Energy Consumption Anatomy of 802.11 Devices and its Implication on Modeling and Design

Andres Garcia-Saavedra*, Pablo Serrano*, Albert Banchs*†, Giuseppe Bianchi‡
{agsaaved,pablo,banchs}@it.uc3m.es, giuseppe.bianchi@uniroma2.it

*Universidad Carlos III de Madrid
Avda. Universidad, 30
28911 Leganés, Spain

†Institute IMDEA Networks
Avda. Mar Mediterráneo, 22
28918 Leganés, Spain

‡CNIT / Universita' Tor Vergata
Via del Politecnico, 1
00133 Roma, Italy

ABSTRACT

A thorough understanding of the power consumption behavior of *real world* wireless devices is of paramount importance to ground energy-efficient protocols and optimizations on realistic and accurate energy models. This paper provides an in-depth experimental investigation of the per-frame energy consumption components in 802.11 Wireless LAN devices. To the best of our knowledge, our measurements are the first to unveil that a *substantial* fraction of energy consumption, hereafter descriptively named *cross-factor*, may be ascribed to each individual frame while it *crosses* the protocol/implementation stack (OS, driver, NIC). Our findings, summarized in a convenient new energy consumption model, contrast traditional models which either neglect or amortize such energy cost component in a fixed baseline cost, and raise the alert that, in some cases, conclusions drawn using traditional energy models may be fallacious.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless Communication*

General Terms

Design, Experimentation, Measurement, Performance

Keywords

WLAN, 802.11, Energy consumption anatomy, Energy measurements, Cross-factor

1. INTRODUCTION

The increase in energy density of current state of the art (Lithium-Ion) batteries is far from following Moore's Law, the current challenge being "just" a twofold density increase in the next 10 years [37]. This is not a good technological premise behind the energy greediness of wireless connectivity, second only to that required to backlight displays in most handheld devices. Moreover, battery powered wireless devices are becoming ubiquitous, and are frequently part of the network infrastructure itself; even besides the obvious case of wireless sensor networks, battery powered relays or opportunistic intermediaries are widely considered in ad hoc, mesh, DTN scenarios, or emergency deployments.

It is hence not nearly a surprise that a *huge* research effort has been dedicated to find ways for reducing energy consumption in the wireless access and communication operation [24, 39]. For instance, with reference to the 802.11 WLAN (WiFi) technology [2], indeed the focus of this paper, energy efficiency improvements span very diverse aspects of the 802.11 operation, from management procedures [34], to usage of opportunistic relays [19] or infrastructure on demand [22], to PHY [31] and MAC [12] layer parameters' optimizations, and so on.

Obviously, a *quantitative* treatment of the attainable energy improvements is greatly simplified by the availability of realistic and accurate energy models, also considering that *fine-grained* per-frame experimental measurements (versus coarse aggregate power consumption statistics) may be non trivial to achieve. Most of the literature works, including but not limited to [7, 9, 11, 12, 15, 17, 23, 30, 40], ground their proposed analyses, optimizations, or algorithm/protocol designs, on the widely accepted paradigm that the energy toll may be ascribed to two components: a baseline one, plus a second one linear with (transmission/reception) air time. The specific weights are of course tailored to the interface state (transmit, receive, idle, sleep), and can be gathered by data sheets [13] or experimental measurements [1, 16].

Questioning the classical per-frame energy model

With such a widespread acceptance, questioning the above mentioned *classical* energy model seems tough. Actually, such model makes perfectly sense if we just focus on the network interface card consumption. But, in practice, processing in the host device drains energy as well. So, the question at stake is whether (and to what extent) there is some *energy toll in the device*, which is imputable to TX/RX processing, but which is *improperly accounted in such classical model*, e.g., because it can be neither considered (*i*) independent of the radio operation and thus (implicitly) accounted in the fixed baseline energy consumption component, nor (*ii*) strictly proportional to the traffic load in bytes, hence (implicitly) accounted in the linear air time energy cost component.

Our paper not only raises this question (apparently unnoticed in most prior work), but, more significantly, provides

Co-NEXT'12, December 10-13, 2012, Nice, France.

a (we believe) compelling answer, via extensive and tailored experiments[1] providing a detailed *anatomy* of the energy consumption in the protocol stack.

Two major findings appear to emerge. First, a *substantial* energy consumption occurs while a frame is delivered across the protocol stack, namely from the operating system to the driver to the NIC (and conversely for reception). Such "new" energy cost component, descriptively referred to as *cross-factor*, cannot be neglected; on the contrary, in some experiments it even accounts to *more than half* of the per-frame energy cost. Second, such cross-factor can be neither dealt with as an extra baseline component, nor (perhaps more surprisingly) as a cost proportional to the traffic load. Actually, this energy toll appears mostly *associated to the very fact that a frame is handled*, i.e., irrespective (to a very large extent) of the actual frame size in bytes.[2]

Our findings, which we wrap into a convenient and easy to exploit new energy model, have a twofold implication. First, they suggest new energy reduction strategies, such as *batching* packets while they travel across the protocol stack, or avoiding stack crossing when possible (the energy savings for both strategies are preliminary quantified via tailored experiments). Second, the fact that a substantial amount of energy is drained by the processing of packet units (i.e., independent of their size, or air time, or modulation and coding scheme) may play havoc with some specific optimizations proposed in the past. For instance, we show that energy-efficient optimizations leveraging relay nodes may yield *qualitatively different* conclusions when the cross-factor energy component is accounted for.

Our contribution

This paper makes the following original contributions.

Power consumption characterization. In contrast to previous works, our measurement methodology is (*i*) based on a convenient power measurement device rather than specialized hardware and complex measurement configurations; (*ii*) it exploits techniques to reduce measurement uncertainties due to scale limitation, and (*iii*) it characterizes the *total* device power consumption versus that consumed by just the wireless interface.

Power consumption anatomy and unveiling of the cross-factor. Targeted measurements devised to break down the energy cost in specific components, reveal (and quantify) that a substantial fraction of energy is consumed by the processing of packets throughout the protocol/implementation stack. Such *cross-factor* energy toll exhibits two notable features: (*i*) in some (common) radio settings, it may become the *dominant* source of energy consumption, and (*ii*) it is primarily associated to the *frame processing itself*, rather than to the amount of bytes handled. Interpreting such cost as proportional to the load seems thus intuitively appealing, and may work as long as the frame size is fixed, but is incorrect for the general case of variable frame sizes.

New energy model and relevant validation. We summarize our findings in a simple and convenient energy model which overcomes traditional models limited to NIC consumption [7, 9, 11, 12, 15, 17, 23, 30, 40]. We validate such new energy model with several experiments.

Practical implications. Focusing (for space reasons) on selected use-case examples, we show that some energy optimizations proposed in the past may yield fundamentally different conclusions when revisited with the awareness of our more realistic energy consumption findings. Of course, such conclusion may or may not apply on a case by case basis, but, on a more general line, our findings appear to raise the alert that there might be other cases where past conclusions should be reconsidered. Moreover, we discuss possible new means to take advantage of our findings for improved energy efficiency; we especially quantify, through software developed for measurement purposes, the savings introduced by two schemes that simplify the crossing of the protocol stack.

2. RELATED WORK

Energy consumption of devices. A number of previous works in the area analyze, like us, the consumption of the complete device, either a laptop [1, 3, 25] or a mobile phone [10, 33]. Some of these works deal with specific issues, such as quantification of the consumption of components other than wireless interfaces (e.g., CPU, screen, memory) [10], power consumption measurements via available APIs for estimating the battery discharge state [25], assessment of trade-off between CPU consumption due to data compression and wireless consumption due to data transmission [3], but do not tackle the per-frame energy consumption domain. Only [1] briefly mentions that the energy consumption associated to packet processing might be non negligible, but does not provide any measurement or evidence. [33] finds that message size can have a non-intuitive impact on the energy consumption, but their guess is either the existence of some power management threshold or a bug in the wireless firmware (indeed, energy bugs in mobile devices are a current concern [28]). We distinguish from all these works in the fact that *we perform a fine-grained per-packet energy consumption decomposition*, versus their energy consumption analyses on a much coarser scale.

Energy consumption of interfaces. Unlike the previous papers, most characterizations of the wireless interface consumption are done on a per-packet basis. The seminal work of [16] shows that transmission/reception of an 802.11 frame has a linear dependency on its length. This result is caused by the four different states a wireless NIC can be in, namely: sleep, idle, receiving and transmitting. [16] also identifies a fixed cost per frame, caused by control frames (e.g., RTS/CTS). The results are extended in [14] for different modulation and coding schemes and transmission power configurations, and a similar approach is followed in a recent work [18] for the case of 802.11n. While in these cases the 802.11 interface is treated as a whole, [32] distinguishes between the (approximately constant) Application-Specific Integrated Circuit (ASIC) consumption, and the Power Amplifier (PA) consumption occurring only outside idle periods. None of these works analyze the energy consumption of a frame as it is delivered to/from the NIC.

