 acknowledgments; it is simply transmitted for a given number of times, depending on the station and AP configuration. For example, the CYW43907 firmware sends three probe request packets. Hence, if the station or AP miss these packets, the whole association process fails. For the results in Figure 8, we have excluded the failures occurring during Step 1, and instead, focused on the rest of the steps. In contrast with probe request, the following packets require layer-2 acknowledgment: probe response, authentication (Step 2), association (Step 3), and 4-way handshake (Step 4). Nevertheless, the number of allowed retransmissions is implementation specific. For example, with CYW43907, the maximum number of retransmissions per packet of Step 2, Step 3, and Step 4 are 3, 7, and 7, respectively. Therefore, intensifying channel utilization causes lower success rate.

As Figure 8 shows, for channel utilization rates higher than 50%, the success rate of WPA3 is higher than WPA2 for the CYW station, while the success rate of WPA2 is higher than WPA3 for the BCM station. By analyzing `hostapd`, we identified a timer is started for the station once a probe request is received. We also observed that this timer is only used if the station employs the SP method. In other words, by receiving a specific probe packet from a station, the AP is informed that the station is obliged to associate with the AP. If the authentication packet is not received from the station before the timer expiry, the AP sends a disassociation packet to the station and fails the association process. As explained earlier, the CYW and BCM stations demonstrate shorter key computation delays for WPA3 and WPA2, respectively; thereby, a shorter duration reduces the chance of failure. On the other hand, during high channel utilization, the packet delivery delays caused by retransmissions further exacerbate the chance of timer expiry.

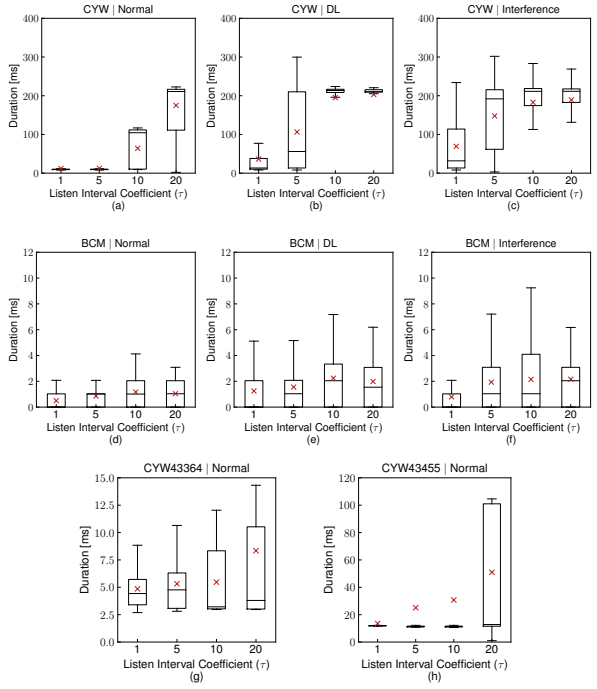

Fig. 9: Per-beacon awake duration. (a)-(c): Awake time of CYW station with normal (a), DL (b), and interference (c) traffic. (d)-(f): Awake time of BCM station with normal (d), DL (e), and interference (f) traffic. (g): Awake time of CYW43364 with normal traffic. (h): Awake time of CYW43455 with normal tarffic. Cross marks represent the mean values. Outliers are not demonstrated. These results show that the duration of receiving each beacon is affected by increasing listen interval and channel utilization.

5 Beacon Reception Overhead

Each AP transmits beacons every Target Beacon Transmission Time (TBTT), which is normally set to 102.4 ms. However, the default wake-up interval causes waste of energy in two types of applications: (i) if the station-AP traffic pattern is primarily uplink, or (ii) if the delayed delivery of downlink packets to the station can be tolerated. To reduce the number of periodic wake-ups, we modified the driver to specify the wake-up duration. Specifically, the *listen interval coefficient* (denoted as τ) specifies wake-up duration as a multiple of Beacon Interval (BI). For example, when $\tau = 10$, the station wakes up every 10×102.4 ms.

We study beacon reception overhead in three scenarios: (i) *normal*: where the overall channel utilization does not exceed 10%; (ii) *DL*: where AP1 transmits voice traffic to Smartphone1 and consumes 95% of channel capacity; (iii) *interference*: where AP2 and Smartphone2 exchange bidirectional voice traffic and consume 95% of channel capacity.

Figure 9 presents the results. For all the stations, increasing the listen interval results in higher awake duration. However, the increases in the awake duration of CYW and CYW43455 are considerably higher compared to BCM and CYW43364. Also, comparing Figures 9(a)-(c) with (d)-(f) confirm the higher sensitivity of the CYW station to DL/UL traffic and interference. The underlying causes of these observations are threefold: the link quality estimation algorithm, beacon transmission delay, and variations of the wake-up timer of the station. We detail these causes as follows.

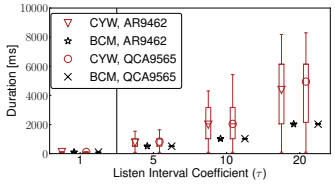

Fig. 10: Inter-beacon awake duration for CYW and BCM stations in the presence of 95% channel utilization. AR9462 and QCA9565 refer to the wireless card used in AP1.

