Toward Enterprise Virtual Power Consumption Monitoring with Joule

Roberto Riggio, Tinku Rasheed
CREATE-NET

***Abstract*—Reliable power consumption metering is important for power consumption management and optimization in wireless access networks. As a matter of fact, it is estimated that by 2015 wireless access networks will account for about 90% of the entire wireless energy cloud footprint. This can be traced back to the massive deployment of WiFi hotspots in the recent years in order to partially offload cellular networks from the load generated by modern data hungry mobile applications. In such a scenario the first step in reducing the power consumption of any IT infrastructure is to *acknowledge* it. Production–level solutions for large–scale monitoring campaign do exist, in the form, for example, of managed Power Over Ethernet devices. These solutions are however expensive to install and to manage especially for large legacy deployments. To overcome this limitation we propose *Joule* a virtual power consumption monitoring solution capable of estimating in real–time the actual power consumption of WiFi Access Points. Field trials performed using various benchmarks and applications show that *Joule* can provide a precise estimation of the power consumed by a typical IEEE 802.11 Access Point.**

I. Introduction

WiFi is becoming a key element of the modern wireless access, complementing and extending cellular networks both in high density areas where they can provide additional capacity to hungry mobile applications and in rural settings where deploying LTE or LTE-A network is not economical. Such trend will result in wireless access networks alone accounting for 90% of the entire wireless cloud footprint by 2015 [1].

In such a scenario limiting the use of wireless networks as point of entry to the cloud is hardly a choice. Instead it becomes of capital importance for network operators to (i) acknowledge each component contribution to cloud energy footprint; and (ii) devise energy efficient wireless networks and change how they are managed. However, deploying power monitoring solutions in production networks is often not a viable solution due to the setup and management costs. This is particularly true in the case of large and/or autonomous (from the energy standpoint) deployments. In the latter case, achieving plug–load visibility can allow for smart power management strategies that can take into account the renewable energy source unavailability as well as the impact of traffic on the device's power consumption profile.

In this paper we propose a new approach for power consumption modeling and metering. This approach, called *Joule* aims at tackling the limitations of other solutions requiring dedicated power metering hardware, namely deployment and operating costs. *Joule* does not require any additional hardware and can be easily integrated with traditional as well as cloud–based network management solutions. *Joule* exploit a set of power consumption models [2] generated during an initial training phase in order to predict the actual power consumption of any networking devices starting from observable statistics. The prototype presented in this work can actually provide a precise estimation of the power consumed by a typical IEEE 802.11 Access Point (APs). The software has been released to the research community under a permissive BSD license[1]. This paper extends our previous work [2] on energy consumption modeling in UMTS femto–cells to a radically different radio technology, namely WiFi. Our key contributions are:

- We present the design and implementation of a practical system that applies power modeling techniques to off–the–shelf WiFi APs.
- We evaluate this implementation using a set of synthetic traffic sources as well as with real traffic in the form of an FTP session.

We envision *Joule* to be used in large–scale energy management frameworks, such as the ones proposed in [3], [4] replacing expensive and hard to manage hardware power consumption meters.

The structure of this paper is the following. In Sec. II, we provide some background on system–level power consumption modeling. The modeling methodology is introduced in Sec. III. Section IV describes the system architecture. We report on the evaluation methodology and on the results of the measurement campaign in Sec. V. Section VI surveys the related work. Section VII provides some final thoughts on *Joule* and on its limitations sketching a roadmap for future work.

II. System–Level Power Modeling

For the purposes of this work, a power consumption model is an empirical model that allows to estimate the power consumption of a device starting from a set of observable parameters. For example, in the case of WiFi APs, observable parameters are: bytes/packets transmitted and received, average CPU load, transmission power, and in general all the parameters that can readily obtained trough standard management interfaces such as SNMP.

Within this work we are interested in the *system–level* power consumption of a typical WiFi AP, i.e. the power consumption of the entire device as opposed to the component–level power consumption where the contribution of CPU, wireless interfaces, etc., is considered separately. The rationale behind our approach is that in typical networking devices, such as APs and switches, CPU and other subsystems' power consumption statistics are highly correlated with the network traffic. As a result it is reasonable to treat the entire system as a *black box*

This work has been partially supported by the European Commission FP7 project ABSOLUTE.