Energy consumption models. The (implicit or explicit) assumption of all previous energy consumption models [7,9,11,12,15,17,23,30,40] is that the PA operations dominate the consumption of the whole device, which allows to model consumption with a finite number of states, e.g., {ac-

[1]Primarily on a Soekris net4826-48 device, but the general findings are duly confirmed by further measurements on two other platforms, an Alix2d2 and a Linksys WRT54GL.

[2]While some previous works had already identified a per-frame energy cost, such cost was ascribed to different factors from the ones we find in this paper (like e.g. control frames).

tive, idle} [9, 11], {transmission, reception, idle} [15, 17, 40], and so on. More specifically, the common approach followed by all these papers (as well as that recently included in the NS3 network simulator [41]) is to model the NIC consumption using data sheet parameters [13], and add to this a fixed amount to account for the non-wireless power consumption of the device. In [30], the authors propose an extended model that accounts for the power conversion efficiency of the PA, but eventually the model suffers from the same limitations. As we will see in this paper, these energy consumption models *fail to capture crucial aspects of how energy is consumed in real world devices*, and therefore their use might bias conclusions.

Energy efficient mechanisms. Proposals for energy-efficient operation can be found at practically all layers of the 802.11 stack. Starting from the lowest layer, [31] precomputes the optimal rate-power configuration for each data frame. Several works aim at reducing the energy wastage in the WLAN by adapting the contention parameters [12, 17] or extending the backoff operation [7, 23]. The use of cooperative relaying for energy efficiency is analyzed in [19]; [27] exploits idle period predictions to switch from active to sleep states. Increasing the sleep state time is the main energy saving target in the standard Power Saving Mechanism (PSM) [39], and in traffic management (and shaping) schemes such as NAPman [34], or in 'infrastructure on demand' schemes [22] devised to (de)activate Access Points based on client load. All these proposals are based either on (*i*) the energy consumption of the PA, which might be detailed but underestimates the consumption of the complete device, or on (*ii*) the coarse-grained estimated consumption of the complete device, which precludes a thorough understanding of the per-packet delivery implications.

3. METHODOLOGY TO MEASURE ENERGY

802.11 devices

For development convenience, most results have been obtained using a **Soekris net4826-48** device, equipped with an Atheros AR5414-based 802.11a/b/g Mini-PCI card, and configured to use the 802.11a PHY. The hardware comprises a 233MHz AMD Geode SC1100 CPU, 2 Mini-PCI sockets, 128 Mbyte SDRAM and 256 Mbyte compact flash circuits for data storage, extended with a 2 GB USB drive. The OS is a Gentoo 10.0 Linux (kernel 2.6.24), and the driver is MadWifi v0.9.4.

Two additional platforms, employing different WLAN PHY and bands, and different hardware architectures, have been further used to verify the most crucial findings (e.g., Fig. 1), to remove the doubt that such findings could be biased by the specifically chosen reference device or WLAN band/card/PHY. These two additional platforms are: (*i*) an **Alix2d2** device, equipped with a Broadcom BCM4319 802.11b/g Mini-PCI card, 500 MHz AMD Geode LX800, 256 MByte SDRAM, kernel 2.6.29, and (*ii*) a **Linksys WRT54GL** device, equipped with an integrated Broadcom BCM4320 802.11b/g chipset, 200Mhz BCM5352 CPU, 16 MByte RAM, kernel 2.6.32.

To generate traffic, we used the `mgen` tool[3] to send UDP packets. Additional devices in monitor mode have been employed to sniff all traffic to confirm that all wireless activity was caused only by our experiments. To ensure that no packets were dropped at any layer of the protocol stack, which may bias the conclusions of the results, we checked that the information logged by the system at the different layers matched the actual wireless transmissions that we observed through the external sniffer.

[3] `http://cs.itd.nrl.navy.mil/work/mgen/`

Power-related issues

Power consumption was measured via a low-cost PCE PA-6000 power meter,[4] which provides instantaneous values of current voltage and power factor (among other parameters), at a sample rate above 3 sample/second. This instrument can be connected in series between an AC or DC power source and the device under study, without *dismantling* it, as for instance needed with some specialized equipments, which thus may restrict experimentation to, e.g., devices using card extenders.

For what concerns powering the devices while gathering measurements, we extensively tested two alternatives: via AC supply, and via DC supply. At last (discussion omitted for space reasons), we resorted to the second configuration, to prevent periodic AC power fluctuations from the wall socket which would have affected accuracy. The PCE PA-6000 power meter was thus powered with 6 AA batteries, and we employed a Protek 3033B device[5] to power the wireless device.

3.1 Improving measurements accuracy

Power measurements were obtained by measuring the voltage v and the current i, and taking the relevant product $p = v \cdot i$. Reducing the native inaccuracies of the measurement instrument employed was a major practical challenge. Indeed, according to the vendors' specification sheet, the PA-6000 provides a resolution of $\Delta v = 0.1\ V$ for the voltage and $\Delta i = 0.01\ A$ for the current. Considering a typical baseline power measurement for the considered device, these inaccuracies yield the following relative errors:

$$v = 12.5 \pm 0.1\ V\ = 12.5\ V\ \pm 0.8\%$$
$$i = 0.20 \pm 0.01\ A = 0.20\ A\ \pm 5\%$$
$$p \approx 2.5 \pm 0.145\ W = 2.5\ W\ \pm 5.8\%$$

where in the last power measurement expression we have made usage of the well known fact that the relative error for the product $p = v \cdot i$ is approximated by the sum of the relative errors for v and i [38].

Reducing uncertainty

In most of our experiments, an uncertainty in the order of more than 5% is too coarse, as it would undermine our ability to quantify small, but for our purposes extremely meaningful, trends (e.g., power consumption variations for an increased frame size). The methodology that we followed in order to improve this accuracy[6] consists in using, instead of a single device, K devices in parallel *running the same experiment* (over different non-interfering wireless channels,

[4] `http://www.industrial-needs.com/technical-data/power-analyser-PCE-PA-6000.htm`

[5] `http://www.protektest.com/ProdInfo.asp?prodId=3033B`

[6] In our case, averaging N samples does not help to reduce uncertainty, which is determined by the "reading scale". Indeed, the average would retain the same accuracy as the original samples [38].

Table 1: Accuracy improvement with K devices.

K	v (V)		i_K (A)		p_K (W)	p_d (W)
1	12.6		0.19		2.4±6.1 %	2.4 ±6.1 %
2	12.4	±0.1	0.41	±0.01	5.1±3.2 %	2.5 ±3.2 %
3	12.2		0.63		7.7±2.4 %	2.56±2.4 %
4	12.3		0.84		10.3±2.0 %	2.58±2.0 %

of course). Thus, the instrument's uncertainty on the current measurements, namely 0.01 A, now applies to the *total* current (as well as the total power, voltage being the same) drained by the K (equivalent) devices, yielding a relative error reduction of a factor K. The power consumed by a single device is finally computed as $1/K$th of the total power, with the same (reduced) relative error (division by a constant does not affect the relative error [38]).

Table 1 shows measurements taken over 30 seconds on $K = 1$ to 4 devices in parallel, for the case of devices without wireless interfaces, which is the configuration that consumes the least power and therefore has the largest relative errors. The table reports measured voltage v, total current i_K, total consumed power p_K with associated uncertainty, and per-device consumed power p_d with associated uncertainty. With $K = 4$, accuracy improves from about 6% of single device measurements to a more satisfactory 2%.[7]

In the rest of the paper we use such a *parallel device* methodology in which each experiment is conducted with K different devices of the same type (software and hardware) performing the same operations. Unless otherwise specified, we will use $K = 4$ for 802.11a and $K = 3$ for 802.11g.[8] As shown by the results of the next section, these K values provide sufficient accuracy to analyze the behavior of the different energy components of 802.11 devices.

4. ENERGY CONSUMPTION ANATOMY

In order to characterize the power consumption of the 802.11 devices, we have conducted an in-depth experimental investigation of the considered 802.11 devices. For space reasons, all the measurements presented here are for the devices operating under the infrastructure mode; however, we verified that the devices show a very similar behavior when operating under the ad-hoc and monitor modes.