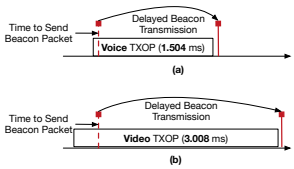

Fig. 11: The presence of a flow belonging to the voice AC can result in delayed beacon transmission.

5.0.1 Link Quality Estimation

Beacon loss happens due to multiple factors such as packet collision, low signal strength, and beacon delay. Management frames, such as beacons, are more prone to collisions because they are not preceded by RTS/CTS. To maintain its association with the AP, each station needs to receive a certain number of beacons per second, depending on the link quality estimation algorithm implemented in the firmware. When the beacon interval is one, even if the station does not receive a beacon at some beacon instances, the station can simply switch back to sleep mode and look for beacon during the next beacon instance. By increasing the listen interval, the tolerance against beacon loss is reduced due to the need for time synchronization and link quality estimation. In this case, the station stays awake for a longer duration to look for incoming packets. To demonstrate this behavior, we measured the awake duration between beacon reception instances in the presence of 95% interference traffic. We measured these intervals by accessing driver's counters that are updated per beacon reception, and then correlating the counter values with power traces to measure awake duration. In these results, to ensure the variations of awake duration are not caused by the Time Synchronization Function (TSF) of AP's wireless transceiver, we ran the experiments using two different wireless transceivers: AR9462 and QCA9565. Figure 10 shows the results. Regardless of the AP's transceiver, we observe that the CYW station is more sensitive to beacon loss, and the sensitivity increases versus listen interval value. Also, the awake duration between two beacon receptions may be as short as one BI or span multiple BIs. For example, by tracking power utilization, we observed that there exists cases where the station stays in awake mode after the reception of a beacon until receiving the next beacon, and this results in 102.4 ms awake duration between beacons. *Although a higher sensitivity to beacon loss is desired to facilitate handoff and avoid disconnection from the AP, these results show its adverse effects on beacon reception overhead.*

5.0.2 Beacon Transmission Delay

The presence of concurrent traffic, especially voice and video, causes variations of inter-beacon transmission intervals and increases awake duration. Cisco estimates 79% of all mobile traffic will be video by 2022 [24]. As per the 802.11e amendment, the voice and video ACs can reserve Transmit Opportunity (TxOP) of size 1.504 ms and 3.008

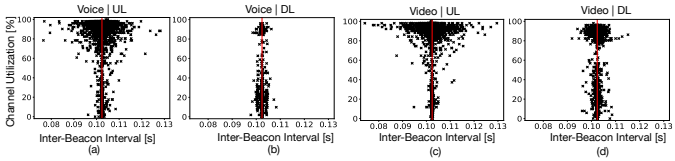

Fig. 12: Impact of DL and UL traffic on beacon transmission delay. (a) and (b): voice AC. (c) and (d): video AC. UL traffic has a higher effect on beacon transmission delay because it prevents the AP from channel access during TxOP.

ms, respectively. During a TxOP, no other station in the network commences a transmission. This is demonstrated in Figure 11. Hence, although at each TBTT the AP is ready for the transmission of a beacon, it would not be allowed to send it if a station has reserved a TxOP. To demonstrate this behavior in a real-world scenario, we measured channel utilization and correlated it with beacon transmission delay. Figure 12 displays the impact of UL and DL traffic on beacon delay. These results confirm the increase of inter-beacon intervals as channel utilization is intensified. Consider three beacon transmission time instances, t_k, t_{k+1}, and t_{k+2}. Assume t_k and t_{k+2} are sent at their designated time. If t_{k+1} is delayed, then $t_{k+1} - t_k > 102.4$ ms and $t_{k+2} - t_{k+1} < 102.4$ ms. If t_{k+2} is delayed, then $t_{k+2} - t_{k+1}$ can be shorter, longer, or equal to 102.4 ms. The results also demonstrate a lower beacon transmission variation with DL traffic. This is because beacons are prioritized over all data and management frames in the AP's wireless driver. For example, the software queues inside Atheros drivers (e.g., ath9k) are organized as follows: a queue for beacon transmission, a queue for management, multicast, and broadcast packets, and eight queues for the 802.1p traffic priorities (which are then mapped to four ACs). The queue assigned to beacon transmission has the highest priority; thereby, the beacons are not affected significantly by the DL traffic sent by the AP.

5.0.3 Wake-up Timer

The timestamp inside beacons contains the value of the TSF timer when the first bit of the timestamp is handed to the physical layer (PHY) from the MAC-PHY interface and also compensates the transmission delay from the PHY to the antenna. The receiving station sets the value of its TSF timer to the timestamp present inside the beacon. To investigate if CYW can wake up on time to receive beacons, we measured the awake time of this station before each beacon instance. Our results show that CYW's beacon loss rate increases versus listen interval because it cannot properly adjust its timer to wake up on time, even in scenarios where channel utilization is around 10%. We also noticed that this duration almost doubles for each step of increasing listen interval coefficient (we do not include the results due to space limitation); thereby confirming the impact of time synchronization on awake duration.