[1]Available at: https://github.com/rriggio/joule

rather than trying to isolate the contribution of the various subsystems to the overall power consumption.

The power consumption model for a certain device can be either generated in–lab under controlled conditions or directly on the field by deploying a managed power consumption meter on the production network and by correlating the power consumption figures with the network statistics gathered using commonly deployed network management solutions, e.g. SNMP–based network monitoring applications.

Such empirical models are typically derived during an initial training phase during which the system being modeled is subjected to a set of well designed workloads meant to represent all — or at least most of — the system's operating states and at the same time power consumption measurements are taken using suitable metering devices. Starting from this dataset matching the system's operating state and the actual power consumption can be done using several modeling techniques, e.g. linear regression.

Once the power consumption models have been generated, they can be exploited to estimate the consumption of an operational network in real–time without requiring any physical power consumption meter. Such a possibility is particularly interesting in a cloud–based network management scenarios where traditional network monitoring and management functionalities are moved to the cloud operator's datacenter and offered to the customer following a Software as a Service model. In such a context deriving a power consumption model for a certain WiFi AP would allow the cloud operator to offer a *virtual* power meter to all its customer using that particular device. Such concept can be applied to other networked IT equipment, such as switches, VoIP phones, IP video cameras, however due to space constraints this work will focus on 802.11 APs. Nevertheless it is worth noticing that the reliability of the models utilized in this work has already been demonstrated for a radically different radio technology, namely a CDMA–based UMTS femto–cells [2].

III. Model Training

A methodology for characterizing the power consumption of UMTS femto–cells has already been presented by the authors in [2]. Here we shall briefly summarize the methodology and we shall prove its applicability to 802.11 APs.

A. Experimental setup

The test environment is composed of a single AP and a single client. The AP is built around the PCEngines ALIX 2D2 (500MHz x86 CPU, 256MB of RAM) processor board equipped with one Mikrotik R52 IEEE 802.11 interface (a/b/g) with RTC/CTS disabled. The AP exploits OpenWRT 12.09 as operating system. The ath5k Wireless NIC driver has been used during the measurements campaign. The AP's operating frequency was set to 5.240GHz (Channel 48). The rate adaptation algorithm has been set to auto and the transmission power has been left to its default value equal to 17dBm for all experiments. The notebook is a regular DELL Latitude 6420 equipped with an Intel PRO/Wireless 3945AB wireless adapter and running Ubuntu 12.04 LTS.

Traffic is injected into the network at either the AP or the client side as a single UDP stream using different datagram sizes and transmission rates. Power consumption measurements are taken also without traffic in order to assess the idle power consumption. Results reported in this section are the average of measurements collected during 120 seconds with a 95% confidence interval. During the training phase we used a total of 10 different datagram sizes and 10 different transmission bitrates (summarized in Table I, set D_1). Streams are generated sequentially for every possible permutation of datagram size/bitrate, resulting in a total of 100 streams in each direction (client to AP, and AP to client). Considering an idle time of 5s between streams the entire measurements campaign took $\approx$ 7 hours to complete.

Training Sets	Datagram sizes [Bytes]	Bitrates [Mb/s]
D_1	32, 64, 128, 256, 384, 640 896, 1152, 1408, 1460	0.1, 0.5, 1, 2, 5 10, 15, 20, 25 30
D_2	32, 64, 128, 256, 384, 640 896, 1152, 1408, 1460	0.1, 1, 10, 30
D_3	32, 512, 1024, 1460	0.1, 0.5, 1, 2, 5 10, 15, 20, 25 30
D_4	32, 512, 1024, 1460	0.1, 1, 10, 30

TABLE I: Training sets.

We used a *Energino*[2] power monitor [5], [6] to measure the actual power consumption of the AP during the experiments. *Energino* is real–time power consumption meter, which allows measuring the energy consumption of any DC powered device, as well as powering it off. The maximum sampling rate for measurements is 10000 samples/s. Fluctuations in the values read from the analog inputs are filtered out by continuously polling the voltage and the current sensors between update periods and by dispatching the average values. For example, if the sampling period is set to 1s, both the voltage and the current readings will be the average of $\approx$ 5000 samples. *Energino* can monitor DC loads up to 60V and absorbing up to $5A$. The resolutions for the voltage and for the current are, respectively, $54mV$ and $26mA$. During our measurements campaign *Energino*'s sampling interval has been set to 200ms.