A pre-requirement for characterization of 802.11 devices consists in quantifying their "baseline" power consumption, i.e., when the devices neither send nor receive traffic. Table 2 reports measurements for the Soekris platform in three "baseline" configurations. Note that plugging the wireless card ("WiFi off") increases consumption by 0.29 W (+12.6%), whereas loading the driver and associating to an AP ("Idle") further increases the consumption by 0.98 W, indeed an extra 25% increment. The power consumed in the "Idle" state, named ρ_{id}, will be used as baseline reference in what follows.

4.1 Understanding transmission costs

Results in this section aim at characterizing the energy cost of transmissions, and providing our best effort to accurately explain and justify the relevant findings. For this reason, in the remainder of this section, results are obtained for unicast *unacknowledged* frames, so as to avoid biasing results with the cost of ACK reception (separately quantified later on). ACKs have been disabled by setting the `noack-policy` bit of the WMM parameters for the Access Point parameter set: this introduces an Information Element in beacon frames that prevents associated stations from replying with ACKs (confirmed by sniffed traces). Unless otherwise specified, each result is obtained by measuring the power consumption over a 20 seconds experiment.

Table 2: Soekris Baseline consumption profile

Config.	Description	Cons. (W)
w/o card	no NIC connected	2.29 ± 2.2%
WiFi off	NIC connected driver not loaded	2.58 ± 2.0% (+0.29)
Idle (ρ_{id})	NIC activated+associated to AP no RX/TX besides beacons	3.56 ± 1.7% (+0.98)

Transmission power consumption patterns

A large number of *total* device power consumption measurements have been carried out, spanning several combinations of four quantities/parameters: (*i*) frame size L in the range 100 to 1500 bytes, (*ii*) modulation and coding schemes (MCS ∈ {6, 12, 24, 48} Mbps), (*iii*) configured transmission power[9] (`txpower` ∈ {6, 9, 12, 15} dBm), and (*iv*) frame generation rate λ_g, up to 2000 frames per second (fps).

It turns out that the most insightful way to represent such results is via *power/airtime plots*, shown in Fig. 1.[10] Since these results appear crucial, we repeated them for *all the three platforms* (Soekris in Fig. 1-a, Alix in Fig. 1-b, and Linksys in Fig. 1-c), adapting when needed the parameters (for instance, the very cheap Linksys device cannot sustain a load greater than about a thousand fps). Such plots report the average power consumed by the whole device, versus the percentage τ_{tx} of channel airtime, computed as

$$\tau_{tx} = \lambda_g T_L, \tag{1}$$

where λ_g is the frame generation rate, and $T_L = T_{PLCP} + (H + L)/MCS$ is the time required to transmit a frame of size L using the modulation and coding scheme MCS, duly accounting for the Physical Layer Convergence Protocol preamble T_{PLCP}, and the MAC overhead H (MAC header plus FCS). For reference purposes, the plots also report the baseline power consumption ρ_{id} when the target device is in "Idle" state.

Besides the *quantitative* differences among the considered platforms, these plots provide compelling evidence that the total device power consumption, denoted $\mathbf{P}$, appears articulated into three main components,[11]

$$\mathbf{P} = \rho_{id} + P_{tx} + P_{xg}(\lambda_g), \tag{2}$$

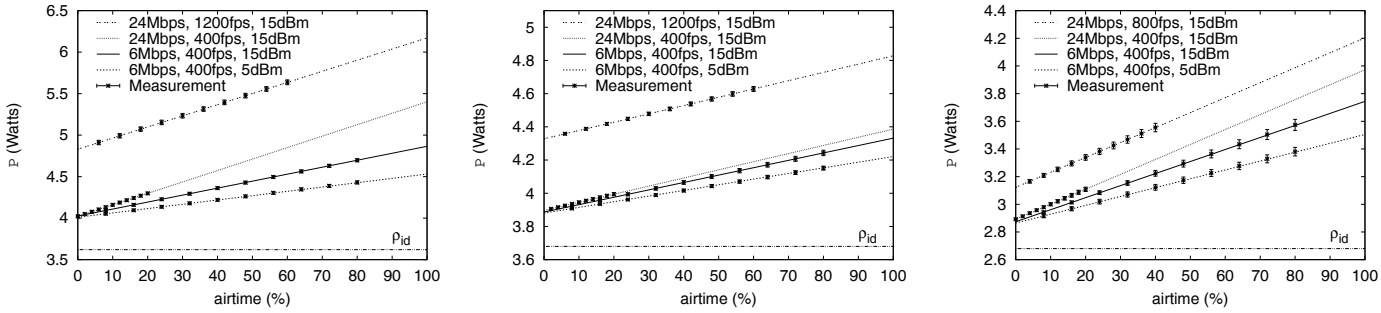
[7]From the experiments conducted in this paper, we confirmed that the maximum difference observed between our measurements and the proposed model matches the measurement inaccuracy predicted, which validates the results of this section.

[8]The reason for using $K = 3$ in the latter case is that 802.11g only allows 3 non-interfering channels.

[9]We have selected four values within the range of allowed transmission power values, which goes from 5 to 15 dBm.

[10]The values shown in the figures are the result of applying a simple linear regression to the measurements and computing their standard asymptotic error [26].

[11]The good match between the experimental figures and equation 2 is confirmed in Fig. 1, in which the values predicted by the equation are plotted using lines.



(a) Soekris **(b) Alix** **(c) Linksys**

Figure 1: Total power consumed by (unacknowledged) transmissions vs. airtime percentage τ_{tx}.

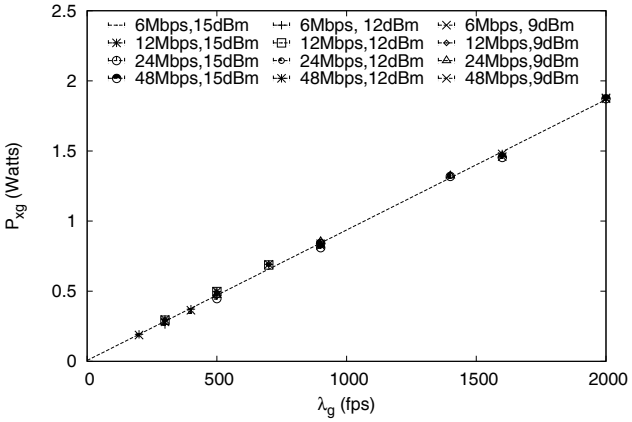


Figure 2: Relation between $P_{xg}(\lambda_g)$ and λ_g.

where:

- The first component, ρ_{id}, is the (platform-specific) baseline power consumption;
- The second component, P_{tx}, is the classical one in traditional energy consumption models, which linearly grows with the airtime percentage τ_{tx}, i.e., $P_{tx} = \rho_{tx}\tau_{tx}$. The slope ρ_{tx} depends on the target platform and on the radio transmission parameters MCS and `txpower`: the greater the MCS and/or the `txpower`, the greater the slope;
- The third component, $P_{xg}(\lambda_g)$, accounts for the fact that the above linear trend does *not* start from the baseline power consumption level ρ_{id}, but rather *starts from a relatively large positive offset* (e.g., in the Soekris case, +12% and +35% increment over the baseline level ρ_{id} for 400 and 1200 fps, respectively); offset which is *not accounted by classical energy models* [7, 9, 11, 12, 15, 17, 23, 30, 40]. Moreover, Fig. 1 suggests that such component depends only on the frame generation rate λ_g.

Per-frame processing toll

To more closely investigate the nature of such emerging power consumption offset $P_{xg}(\lambda_g)$, Fig. 2 plots its value obtained from several measurements taken for different configuration of the NIC parameters (MCS, `txpower`) over the Soekris platform (results are qualitatively analogous for the other two platforms). The plot clearly shows that $P_{xg}(\lambda_g)$ is *proportional* to the frame generation rate λ_g, whereas it is practically *independent of* the frame size or the radio settings.

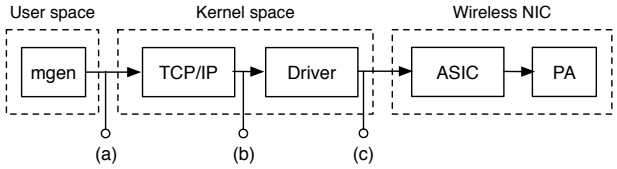


Figure 3: Interfaces and modules crossed during transmission.

Thus, if we denote with $\gamma_{xg} = P_{xg}(\lambda_g)/\lambda_g$ the proportionality constant, it appears that γ_{xg} is the *enery toll associated to the processing of each individual frame*, irrespective of its size or radio transmission parameters. Note that this energy toll is *not* associated to protocol operations such as RTS/CTS or ACKs, indeed disabled in such experiments.