The studies presented in this section clarify that *increasing listen interval cannot be simply used as a method to improve energy efficiency* because transceivers exhibit different overheads considering listen interval duration, beacon transmission delay, and interference.

As the value of τ increases, the chance of receiving multicast and broadcast packets drops. This is because the DTIM value on commercial APs is usually between 1 to 3, which means multicast and broadcast packets are transmitted every 1 to 3 BIs. Various solutions can be used to address this problem when $\tau > 1$. For example, reliable delivery of multicast and broadcast packets can be achieved by converting these packets into unicast packets. Also, the AP may prevent the transmission of unnecessary packets such as ARP. For ARP packets, a proxy implemented on the AP can be leveraged to reply to queries, without having to involve stations.

6 Maintaining Association

Maintaining association is essential to ensure: (i) stations spending most of their time in sleep mode can communicate with the AP without having to reassociate, and (ii) the AP can wake up and deliver downlink packets to the station. In this section, we identify the underlying challenges of maintaining association and present solutions to address them.

Referring back to Figure 2, the testbed configurations used for this experiment are as follows. AP Monitor is a user-space daemon running on AP1 to monitor association status, connected time, and idle time of the station. AP Monitor continuously transmits its status to the Collector via an Ethernet connection. The Sniffer is used to capture the packets exchanged between AP1 and the station. The sniffer reports to the Collector via an Ethernet connection. The Collector, via a serial connection with the station, gathers the number of received beacons by the station during the current association period. The Pinger is used to ping the station. AP1, Sniffer, and Pinger report their statistical information to the Collector via Ethernet connections.

For the results in §6.1 and §6.2, in addition to the IoT station under test, there are 12 other IoT stations (such as cameras, thermostat, and Amazon Echos/Dots) associated with the AP, as Figure 2 shows.

6.1 Probing or Keep Alive Packets?

Associated stations need to periodically communicate with the AP to renew their *inactivity timer* maintained by the AP. Referring to `hostapd` operation, if a station does not communicate with the AP for `ap_max_inactivity` duration, the AP either disassociates the station immediately, or sends a poll frame to the station and maintains the association if a response is received. This configuration is available via the `skip_inactivity_poll` flag.

In the first experiment, we evaluate the effect of listen interval on maintaining association. Since modifying the duration of `ap_max_inactivity` on commercial APs is not possible, we use its default value of 300 seconds in our experiments. The `skip_inactivity_poll` is disabled. The first and second row of Figure 13 present the results when using $\tau = 1$ and $\tau = 20$, respectively.

Figures 13(a)-(c) show that the AP can successfully poll the station every 300 seconds until 5000 seconds. At this point, as we can observe in Figure 13(b), the number of beacon packets that include the AID of the station is suddenly increased to 9. A closer look revealed an interesting

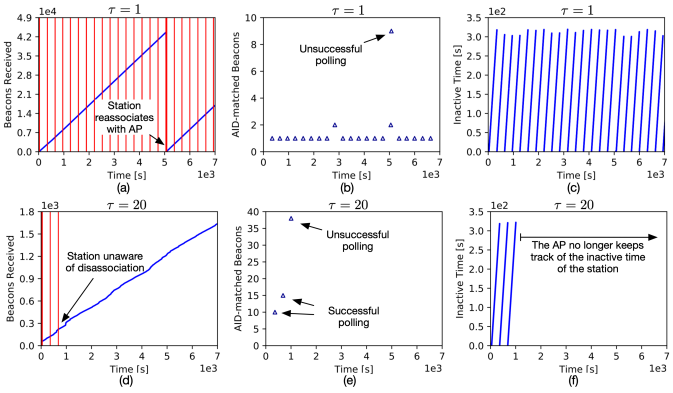

Fig. 13: AP-based polling and the impact of listen interval. The listen interval (τ) in sub-figures (a)-(c) is 1. The listen interval (τ) in sub-figures (d)-(f) is 20. (a) and (d): Number of beacons received by the station (blue curves) as well as the communications between the AP and station (red bars). (b) and (e): Number of AID-matched beacons (indicating buffered packets for the station) per 30 seconds intervals. (c) and (f): Inactivity time of the station from the AP point of view.

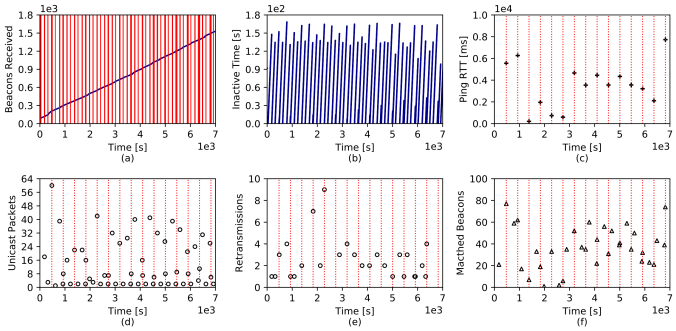

Fig. 14: Maintaining association by using station-side keep-alive packets. Listen interval coefficient (τ) is 20. (a) Number of beacons received by the station (blue curve) as well as the communication instances between the AP and station (red bars). (b) Inactivity time of the station from the AP point of view. (c) Round-Trip Time (RTT) of pinging. (d) Number of unicast packets exchanged between the station and AP per 30-second intervals. (e) Number of retransmissions between the station and AP per 30-second intervals. (f) Number of AID-matched beacons sent by the AP per 30-second intervals. The dotted lines in (c), (d), (e), and (f) indicate ping transmission instances.