B. Modeling methodology

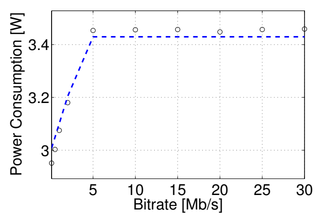
In a regular WiFi AP we can identify three main subsystems, namely: CPU, wired backhaul interface (typically an Ethernet interface), and WiFi. In this work we model the power consumption of just the WiFi subsystem. The reason behind this choice is that frame processing and forwarding takes place in dedicated silicon, i.e. the switch chip in the case of Ethernet frames and the WiFi chip in the case of the WiFi frames, as a result little load is imposed on the main CPU. Finally, the Ethernet frame processing is responsible for a negligible power consumption at the system level (one order of magnitude lower than the WiFi power consumption). We empirically verified the previous claims, however the results of such measurements are not reported due to space constraints.

As the main purpose of this work is to demonstrate the usability of *Joule* as energy monitoring subsystem, we do not propose new power consumption models but rather we reuse the ones developed in our previous works [2]:

$$P = \alpha(d)sat(x,d) + \beta(d)\delta(x) + \gamma \tag{1}$$

where P is the AP power consumption in Watts, x is the offered load in Mbps, d is the datagram size in bytes, $\delta(x)$ is

[2]Available at: http://www.energino-project.org.


(a) 128 bytes

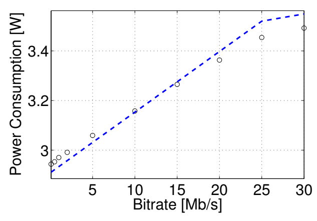
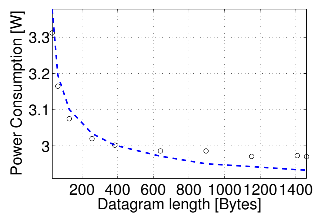

(b) 1460 bytes

Fig. 1: Average power consumption at the AP as a function of the bitrate for a constant datagram length. The AP is acting as transmitter.


(a) 1 Mb/s

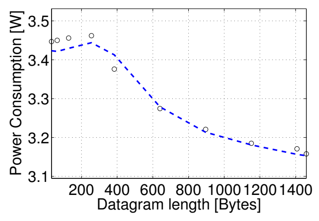

(b) 10 Mb/s

Fig. 2: Average power consumption at the AP as a function of the datagram length when the offered load is held fixed. The AP is acting as transmitter.

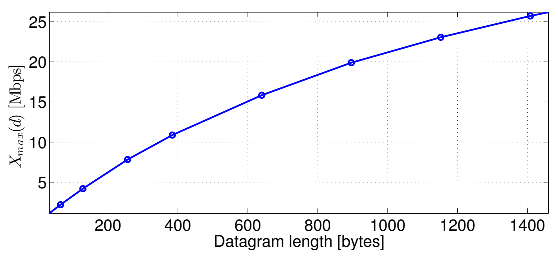


Fig. 3: Network throughput as function of datagram size.

the step function, and

$$sat(x,d) = \begin{cases} x & 0 \le x \le x_{max}(d) \\ 0 & x \le 0 \\ x_{max}(d) & x > x_{max}(d) \end{cases} \tag{2}$$

$$\delta(x) = \begin{cases} 1 & x > 0 \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

In this model, $sat(\cdot)$ represents the power consumption saturation regime reached when the offered load exceeds network capacity. This can be seen in Fig. 2 where the average power consumption is shown as function of the network load for a constant datagram length. The measured values for x_{max} are reported in Fig. 3. $\alpha(d)$ captures the fact that the rate of increase in power consumption with offered load is observed to depend on datagram size (also shown in Fig. 1). $\beta(d)$ captures the dependence of power consumption on datagram length when the offered load is fixed as shown in Fig. 2. γ captures the baseline power consumption when the AP is idle (no traffic besides the standard WiFi beacons).