Cross-factor

To grasp deeper insights on the reasons behind such a per-frame energy toll γ_{xg}, (named *cross-factor*, for reasons that will become clear throughout this section), we have engineered tailored measurements on the Soekris platform, devised to *quantify* how this energy toll splits across the frame processing chain along the protocol-implementation stack (roughly) depicted in Fig. 3.

Specifically, we have run three sets of experiments, where we discard packets at a given level of the stack and we measure the relevant power consumed up to that level:

- **App.** - packets are regularly generated by `mgen`, but are discarded before being delivered to the OS, i.e., at the mark (a) in Fig. 3, by sending them to the "sink device" (`/dev/null`);
- **TCP/IP** - packets are discarded at the bottom of the TCP/IP stack (mark (b) in Fig. 3), by deactivating the ARP lookup function, so that the device cannot retrieve the MAC destination from the ARP cache and therefore must drop the frame;
- **Driver** - packets are discarded after the MadWifi driver's processing (mark (c) in Fig. 3), by commenting the `hardstart` command which performs the actual delivery of the frame to the NIC.

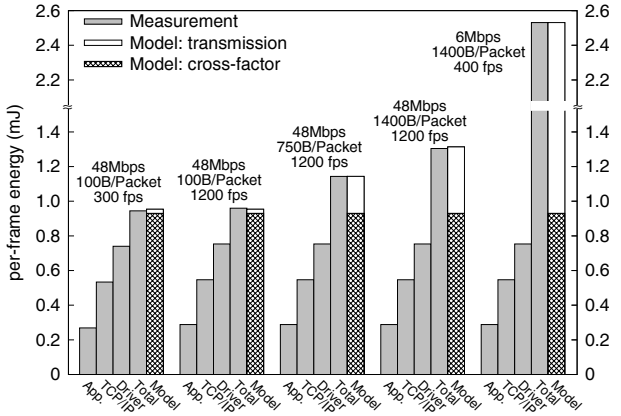


Figure 4: Per-frame energy cost in transmission.

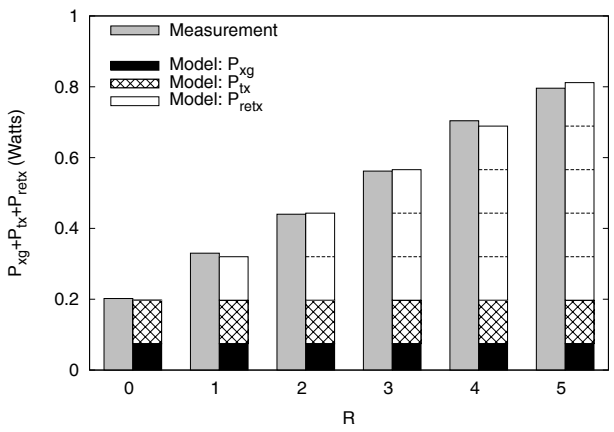


Figure 5: Impact of retransmissions on the power consumption.

Representative measurements (energy per frame) are shown in Fig. 4, along with the total energy consumption per properly transmitted frame ('Total') and the values predicted by applying equation 2 ('Model').

The figure clearly shows that the energy toll due to frame processing is practically *independent of* the frame generation rate *and* the frame size. Moreover, it shows that the energy consumed while crossing the host device stack (i.e., up to the driver included) is *substantial*, around 0.75 mJ per frame, and may become the *major* energy cost in several scenarios (e.g., short packets and/or large MCS - in essence short airtime).

Finally, even if direct measurements were not technically attainable below the driver level, Fig. 4 permits to determine that a further constant per-frame energy drain occurs at the driver-to-NIC interface level and/or below. Its quantification may be estimated by analyzing the energy consumed with very short packets and large MCS, being wireless transmission cost marginal in this case (very small airtime). Summarizing, for a Soekris device, the cross-factor coefficient amounts to about 0.93 mJ. Such per-frame processing cost appears to roughly split as follows: 24% application; 33% TCP/IP stack, 21% driver, and 22% NIC.

The above results clearly show that the energy toll is caused by the frame processing at the different layers of the protocol stack, which depends on the operating system implementation. To gain insight into the impact of the OS on the cross-factor, we evaluated the energy consumption of Soekris devices running OpenBSD. The measurements obtained confirm that the qualitative behavior with OpenBSD is the same as with Linux, and show that the cross-factor is of the same order for both operating systems (1.27 mJ for OpenBSD and 0.93 mJ for Linux).

A key result from the above is that the the energy toll is independent of the frame size. To better understand the reasons for this, we conducted some tailored experiments with applications that perform memory and CPU operations with data elements of different sizes, and observed that the resulting energy consumption is largely independent of the size of the data elements involved. This explains why the energy consumed in crossing the stack, which involves similar types of operations, is agnostic of the frame size.

Retransmissions

Intuitively, retransmissions at the MAC layer, e.g. after a failed transmission, should not be affected by the cross-factor toll. This can be verified by *provisionally assuming* that this is the case, i.e., modeling retransmission cost as purely due to the over the air transmission cost component, and then checking whether the resulting model matches experimental measurements.

Along this line, let P_{retx} be the power drained by retransmissions, and assume that

$$P_{retx} = R \cdot \rho_{tx}\tau_{tx} = R \cdot \rho_{tx}\lambda_g T_L. \quad (3)$$

where R is the number of retransmissions. Then, the *total* power consumed by packets retransmitted R times is readily obtained as the baseline component ρ_{id} plus:

$$P_{xg}(\lambda_g) + P_{tx} + P_{retx}, \quad (4)$$

where the first addendum is the per-frame processing toll (paid once), the second addendum is the power consumed by the very first transmission, and the last addendum is the extra retransmission cost as per (3). Fig. 5 compares the modeling prediction (4) with the power (additional to the baseline component ρ_{id}) consumed by a device configured to send 1400 B UDP frames generated at a rate of 80 fps to fake addresses (to prevent the reception of ACKs). The number of allotted retransmissions R (configured via the `ah_setupTxDesc` driver's descriptor) was varied from 0 to 5, and, for simplicity (i.e., to avoid the need to non trivially configure the driver so as to prevent MCS downgrade in front of persistent losses), frames were transmitted over the wireless channel using the 6 Mbps basic MCS. As shown in Fig. 5, theoretical results tightly match the experimental measurements, thus confirming that the cross-factor has (if any) negligible impact on retransmission.

4.2 Reception power consumption analysis

The analysis of the power consumption of the device while *receiving* frames is somewhat dual to that carried out in depth for the previous transmission case, hence may be dealt with much faster. We use the same configurations of MCS and `txpower` as in Section 4.1 (ACK disabled as well), with different combinations of the frame length L and frame *reception* rate λ_r. The resulting *power/airtime* plot is shown

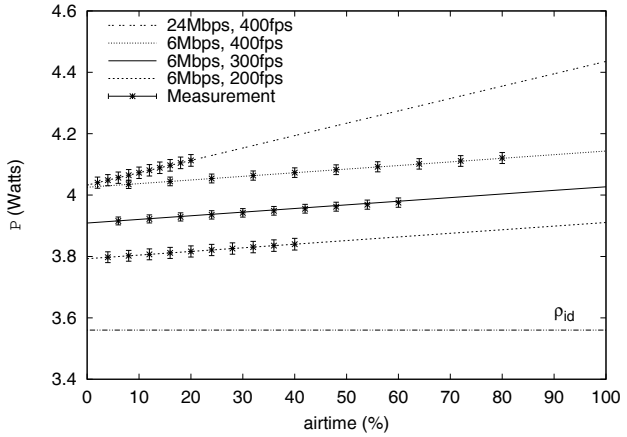


Figure 6: Power consumed by (unacknowledged) reception versus airtime.

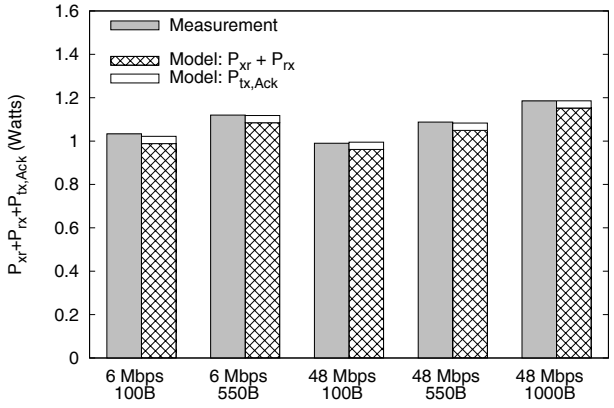


Figure 7: Impact of ACKs on reception.

in Fig. 6 (Soekris device), airtime now given by $\tau_{rx} = \lambda_r T_L$. The `txpower` parameter is not shown, as it does not affect the power consumption (as indeed well known from [14]).