behavior. After the first beacon packet, the station needs to send two PS-Poll packets to receive an ACK from the AP. The AP then sends six Null packets to the station, but no ACK is received; then, the collision avoidance mechanism (RTS/CTS) is used to communicate with the station, and 20 RTS packets are sent. Meanwhile, `hostapd`'s probing timer expires and the station is disassociated by the AP. Fortunately, link reliability improves shortly and the disassociation packet is conveyed to the station after the next beacon instance; thereby, since the station is informed about the disassociation event, it reassociates with the AP. This scenario clearly shows the impact of link unreliability and delayed station wake up on disassociation. Therefore, even when the listen interval matches the beacon period, the internal timer of `hostapd` may expire before the station is woken up to reply to the probe message.

Figures 13(d)-(f) present the same experiment with τ = 20. Two polling events at 300 and 600 seconds are performed successfully. The third polling, however, fails because the AP does not hear back PS-Poll from the station. We observed that AP sends 35 beacons that include the station's AID. Since no response is received, the AP infers that the station is no longer in range; thereby, the AP does not send a disassociation packet to the station. Therefore, as Figure 13(d) shows the trend of receiving beacons from the AP, *from the station's point of view, the station is still connected to the AP.* A *disassociation-unaware station* does not make any attempt to re-associate with the AP as long as it receives beacon packets. For event-based applications that trigger uplink traffic (e.g., video streaming when motion is detected, reporting temperature when a threshold is passed), this behavior may require the station to re-associate whenever inter-event interval is longer than 300 seconds. For applications that require downlink communication with stations, the disassociation from the AP prevents the user or cloud platform from communicating with the station. Based on these discussions, we conclude that *maintaining association by relying on the AP to send probes is unreliable because the station may be left unaware of disassociation event.*

Keep-Alive Packets. The above problem with AP-triggered probing can be eliminated by using station-generated keep-alive packets. A keep-alive message can be a UDP, TCP, Null or ARP packet. Normally, such packets are generated by the protocol stack running on the application subsystem. However, to reduce the burden of packet generation and reduce energy consumption, we offload keep-alive packet generation to the wireless subsystem; thereby avoiding the need to wake up the application processor. To validate the effectiveness of transceiver-generated keep-alive packets, we utilized driver APIs to set up Null packet generation. Also, to verify successful downlink packet delivery, the Pinger (see Figure 2) pings the station every 450 seconds. Figure 14 shows the results for $\tau = 20$. In terms of connectivity, Figures 14(a) and (b) confirm the reliability of maintaining association by the station and AP. The inactivity timer kept by the AP for the station is renewed every 150 seconds, which corresponds to a Null keep-alive packet or the ping packet. Figure 14(c) confirms that ping packets are always delivered to the station, although some RTT instances are as long as 8 seconds. Comparing Figures 14(c) and (f) reveals the relationship between the number of AID-matched beacons and ping delay. Specifically, the main cause of communication delay is the number of beacons sent by the AP until a successful packet exchange is achieved. The number of beacons sent by the AP to wake up the station is, on average, 36. The impact of packet loss can be observed in Figures 14(d) and (e). Figure 14(d) shows that packet retransmissions are necessary for both pinging and keep-alive delivery.

6.2 Key Renewal

In the previous sections, we showed the importance of periodic communication with AP to renew its inactivity timer. In this section, we demonstrate that key renewal is also a time-critical operation and may result in disassociation.

With both WPA2 and WPA3, each station is assigned two keys: a PTK for unicast communication, and a GTK for multicast and broadcast communication. Both keys are periodically renewed based on the timer values defined in

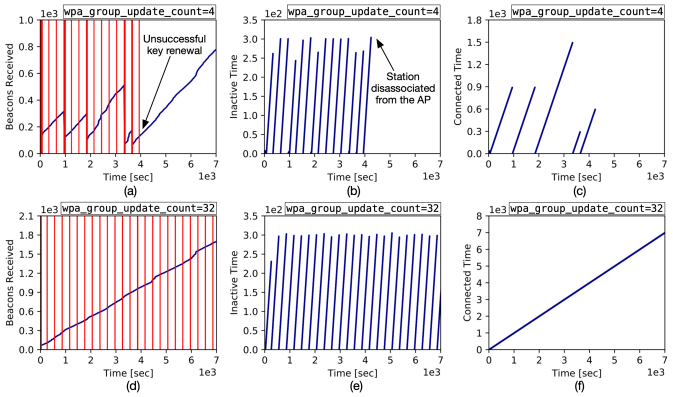

Fig. 15: Impact of key renewal on association durability. Listen interval coefficient (τ) is 20. In both sub-figures (i) and (ii): (a) Number of beacons received by the station (blue lines) as well as the communications between the AP and station (red bars). (b) Inactivity time of the station from the AP point of view. (c) Connected time of the station from the AP point of view.

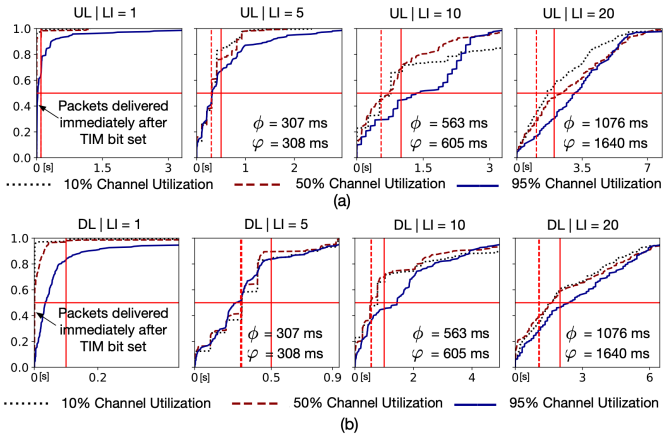

Fig. 16: ECDF of wake up delay for PSM. The vertical red line represents listen interval (τ) and the horizontal red line represents the 50th percentile. The notation ϕ denotes the theoretical expected wake up delay and the notation φ is the 50th percentile of empirical wake-up delay for 10% channel utilization.

`hostapd`'s configuration. The GTK is also renewed whenever a station leaves the network. This renewal is necessary to prevent the station from receiving multicast or broadcast packets of the network that the station no longer belongs to. This means mobility and disassociation of any station may require the AP to communicate with all stations.

In the experiments of this section, we use Smartphone1 (see Figure 2) to trigger GTK renewal. This smartphone has been programmed to leave the network every 450 seconds and then reassociate after 3 seconds. The first row of Figure 15 shows the results with the default configuration of `hostapd`. As it can be observed, the GTK renewal is performed successfully until around 4000 seconds. At this time, due to link unreliability, the AP fails to renew the key and disassociates the station. However, since the station does not receive the disassociation packet, the result is a *disassociation-unaware station*.

Customized hostapd configuration. To address the aforementioned problem, we modify `hostapd`'s configuration to increase the number of key renewal retries. To leverage this feature, we increase the value of `wpa_group_update_count` to 32, compared to its default value of 4. The results in the second row of Figure 15 confirm that with the new value, the AP can successfully update the key, even in the presence of long listen interval ($\tau = 20$). The main takeaway is that *key renewal may result in the disassociation of IoT stations when the listen interval is long or when the communication link is unreliable. And fixing this problem requires customizing the configuration values of* `hostapd`.

7 Wake up Delay

Ensuring reliable and short station wake up delay prevents cases such as polling failure (§6.1) and key renewal failure (§6.2). Such assurance is also important for applications where timely communication with stations is desired; for example, when sending an actuation command to a station or waking up a surveillance camera by the user to start video streaming. In this section, we consider the PSM and APSD power saving-methods and measure the wake up

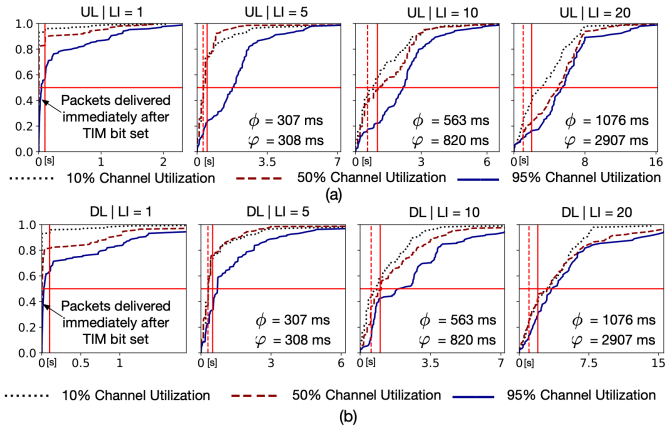

Fig. 17: ECDF of wake up delay for APSD. The vertical red line represents listen interval (τ) and the horizontal red line represents the 50th percentile. The notation ϕ denotes the theoretical expected wake up delay and the notation φ is the 50th percentile of empirical wake-up delay for 10% channel utilization.

delay of stations. Using PSM, when the station receives a beacon including its AID, it sends a PS-Poll packet to the AP to trigger downlink delivery; whereas, using APSD, a Null packet is sent to the AP. The Pinger (§3) sends a ping packet to the CYW station every t seconds, where t is uniformly chosen from the range $[1, 15]$ seconds. We measure wake up delay as the interval between the transmission of first beacon including the station's AID until the transmission of a PS-Poll or Null packet by the station. Figures 16 and 17 present the results for PSM and APSD, respectively.