Fig. 6 exhibits the same qualitative pattern found in the transmission scenario. The increment of the power consumption over ρ_{id} is composed of two components: a first one linear with the airtime and accounting for the power required to receive frames, P_{rx} (indeed in line with traditional energy models), and a second one proportional to the number of frames received and accounting for the cross-factor energy toll, $P_{xr}(\lambda_r)$. The total power consumption at the receiving side is thus:

$$\mathbf{P} = \rho_{id} + P_{rx} + P_{xr}(\lambda_r) = \rho_{id} + \rho_{rx}\tau_{rx} + \lambda_r\gamma_{xr}, \qquad (5)$$

where γ_{xr} is the cross-factor in reception, i.e., the per-packet processing toll to deliver the received frame across the protocol stack, and $\rho_{rx}\tau_{rx}$ is the traditional reception cost proportional to the airtime. Again, Fig. 6 confirms that results from the above equation (lines) closely match the experimental measurements (symbols).

4.3 Characterization of additional aspects

To complete our anatomy, it remains to characterize the additional power consumed for sending/receiving acknowledgments (both the previous sections have considered unacknowledged operation), and the power consumption experienced while overhearing a collision.

ACKs and other control frames

Since ACK frames, like retransmissions, do not have to cross the stack but are internally generated by the NIC, we assume that their power consumption can be characterized by just the cost of the relevant ACK transmission or reception. Under such assumption, the power consumed for replying with ACKs to received frames (arriving at rate λ_r) is trivially given by

$$P_{tx,Ack} = \rho_{tx}\lambda_r T_{Ack}, \qquad (6)$$

where $T_{Ack} = T_{PLCP} + ACK/MCS_C$ is the time required to transmit an ACK frame, i.e., a PLCP preamble plus the 14B ACK frame transmitted at the modulation and coding scheme MCS_C configured for control traffic. Similarly, the power consumed to receive an ACK is readily computed as

$$P_{rx,Ack} = \rho_{rx}\lambda_g T_{Ack}. \qquad (7)$$

For space reasons, we show in Fig. 7 the experimental validation for the ACK transmission case, only. Such experimental results, obtained with $\lambda_r = 1000$ fps, confirm that the measurements match the results predicted by the model, which includes the energy consumed by the reception of frames ($P_{xr} + P_{xr}$) and the transmission of the ACKs ($P_{tx,Ack}$).

We also verified that other control frames that do not cross the stack, such the RTS and CTS frames, show the same behavior and their consumption is given by the cost of the corresponding transmission or reception only (results are not shown for space reasons).

Collisions and other transmissions

At last, we analyze the impact on the energy consumption in reception when the medium is occupied by collisions or by transmissions addressed to another device (i.e., sent to another MAC address).

To analyze the energy consumed by collisions, we configured a communication between two nodes and set up another node to act as interferer. This interferer was implemented by setting the carrier sense threshold at the highest value, which practically results in no carrier sensing, and using the lowest values for the CW, $SIFS$ and MCS parameters while deactivating the use of ACKs. In order to control the amount of time the interferer was sending data (i.e., the 'interference rate'), we used the `quiet element` option to silence the interface for a given amount of time every beacon period. Prior to our measurements, we performed extensive tests using different configurations of the `txpower` parameter and varying the relative physical location of the devices, to have a configuration in which simultaneous transmissions resulted in all frames being lost (i.e., no capture effect).

We present in Table 3 the main results from our analysis, namely, (i) power consumed in reception depends exclusively on the traffic actually received (see, e.g., when the interference rate goes from 0% to 50%), and (ii) collisions have the same impact as an idle medium (e.g., the cases with 100% interference rate coincides with ρ_{id}). Based on this, we conclude that collisions have no practical impact on the energy consumption at the receiver (for the transmitter, they have already been modeled in our analysis of retransmissions).

Table 3: Impact of collisions on reception.

Sent	Int. Rate	Received	Meas.	Model
1.2 Mbps	0%	1.2 Mbps	3.67	3.68
2.4 Mbps	50%	1.2 Mbps	3.67	3.68
2.4 Mbps	100%	0 Mbps	3.56	3.56
2.4 Mbps	0%	2.4 Mbps	3.80	3.81
4.8 Mbps	50%	2.4 Mbps	3.80	3.81
4.8 Mbps	100%	0 Mbps	3.56	3.56

To analyze the impact of the transmissions addressed to another station, we configured a communication between two nodes and measured the energy consumption at a third node that was in the transmission range of this communication. We observed that the energy consumed by this node was the same as if the medium was idle, which confirms that transmissions addressed to other stations practically do not consume energy. This is in agreement with our previous results: according to (5), the energy cost of listening to the PLCP plus headers is only 38 μJ/frame (for 6 Mbps MCS), which has practically no impact on the overall consumption.

5. ENERGY CONSUMPTION MODEL

The complete model

Based on the results obtained in the previous section, we now build the *complete model* for the power consumption of 802.11 devices. Summarizing our findings of (1)–(7), we have that the power consumed by an 802.11 device consists of the following components: (*i*) the idle consumption, ρ_{id}, (*ii*) the cross-factor for the packets generated by the application, P_{xg}, (*iii*) the power required to transmit them, P_{tx}, (*vi*) the power consumed in retransmissions, P_{retx}, (*v*) the power spent in receiving frames, P_{rx}, (*vi*) the cross-factor for the received frames, P_{xr}, and (*vii*) the power spent on sending and receiving ACK frames, $P_{rx,Ack}$ and $P_{tx,Ack}$:

$$\mathbf{P} = \rho_{id} + P_{tx} + P_{xg} + P_{retx} + P_{rx} + P_{xr} + P_{rx,Ack} + P_{tx,Ack}. \quad (8)$$

By substituting the expressions obtained in the previous section for all the above components and regrouping the terms, we obtain:

$$\begin{aligned} \mathbf{P} &= \rho_{id} + \rho_{tx}(\lambda_g T_L + \lambda_g T_L R + \lambda_r T_{Ack}) \\ &+ \rho_{rx}(\lambda_r T_L + \lambda_g T_{Ack}) + \gamma_{xg}\lambda_g + \gamma_{xr}\lambda_r. \end{aligned} \quad (9)$$

By taking into account that $\lambda_g T_L + \lambda_g T_l R + \lambda_r T_{Ack}$ corresponds to the transmission airtime percentage τ_{tx}, and $\lambda_r T_L + \lambda_g T_{Ack}$ to reception airtime percentage τ_{rx}, the above equation can be rewritten as:

$$\boxed{\mathbf{P} = \rho_{id} + \rho_{tx}\tau_{tx} + \rho_{rx}\tau_{rx} + \gamma_{xg}\lambda_g + \gamma_{xr}\lambda_r.} \quad (10)$$

The above expression gives the model for the power consumption of an 802.11 device that we propose in this paper. As already mentioned in the previous section, the key difference between the above model and the 'traditional' one used in many previous works [7, 9, 11, 12, 15, 17, 23, 30, 40] is that the traditional model only includes the first three components (namely ρ_{id}, $\rho_{tx}\tau_{tx}$ and $\rho_{rx}\tau_{rx}$) while our model adds to these three components two additional ones ($\gamma_{xr}\lambda_r$ and $\gamma_{xg}\lambda_g$). As shown by our measurements, these two additional components account for a very significant portion of the power consumption, which renders the traditional model highly inaccurate.

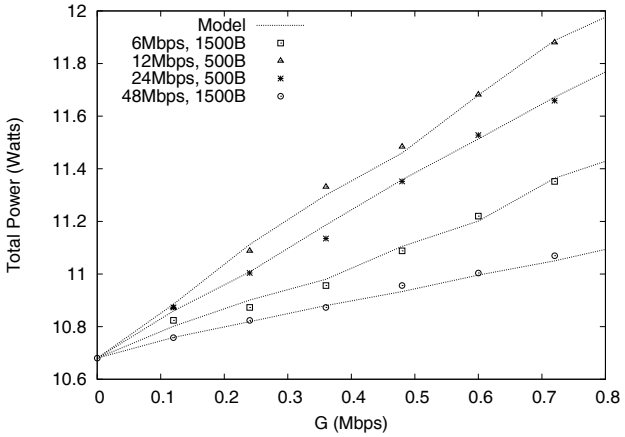


Figure 8: Model validation with multiple stations.