The time instances the AP receives the ping packet from the Pinger follow a continuous uniform distribution. For a station with listen interval τ, we denote the beacon instances during the listen interval as $[t_k, ..., t_{k+\tau}]$. When a ping packet arrives at the AP during interval $[t_k, t_{k+1})$, the AP needs to send τ beacon packets before the next wake-up instance of the station. Similarly, if the packet arrives at the AP during interval $[t_{k+1}, t_{k+2})$, the AP needs to send $\tau - 1$ beacon packets. Therefore, the expected number of beacons sent until station wake-up is $\frac{1}{\tau}(\tau + (\tau - 1) + ... + 2 + 1) =$

$(\tau+1)/2$, when $\tau > 1$. The expected wake up delay (ϕ), demonstrated by the dashed vertical lines in Figures 16 and 17, is computed as $102.4 \times (\tau+1)/2$ ms, for $\tau > 1$. The maximum wake up delay is $\tau \times 102.4$ ms, demonstrated by the solid vertical lines in Figures 16 and 17. As the results show, the observed values of median wake up delay (φ) increase beyond the theoretical expected values (ϕ) in the presence of background traffic and also when $\tau > 5$. For example, consider using APSD and UL traffic. For $\tau = 10$, the expected (ϕ) and maximum delays are 563 ms and 1024 ms, respectively; however, the empirical results show median values (φ) of 820 ms and 2162 ms for 10% and 95% traffic, respectively. We observed a similar behavior in Figure 14, where, although the expected and maximum number of beacons are 10.5 and 20, the average number of beacons sent was 36. These prolonged wake up delays are caused by missing and delayed transmission of beacon, PS-Poll, and Null packets. Assuming the probability of successful beacon reception is p, the expected value of the number of beacons required to achieve success is $\tau \times 1/p$ when listen interval τ is enforced. Assuming uplink success probability q, the expected value of the number of packet transmissions (beacon and PS-Poll/Null) adds up to $\tau \times 1/p + 1/q$.

For both power-saving methods, and compared to DL traffic, we observe UL traffic has a considerably higher effect on prolonging wake up delay. For example, for 95% channel utilization, the 90th percentile wake up delay of PSM with $\tau = 5$ increases from 775 ms to 1437 ms when changing the traffic type from DL to UL. Aligned with our observations in Figure 3, with DL traffic, the packets sent by AP1 cause collision with the PS-Poll or Null packet sent by the station. Whereas, in the UL scenario, both the beacons sent by AP1 *and* the PS-Poll or Null packet sent by the station are susceptible to collide with Smartphone1's traffic.

In all the results, tail is increased as channel utilization intensifies. Also, we observe the higher effect of channel utilization on prolonging APSD wake up delay, compared to PSM. For example, for $\tau = 10$ and 10% DL channel utilization, the 50th percentile wake up delay increases by 27% when switching from PSM to APSD; and this increase is almost 2x for 95% DL channel utilization. Since the priority of beacon packets is not affected by the power-saving method used, we looked into the type and priority of the packet sent by the station in response to an AID-matched beacon. We noted that PS-Poll is categorized as a management packet, whereas, Null packets are data packets. Specifically, PS-Poll packets are always sent using the base rate, and Null packets are transmitted at the normal data rates. For example, in our results, the rate of PS-Poll packet was 6 Mbps, while the rate of Null packet was at least 24 Mbps. Also, management packets are prioritized over all data packets in the driver's software-queues, and these packets are transmitted alongside voice AC packets in the driver's hardware queues. Although Null data packets are prioritized over packets belonging to video, best-effort, and background ACs, these packets are treated as regular voice AC packets, and thereby, contend with other voice packets.

The key takeaways are: *Wake-up delays are lower with PSM compared to APSD, in the presence of concurrent traffic. However, PSM requires sending a PS-Poll packet for retrieving each packet from the AP, whereas, APSD utilizes only a single Null frame to retrieve all the queued packets. Hence, the overall energy consumption when using PSM can still be higher when more than one packet are queued for downlink delivery at the AP during each listen interval, on average. Also, the effect of DL traffic on wake-up delay is lower than that of UL traffic.*

8 Related Work

The suitability of using WiFi for IoT and industrial applications has been studied in recent works. Luvisotto et al. [6] discuss the salient features of WiFi's physical layer to address the requirements of industrial communication systems, compared to cellular and 802.15 networks. Similarly, Tramarin et al. [7] discussed the suitability of 802.11n in industrial applications. Abedi et al. [3] report the physical layer energy consumption range of WiFi and BLE are 10-100 nJ/bit and 275-300 nJ/bit, respectively; thereby confirming the higher efficiency of WiFi.

Seneviratne et al. [12] present the loss of DHCP packets as the main cause of association failure and delay. Pei et al. [11] conducted a large-scale empirical measurement and revealed that up to 45% of the users experience association failure, and 15% of successful associations are longer than five seconds. They also demonstrate that the scan process comprises about 47% of the association duration. Tozlu et al. [25] evaluate the impact of various configurations on the energy efficiency of WiFi stations, and show that WPA2/AES-PSK achieves the highest security-performance tradeoff. Montori et al. [26] studied the association time of three authentication schemes, namely, Open, WEP, and WPA2, and measured their effect on the discharge rate of three battery types. Abedi et al. [3] show that the overall energy efficiency of WiFi is lower than BLE due to the cost of association and maintaining association. They propose Wi-LE, which eliminates these costs by placing data in beacon packets. Chen and Qiao [27] propose a handoff mechanism to eliminate the need to dwell on each scanned channel to receive the response messages. Once a station sends a probe message, it switches back to its current channel and delegates the task of collecting information from nearby APs to the currently-associated AP. None of the existing works present a detailed study of the various constituents of association process considering different hardware and software components and variable traffic loads. Also, WPA3 has been neglected in existing studies and none of them utilize PMK offloading or caching to reduce association costs.