Out of the 9 variables in (10), 5 are constant parameters that depend on the device and the configuration of its communication parameters (ρ_{id}, ρ_{tx}, ρ_{rx}, γ_{xr} and γ_{xg}), while the other 4 parameters are variables that depend on the number of stations in the WLAN and their traffic generation behavior (τ_{tx}, τ_{rx}, λ_r and λ_g). In the following, we characterize the 5 constant parameters that determine the power consumption of the considered 802.11 devices for different values of MCS and `txpower`. To obtain these parameters, we use the expressions for the simple linear regression and the standard asymptotic error [26]. Following this, we obtain the numerical values given in Tables 4, 5 and 6 for the parameters that characterize the energy consumption of the Soekris, the Linksys and the Alix devices, respectively.

Model Validation

To validate our model in a general scenario with multiple sending and receiving stations, we consider a WLAN with one AP and three stations. Each station generates unicast traffic to the AP at a rate G, while the AP sends unicast traffic at the same rate G to each station. To apply the model of (10), we need to obtain the parameters τ_{tx}, τ_{rx}, λ_r and λ_g. These can be obtained from typical statistics recorded by the wireless driver, namely, number of generated frames (N_g), successful frames (N_{tx}), transmissions attempts (N_{at}), and received frames (N_{rx}). With these, if the experiment is run for a duration of T, λ_g and λ_r are computed as

$$\lambda_g = N_g/T, \quad \lambda_r = N_{rx}/T. \quad (11)$$

To compute τ_{tx} we account for all transmission attempts of the device plus the time spent sending the Acks, i.e.,

$$\tau_{tx} = (N_{at}T_L + N_{rx}T_{Ack})/T. \quad (12)$$

Similarly, to compute τ_{rx} we need to take into account the frames and the Acks received,

$$\tau_{rx} = (N_{rx}T_L + N_{tx}T_{ack})/T. \quad (13)$$

We compare the results given by our model against those obtained from measurements. Fig. 8 depicts these results for various combinations of L and MCS, sweeping along different traffic generation rates G in the x axis. We conclude from the figure that the proposed model is able to accurately predict the power consumption in a general scenario.

Table 4: Parametrization of the power consumption model for the Soekris device.

MCS		6 Mbps	12 Mbps	24 Mbps	48 Mbps
ρ_{rx} (W)		$0.16 \pm 8\%$	$0.27 \pm 5.6\%$	$0.6 \pm 11\%$	$1.14 \pm 3.5\%$
ρ_{tx} (W)	6 dBm	$0.52 \pm 3.1\%$	$0.55 \pm 4.6\%$	$0.81 \pm 5.3\%$	$1.2 \pm 1.6\%$
	9 dBm	$0.57 \pm 2.1\%$	$0.59 \pm 1.8\%$	$0.88 \pm 2.3\%$	$1.24 \pm 2.7\%$
	12 dBm	$0.70 \pm 1.7\%$	$0.73 \pm 2.2\%$	$1.02 \pm 2.8\%$	$1.37 \pm 3.1\%$
	15 dBm	$0.86 \pm 2.2\%$	$0.89 \pm 2.3\%$	$1.17 \pm 2.5\%$	$1.58 \pm 3.3\%$
ρ_{id} (W)	$3.56 \pm 1.7\%$	γ_{xg} (mJ)	$0.93 \pm 1.2\%$	γ_{xr} (mJ)	$0.93 \pm 2.2\%$

Table 5: Parametrization of the power consumption model for the Linksys device.

MCS		6 Mbps	12 Mbps	24 Mbps	48 Mbps
ρ_{rx} (W)		$0.19 \pm 5.3\%$	$0.29 \pm 3.4\%$	$0.53 \pm 2.3\%$	$0.74 \pm 4.4\%$
ρ_{tx} (W)	6 dBm	$0.70 \pm 1.1\%$	$0.72 \pm 2.2\%$	$0.75 \pm 2.0\%$	$0.81 \pm 3.7\%$
	9 dBm	$0.77 \pm 1.4\%$	$0.81 \pm 2.6\%$	$0.84 \pm 2.3\%$	$0.88 \pm 3.4\%$
	12 dBm	$0.84 \pm 1.2\%$	$0.85 \pm 1.5\%$	$0.92 \pm 2.4\%$	$0.99 \pm 4.0\%$
	15 dBm	$0.97 \pm 0.9\%$	$1.0 \pm 1.5\%$	$1.04 \pm 2.1\%$	$1.08 \pm 3.7\%$
ρ_{id} (W)	$2.73 \pm 0.4\%$	γ_{xg} (mJ)	$0.46 \pm 3.3\%$	γ_{xr} (mJ)	$0.43 \pm 4.2\%$

Table 6: Parametrization of the power consumption model for the Alix device.

MCS		6 Mbps	12 Mbps	24 Mbps	48 Mbps
ρ_{rx} (W)		$0.24 \pm 4.2\%$	$0.27 \pm 3.7\%$	$0.31 \pm 6.4\%$	$0.44 \pm 6.8\%$
ρ_{tx} (W)	6 dBm	$0.27 \pm 7.4\%$	$0.33 \pm 9.1\%$	$0.35 \pm 11.4\%$	$0.38 \pm 5.2\%$
	9 dBm	$0.30 \pm 6.7\%$	$0.35 \pm 8.6\%$	$0.36 \pm 11.1\%$	$0.39 \pm 5.3\%$
	12 dBm	$0.35 \pm 5.7\%$	$0.38 \pm 7.9\%$	$0.39 \pm 7.7\%$	$0.43 \pm 7.0\%$
	15 dBm	$0.4 \pm 7.5\%$	$0.44 \pm 6.8\%$	$0.45 \pm 8.9\%$	$0.46 \pm 8.7\%$
ρ_{id} (W)	$3.68 \pm 0.5\%$	γ_{xg} (mJ)	$0.11 \pm 7.6\%$	γ_{xr} (mJ)	$0.09 \pm 8.5\%$

6. IMPLICATIONS ON DESIGN

The new energy consumption insight gathered in this paper may have significant implications on the design of energy-efficient mechanisms. On the one hand, existing schemes may need to be revisited so as to properly account for the impact of the cross-factor component. Indeed, according to traditional power consumption models (i.e., only baseline component plus a toll proportional to the airtime), mechanisms yielding shorter airtimes would *surely* bring about energy gains. With the cross factor, this *might not be anymore the case*, when the power savings attained at the radio interface are paid with an increased frame handling and its associated (*non marginal*) power consumption. On the other hand, the gained knowledge that a frame crossing the stack brings about a *fixed* penalty unrelated to the frame size may be exploited to devise techniques to *avoid* or *reduce* such energy toll.

In the following, with no pretense of completeness, we present *quantitative* examples that show how our new insights may affect existing energy efficient mechanisms as well as inspire novel approaches.

6.1 Reconsidering existing schemes

Packet relaying

Packet relaying in WLANs is commonly used to improve performance [6] and energy efficiency [19]. The rationale is that the use of a relay permits shorter transmission times, which compensate the impact of the extra number of hops, introducing a net gain. However, classical energy-efficiency analyses do not balance the airtime energy saving with the energy drain introduced by the additional frame processing, a penalty which may *fundamentally affect the relevant conclusions*.

To quantitatively support this claim, we deployed a two-hop scenario comprising three nodes (sender, relay and receiver), and compared the power consumption in two different configurations (MCS chosen as in [6]): (i) traffic directly sent to the receiver (1-hop, at 6 Mbps), and (ii) relay node used (2-hops, both at 48 Mbps). Traffic is generated at a rate of $\lambda_g = 400$ fps with different frame sizes L, and packet forwarding in the relay is performed at the routing layer. In both configurations, the relay node is always active.[12]

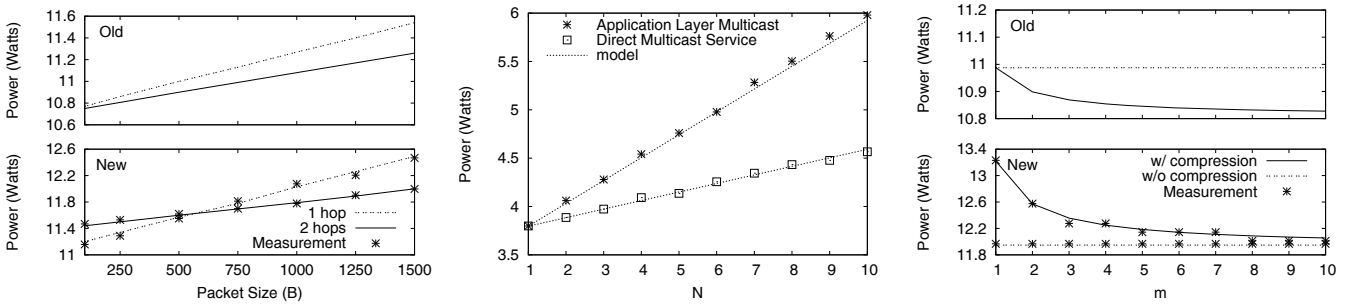
Three types of results are shown in Fig. 9a: (i) experimental measurements, (ii) theoretical predictions using a traditional model that neglects the impact of crossing the protocol stack ('old'), and (iii) predictions using the model presented in this paper ('new'), with a cross-factor of 0.8 mJ at the relay to capture the cost of forwarding a packet at the routing layer.