Vasudevan et al. [28] propose a passive mechanism to measure the potential bandwidth of APs. Assuming beacon packet transmissions are not prioritized, stations can compute the buffering delay of nearby APs by monitoring their beaconing delay. Their proposed solution assumes the stations are always listening to the beacon packets and leverage beacon transmission delays to associate with the lowest-delay AP nearby. Molina et al. [29] evaluated beacon transmission jitter in high and normal channel utilization scenarios. Their results show the inter-quartile range (IQR) of beacon jitter increases monotonically versus channel load. Stations can detect channel saturation by using the Kolgomorov-Smirnov (KS) test and comparing beacon jitters

with a reference distribution representing the non-saturated scenario. In sum, the existing works did not study the adverse effects of channel load and beacon delay on various station types that leverage non-standard listen intervals ($\tau > 1$) to enhance their energy efficiency.

Sheth and Dezfouli [1] proposed a packet scheduling mechanism to reduce the idle time of stations implementing Adaptive PSM. Peck et al. [30] state that either high delay or high energy is acceptable from a user's point of view. Based on the variations of station-server communication delay, the sleep duration is dynamically adjusted to satisfy the desired trade-off. Jang et al. [31] proposed an adaptive tail time adjustment mechanism by measuring inter-packet intervals. Once a stream of packets arrives at a station, inter-packet arrival delay is predicted using a moving average. Primarily designed for VoIP traffic, Liu et al. [32] propose a mechanism to reduce contention among stations using APSD to request packet delivery from the AP. Chen et al. [33] proposed M-PSM for scenarios where the load of stations is light and delay-insensitive. The ultimate goal of M-PSM is to prioritize communication during intervals that the link quality to the AP allows utilizing higher communication rates. A per-station DTIM allocation mechanism has been proposed in [10]. Instead of maintaining one DTIM for all the stations, the AP enables the stations to request their own desired DTIM period. Khorov et al. [34] present an overview of periodic reservation methods available in 802.11s, 802.11ad, 802.11ax, and 802.11be. They propose solutions for addressing the challenges of establishing communication reservation periods for exchanging real-time multimedia traffic between stations in the presence of interference. The aforementioned works introduce enhancements to the power saving methods of 802.11 to tailor them for various application scenarios. In contrast, in this work, we followed an application-agnostic, extendable analysis of the fundamental operation of power saving methods considering various hardware platforms and traffic scenarios.

9 Conclusion

Association, periodic beacon reception, maintaining association, and station wake-up are fundamental operations to ensure connectivity in WiFi networks. In this paper, we performed an empirical evaluation of these operations using various hardware and software platforms and identified the underlying causes of energy inefficiency, delay, and unreliability. In addition to identifying the challenges of building WiFi-based, low-power IoT systems, this research also highlighted the importance of firmware customization, AP configuration, and transceiver choice, based on the application at hand and environmental parameters.

References

[1] J. Sheth and B. Dezfouli, "Enhancing the energy-efficiency and timeliness of iot communication in wifi networks," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 9085–9097, 2019.

[2] H. Pirayesh, P. K. Sangdeh, and H. Zeng, "EE-IoT: An Energy-Efficient IoT Communication Scheme for WLANs," in *IEEE INFOCOM*, 2019, pp. 361–369.

[3] A. Abedi, O. Abari, and T. Brecht, "Wi-LE: Can WiFi Replace Bluetooth?" in *18th ACM HotNets*, 2019, pp. 117–124.

[4] J. Huang, F. Qian, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck, "A close examination of performance and power characteristics of 4G LTE networks," in *10th ACM MobiSys*. ACM, 2012, pp. 225–238.

[5] C.-C. Li, V. K. Ramanna, D. Webber, C. Hunter, T. Hack, and B. Dezfouli, "Sensifi: A wireless sensing system for ultra-high-rate applications," *arXiv preprint arXiv:2012.14635*, 2020.

[6] M. Luvisotto, Z. Pang, and D. Dzung, "Ultra high performance wireless control for critical applications: Challenges and directions," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 3, pp. 1448–1459, 2017.

[7] F. Tramarin, S. Vitturi, M. Luvisotto, and A. Zanella, "On the use of IEEE 802.11n for industrial communications," *IEEE Transactions on Industrial Informatics*, vol. 12, no. 5, pp. 1877–1886, 2015.

[8] C. Gray, R. Ayre, K. Hinton, and L. Campbell, "'smart' is not free: Energy consumption of consumer home automation systems," *IEEE Transactions on Consumer Electronics*, vol. 66, no. 1, pp. 87–95, 2019.

[9] A. Vinhas, V. Bernardo, M. Pascoal Curado, and T. Braun, "Performance analysis and comparison between legacy-PSM and U-APSD," 2013.

[10] J. Wang and T. Nagy, "Maintaining delivery traffic indication message (DTIM) periods on a per-wireless client device basis," Patent, Aug. 23, 2011, US Patent 8,005,032.