Not (anymore) surprisingly, results for the two models are *qualitatively* different. According to the traditional model, packet relaying always provides a gain, since the energy consumption of the 2-hops case is always smaller than that of the 1-hop case. In contrast, according to the actual measurements and our model, we only gain from using the relay when packets are sufficiently long (i.e., when the airtime cost becomes dominant over the cross-factor penalty).

Multicasting in WLAN

In order to multicast a packet stream from an AP to N stations in a WLAN, two alternatives are possible: (i) an application layer multicast (ALM) service [20], and (ii) the

[12]In most of the analyses on energy efficiency of relaying, the relay does not use the "sleep mode" (see e.g. [19, 36]).



(a) Power consumption with and without relay as a function of the frame size. **(b) Power consumption as a function of the number of multicast receivers.** **(c) Power consumed as a function of the compression factor.**

Figure 9: Revisiting previous schemes under the new model.

Direct Multicast Service (DMS), part of the 802.11aa standard [4].[13] In the first case, the application generates each frame for each destination; in the second case, the MAC layer takes care of replicating the frame for each station subscribed to the multicast group.

Both approaches generate the same traffic over the air. Thus, according to the traditional model they should consume the same energy, whereas we expect DMS to be *significantly* more energy efficient, since less frames cross the protocol stack.

Indeed, we have experimentally verified this claim by deploying both techniques in a WLAN testbed, and by measuring the relevant power consumption of the AP. The experimental settings are: $MCS = 48$ Mbps, $L = 1000$ B, $\lambda_g = 200$ fps and a varying number of stations N. Fig. 9b shows that measurements well match the model predictions.[14] More interestingly, results show that DMS can save up to 25% (1.5 W) of the total power consumption (i.e., as much as 60% of the consumption over the baseline energy cost ρ_{id}) with respect to ALM for $N = 10$.

Data compression in multi-hop networks

In wireless multi-hop networks, data compression at intermediary nodes has been proposed to reduce the information relayed to the next hop [5, 35]. According to traditional energy models, this operation *surely* saves energy, whereas our new energy consumption insights suggest that this may not be always true.

To analyze this, we used a three-node testbed consisting of a source, a sink and a relay, all using $MCS = 48$ Mbps. The source node generates 500-byte packets at 1200 fps and sends them to the relay. The relay runs an application that receives these packets, and *emulates* compression by forwarding 1 frame for every m frames received. Thus, our experiments do not capture the processing toll of the compression, and hence results reflect the best possible case for the performance of this compression scheme.

Fig. 9c shows total power consumption results (experimental ones, as well as predictions from old and new energy model), for different values of the compression ratio m, when data is compressed (and forwarded) at the application layer. These results are compared against the case where data is not compressed at the relay node but simply forwarded towards the sink at the routing layer.

As anticipated, the old model (top curve) predicts that compression is always advantageous. However, experimental results, matched by the new model predictions (bottom curve) show that data compression does not provide any gain in terms of energy consumption, not even for compression rates as high as 10. The reason is that the energy gain resulting from the data compression is outweighted by the extra cost of handling the packets at the application layer (cross-factor of 0.93 mJ for sending and 0.93 mJ for receiving) instead of the routing layer (cross-factor of 0.8 mJ for forwarding). This example thus shows that mechanisms devised on the basis of traditional energy models may not only fail to provide the expected energy gains but may even *worsen* the actual energy consumption.

6.2 Novel ways to tackle energy efficiency

Packet Batching

As emerged in our work, energy consumption across the protocol stack relates to the handling of frame *units*, and is practically independent of the frame size. This suggests a straightforward energy saving strategy: *batch packets* into bundles at the highest suitable layer for a considered scenario, deliver the bundle across the stack, thus paying the energy price associated to a *single* unit, and then restore the original frames as late as possible down the stack. Unlike previous aggregation schemes for wireless networks, this mechanism (i) does not change the packets that are actually sent, but only modifies the way they are handled *within* the device [21], and (ii) does not save energy by reducing the cumulative *tail energy* consumed as a result of lingering in high power states after completing a transmission [8, 29].

We quantified the attainable energy savings by implementing the scheme depicted in Fig. 10, which consists of (i) an "aggregator" at the application layer, which waits for n packets to generate a bundle and pass it to the TCP/IP stack, and (ii) a "de-aggregator" at the wireless driver, which splits the bundle back into the original frames. Experimental measurements are reported in Fig. 11 for 100 bytes packets, bundled up to an "aggregation factor" $n = 10$, and for various (application layer) frame generation rates λ_g. Frames are transmitted over the wireless channel at $MCS = 48$ Mbps.

[13] Both alternatives apply to traffic generated by a station as well as by a sender in the Internet; in the latter case, the traffic reaches the AP, which uses these techniques to multicast the traffic in the WLAN.

[14] The model accounts for a cross-factor of 0.75 mJ to reach the MAC and of 0.18 mJ from the MAC to the wireless card.

Figure 10: Packet batching with $n = 2$.

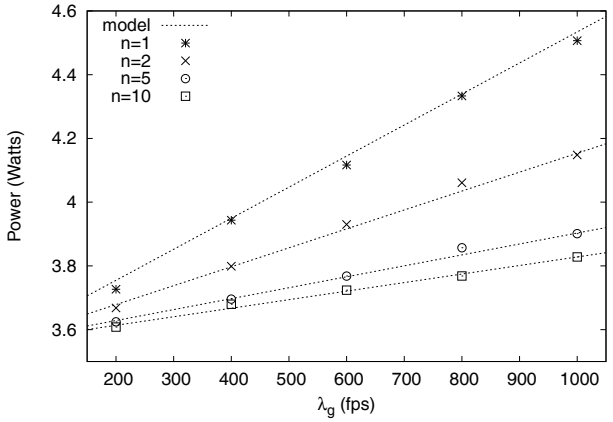


Figure 11: Energy consumption as a function of the 'aggregation factor'.

Results shown in Fig. 11 have a twofold implication. First, they provide further evidence that the cross-factor toll is practically independent of the frame size: the model matches well the measurements, and the use of an n-bundle reduces the energy toll above the driver by n. Second, energy savings are notable: with 1000 fps, an aggregation factor of 10 yields a saving of almost 0.8 W, and even the aggregation of just two packets may yield considerable savings (e.g., from 4.5 W to 4.15 W).

Obviously, casting the above described scheme into target applications (or even more general frameworks) is not straightforward,[15] and is out of the scope of this paper. Nevertheless, the above results suggest that such effort may be rewarded with notable energy saving.

Raw sockets

Since energy is consumed while crossing each layer of the protocol stack, another way to reduce energy consumption is to *skip layers* when they are not strictly necessary (e.g., in direct host-to-host wireless communication). To quantify the relevant gains, we have implemented an application that uses `raw` sockets, thus skipping the TCP/IP OS stack. Table 7 compares the power consumed using raw sockets ('`raw`') versus that consumed by using standard sockets ('UDP'), and reports the difference ('Δ'), for different configurations of L, MCS and λ_g.

Results show that the cross-factor can be reduced by approx. 0.2 mJ when skipping the TCP/IP layer, in line with the results of Fig. 4. This suggests to application developers with severe energy concerns that an extra development effort to avoid an unnecessary protocol stack is worth.

[15]Further technical problems must be dealt with, including the interaction with the TCP/IP protocol stack (e.g., if the target application requires data to be delivered as independent TCP/IP packets) and the application's requirements (e.g., the target application scenario must tolerate the extra batching delay introduced).

Table 7: Impact of using raw sockets.

MCS	L (B)	fps	Power (W) `raw`	Power (W) UDP	Power (W) Δ	Δ/fps
6 Mbps	1000	0.5k	4.26	4.35	0.09	0.19 mJ
12 Mbps	500	1k	4.50	4.69	0.19	0.19 mJ
24 Mbps	100	2k	5.05	5.48	0.43	0.21 mJ
48 Mbps	100	2k	5.03	5.44	0.41	0.20 mJ

7. CONCLUSIONS

In this paper, we have conducted a thorough measurement analysis of the power consumption of 802.11 devices that, in contrast to previous works, provides a detailed anatomy of the per-packet consumption and characterizes the total consumption of the device, and not only of its wireless interface. Our analysis is, to the best of our knowledge, the first one to reveal that a substantial fraction of energy is consumed when packets cross the protocols stack (the *cross-factor*). While other platforms than the ones analyzed (like e.g. mobile devices) may present quantitatively different results, our analysis shows that the cross-factor is likely to be substantial and cannot be neglected. Based on our findings, we have proposed a convenient energy consumption model that accurately predicts the power consumption of WLAN devices. We have shown that some schemes targeting energy efficiency may not provide the expected gains, and even worsen performance, when the cross-factor is taken into account. We have further shown some illustrative examples where the understanding gained with our analysis can be used to devise novel algorithms that save energy by reducing the cross factor, either by bundling packets, skipping parts of the protocol stack, or operating at the MAC layer. The lessons learned from these experiments provide some guidelines for applications developers pursuing energy-efficient operation in WLANs.