[11] C. Pei, Z. Wang, Y. Zhao, Z. Wang, Y. Meng, D. Pei, Y. Peng, W. Tang, and X. Qu, "Why it takes so long to connect to a WiFi access point," in *IEEE INFOCOM*, 2017, pp. 1–9.

[12] S. Seneviratne, A. Seneviratne, P. Mohapatra, and P.-U. Tournoux, "Characterizing WiFi connection and its impact on mobile users: practical insights," in *8th ACM WinTech*. ACM, 2013, pp. 81–88.

[13] Cypress Semiconductor. CYW43907: IEEE 802.11 a/b/g/n SoC with an Embedded Applications Processor. [Online]. Available: http://www.cypress.com/file/298236/download

[14] BCM4343W: 802.11b/g/n WLAN, Bluetooth and BLE SoC Module. [Online]. Available: https://products.avnet.com/opasdata/d120001/medias/docus/138/AES-BCM4343W-M1-G_data_sheet_v2_3.pdf

[15] Amazon Inc. (2020) FreeRTOS: Real-time operating system for microcontrollers. [Online]. Available: https://www.freertos.org

[16] Amazon Inc. (2020) FreeRTOS AWS Reference Integrations. [Online]. Available: https://github.com/aws/amazon-freertos

[17] Microsoft Inc. (2020) Azure RTOS ThreadX documentation. [Online]. Available: https://docs.microsoft.com/en-us/azure/rtos/threadx/

[18] Microsoft Inc. (2020) Azure RTOS ThreadX. [Online]. Available: https://github.com/azure-rtos/threadx

[19] Free Software Foundation Inc. (2020) LwIP: A Lightweight TCP/IP stack. [Online]. Available: https://savannah.nongnu.org/projects/lwip/

[20] Microsoft Inc. (2020) Azure RTOS NetX Duo documentation. [Online]. Available: https://docs.microsoft.com/en-us/azure/rtos/netx-duo/

[21] B. Dezfouli, I. Amirtharaj, and C.-C. Li, "EMPIOT: An energy measurement platform for wireless IoT devices," *Journal of Network and Computer Applications*, vol. 121, pp. 135–148, 2018.

[22] CYW43364 Single-Chip IEEE 802.11 b/g/n MAC/Baseband/Radio. [Online]. Available: https://www.cypress.com/file/298001/download

[23] CYW43455 Single-Chip 5G WiFi IEEE 802.11n/ac MAC/Baseband/ Radio with Integrated Bluetooth 5.0. [Online]. Available: https://www.cypress.com/file/358916/download

[24] Cisco, "Cisco visual networking index: global mobile data traffic forecast update, 2017–2022," 2019. [Online]. Available: https://s3.amazonaws.com/media.mediapost.com/uploads/CiscoForecast.pdf

[25] S. Tozlu, M. Senel, W. Mao, and A. Keshavarzian, "Wi-Fi enabled sensors for internet of things: A practical approach," *IEEE Communications Magazine*, vol. 50, no. 6, pp. 134–143, 2012.

[26] F. Montori, R. Contigiani, and L. Bedogni, "Is WiFi suitable for energy efficient IoT deployments? A performance study," in *IEEE 3rd International Forum on Research and Technologies for Society and Industry (RTSI)*. IEEE, 2017, pp. 1–5.

[27] X. Chen and D. Qiao, "Hand: Fast handoff with null dwell time for ieee 802.11 networks," in *IEEE INFOCOM*, 2010, pp. 1–9.

[28] S. Vasudevan, K. Papagiannaki, C. Diot, J. Kurose, and D. Towsley, "Facilitating access point selection in IEEE 802.11 wireless net-

works," in *5th ACM SIGCOMM IMC*. Usenix Association, 2005, pp. 26–26.
[29] L. Molina, A. Blanc, N. Montavont, and L. Simić, "Identifying channel saturation in WiFi networks via passive monitoring of IEEE 802.11 beacon jitter," in *15th ACM MobiWac*. ACM, 2017, pp. 63–70.
[30] B. Peck and D. Qiao, "A practical PSM scheme for varying server delay," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 1, pp. 303–314, 2015.
[31] S. Y. Jang, B. Shin, and D. Lee, "An adaptive tail time adjustment scheme based on inter-packet arrival time for IEEE 802.11 WLAN," in *IEEE International Conference on Communications (ICC)*. IEEE, 2016, pp. 1–6.
[32] L. Liu, X. Cao, Y. Cheng, and Z. Niu, "Energy-efficient sleep scheduling for delay-constrained applications over WLANs," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 5, pp. 2048–2058, 2014.
[33] X. Chen, S. Jin, and D. Qiao, "M-PSM: Mobility-aware power save mode for IEEE 802.11 WLANs," in *31st ICDCS*. IEEE, 2011, pp. 77–86.
[34] E. Khorov, A. Lyakhov, A. Ivanov, and I. F. Akyildiz, "Modeling of Real-Time Multimedia Streaming in Wi-Fi Networks With Periodic Reservations," *IEEE Access*, vol. 8, pp. 55 633–55 653, 2020.