8. ACKNOWLEDGMENTS

This work has been supported by the European Community's Seventh Framework Programme under grant agreement no. 257263 (FLAVIA project) and by the Spanish Ministry of Economy and Competitiveness under grant agreement TEC2010-10440-E (DISTORSION project). We would like to thank F. Giust, M. Gramaglia and P. Salvador for their help in running remote experiments. We would also like to thank the shepherd and anonymous reviewers for their valuable feedback.

9. REFERENCES

[1] Power Consumption and Energy Efficiency Comparisons of WLAN Products. White paper, Atheros Comm., Apr. 2004.

[2] IEEE 802.11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, 2007.

[3] Data Transfer over Wireless LAN Power Consumption Analysis. White paper, Intel Corp., Feb. 2009.

[4] IEEE 802.11: Amendment 2. MAC Enhancements for Robust Audio Video Streaming, 2012.

[5] S. J. Baek, G. de Veciana, and X. Su. Minimizing energy consumption in large-scale sensor networks through distributed data compression and hierarchical aggregation. *IEEE J. Selected Areas Comm.*, 22(6):1130–1140, Aug. 2004.

[6] P. Bahl et al. Opportunistic use of client repeaters to improve performance of WLANs. *IEEE/ACM Trans. on Networking*, 17(4):1160–1171, Aug. 2009.

[7] V. Baiamonte and C.-F. Chiasserini. Saving energy during channel contention in 802.11 WLANs. *Mobile Networks and Appl.*, 11(2):287–296, Apr. 2006.

[8] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani. Energy consumption in mobile phones: A measurement study and implications for network applications. In *Proc. of ACM Internet Measurement Conference*, Nov. 2009.

[9] R. Bruno, M. Conti, and E. Gregori. Optimization of efficiency and energy consumption in p-persistent CSMA-based wireless LANs. *IEEE Trans. on Mobile Computing*, 1(1):10–31, Mar. 2002.

[10] A. Carroll and G. Heiser. An analysis of power consumption in a smartphone. In *Proc. of USENIX ATC*, June 2010.

[11] M. Carvalho, C. Margi, K. Obraczka, and J. Garcia-Luna-Aceves. Modeling energy consumption in single-hop IEEE 802.11 ad hoc networks. In *Proc. of ICCCN*, Oct. 2004.

[12] J.-C. Chen and K.-W. Cheng. EDCA/CA: Enhancement of IEEE 802.11e EDCA by contention adaption for energy efficiency. *IEEE Trans. on Wireless Comm.*, 7(8):2866–2870, Aug. 2008.

[13] S. Chiaravalloti, F. Idzikowski, and L. Budzisz. Power consumption of WLAN network elements. Technical report, Tech. Univ. Berlin, Aug. 2011.

[14] J.-P. Ebert, B. Burns, and A. Wolisz. A trace-based approach for determining the energy consumption of a WLAN network interface. In *Proc. of European Wireless*, Feb. 2002.

[15] M. Ergen and P. Varaiya. Decomposition of energy consumption in IEEE 802.11. In *Proc. of IEEE ICC*, June 2007.

[16] L. Feeney and M. Nilsson. Investigating the energy consumption of a wireless network interface in an ad hoc networking environment. In *Proc. of IEEE INFOCOM*, Apr. 2001.

[17] A. Garcia-Saavedra, P. Serrano, A. Banchs, and M. Hollick. Energy-efficient fair channel access for IEEE 802.11 WLANs. In *Proc. of IEEE WoWMoM*, June 2011.

[18] D. Halperin, B. Greenstein, A. Sheth, and D. Wetherall. Demystifying 802.11n power consumption. In *Proc. of HotPower*, Oct. 2010.

[19] X. He and F. Li. Throughput and energy efficiency comparison of one-hop, two-hop, virtual relay and cooperative retransmission schemes. In *Proc. of European Wireless*, Apr. 2010.

[20] M. Hosseini, D. Ahmed, S. Shirmohammadi, and N. Georganas. A survey of application-layer multicast protocols. *IEEE Comm. Surveys and Tutorials*, 9(3):58 –74, Sept. 2007.

[21] A. Jain and M. Gruteser. Benefits of packet aggregation in ad-hoc wireless network. Technical report, Univ. of Colorado at Boulder, Aug. 2003.

[22] A. P. Jardosh et al. Green WLANs: On-Demand WLAN Infrastructures. *Mobile Networks and Appl.*, 14(6):798–814, Dec. 2009.

[23] E.-S. Jung and N. H. Vaidya. An Energy Efficient MAC Protocol for Wireless LANs. In *Proc. of IEEE INFOCOM*, June 2002.

[24] G. Y. Li et al. Energy-efficient wireless communications: tutorial, survey, and open issues. *IEEE Wireless Comm.*, 18(6):28–35, Dec. 2011.

[25] E. Lochin, A. Fladenmuller, J. yves Moulin, S. Fdida, and A. Manet. Energy consumption models for ad-hoc mobile terminals. In *Proc. of Med-Hoc Net*, June 2003.

[26] J. S. Milton and J. C. Arnold. *Introduction to probability and statistics, 4th ed.* McGraw-Hill Higher Education, 2003.

[27] A. W. Min, R. Wang, J. Tsai, M. A. Ergin, and T.-Y. C. Tai. Improving energy efficiency for mobile platforms by exploiting low-power sleep states. In *Proc. of Computing Frontiers*, May 2012.

[28] A. Pathak, Y. C. Hu, and M. Zhang. Bootstrapping energy debugging on smartphones: a first look at energy bugs in mobile devices. In *Proc. of ACM HotNets*, Nov. 2011.

[29] F. Qian et al. Characterizing radio resource allocation for 3G networks. In *Proc. of ACM Internet Measurement Conference*, Nov. 2010.

[30] D. Qiao, S. Choi, A. Jain, and K. G. Shin. MiSer: an optimal low-energy transmission strategy for IEEE 802.11a/h. In *Proc. of ACM Mobicom*, Sept. 2003.

[31] D. Qiao, S. Choi, and K. Shin. Interference analysis and transmit power control in IEEE 802.11a/h wireless LANs. *IEEE/ACM Trans. on Networking*, 15(5):1007–1020, Oct. 2007.

[32] E. Rantala, A. Karppanen, S. Granlund, and P. Sarolahti. Modeling energy efficiency in wireless internet communication. In *Proc. of ACM MobiHeld*, Aug. 2009.

[33] A. Rice and S. Hay. Measuring mobile phone energy consumption for 802.11 wireless networking. *Pervasive and Mobile Computing*, 6(6):593–606, Dec. 2010.

[34] E. Rozner, V. Navda, R. Ramjee, and S. Rayanchu. NAPman: network-assisted power management for WiFi devices. In *Proc. of ACM MobiSys*, June 2010.

[35] A. B. Sharma, L. Golubchik, R. Govindan, and M. J. Neely. Dynamic data compression in multi-hop wireless networks. In *Proc. of ACM SIGMETRICS*, June 2009.

[36] C. Sun and C. Yang. Is two-way relay more energy efficient? In *Proc. of IEEE GLOBECOM*, Dec. 2011.

[37] J.-M. Tarascon. Key challenges in future Li-battery research. *Phil. Trans. of Royal Society A*, 368(1923):3227–3241, July 2010.

[38] J. R. Taylor. *An introduction to error analysis*. Oxford University Press, 1982.

[39] S.-L. Tsao and C.-H. Huang. A survey of energy efficient MAC protocols for IEEE 802.11 WLAN. *Computer Comm.*, 34(1):54–67, Jan. 2011.

[40] X. Wang, J. Yin, and D. P. Agrawal. Analysis and optimization of the energy efficiency in the 802.11 DCF. *Mobile Networks and Appl.*, 11(2):279–286, Apr. 2006.

[41] H. Wu, S. Nabar, and R. Poovendran. An energy framework for the network simulator 3 (NS-3). In *Proc. of SIMUTools*, Mar. 2011.