The power consumption of the AP in idle mode, i.e., without any data but the standard IEEE 802.11 beacons, has been measured as $\gamma = 2.955$W. The dependence of $\alpha(d)$ on datagram size is primarily due to the contribution of fixed overheads (framing, contention, etc.), we can select

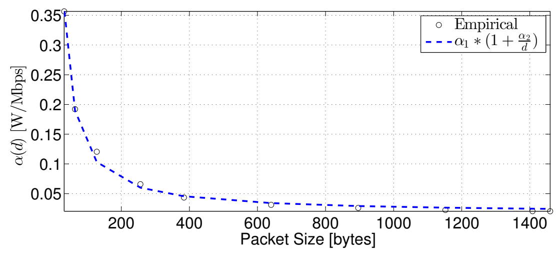


Fig. 4: $\alpha(d)$ values when the AP is acting as transmitter.

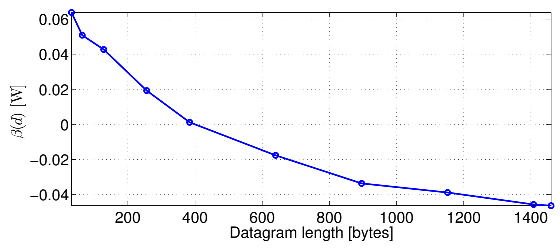


Fig. 5: $\beta(d)$ values when the AP is acting as transmitter.

Parameter	Transmitter	Receiver
$\alpha_0[\frac{W}{Mb/s}]$	0.0065	0.0026
$\alpha_1[Bytes]$	967	1749
γ [W]	2.955	

TABLE II: Energy model parameters

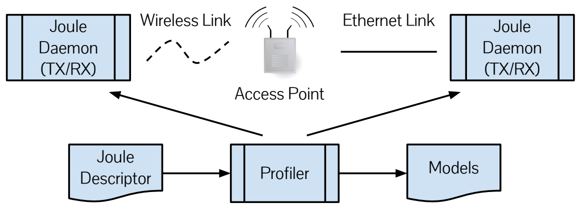


Fig. 6: *Joule* network architecture. Synthetic traffic patterns are loaded from the descriptor by the *Profiler* and executed by the *Daemons*. Statistics are collected by the *Profiler* and used to generate the power models.

$$\alpha(d) = \alpha_0(1 + \frac{\alpha_1}{d}) \tag{4}$$

where α_0, α_1 are parameters. α_1 can be thought of as the per datagram overhead, specified in bytes, while α_0 is a factor converting between units of bytes and energy. Figure 4 reports the empirical values of $\alpha(d)$ (as dots) and the fitted model (as dashed line). The fitting parameters are reported in Table II. The empirical values for $\beta(d)$ are shown Fig. 5.

IV. System Design

A. Architecture

As noticed before, measuring the energy consumption of a WiFi AP can be done trough suitable power meters. However, even for small-sized networks, i.e. tens of APs, deploying such solutions can be a daunting task in terms of both deployment and management costs. In order to work–around such problems we design and implemented *Joule*. The system architecture, sketched in Fig. 6, is composed of two main components:

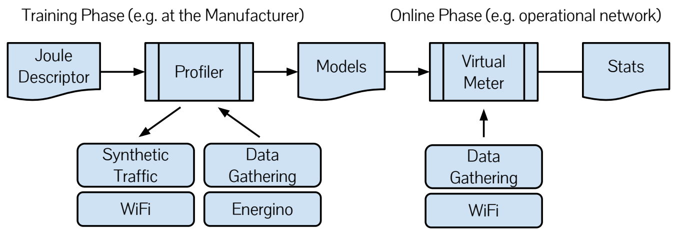


Fig. 7: The *Joule* workflow. During the *Training Phase* the *Profiler* generates different workloads while collecting power consumption samples. The models generated using these statistics are used by the *Virtual Meter* during the *Online Phase* to estimate the actual power consumption of the AP from measurable network statistics.

- The *Daemons*. They are software processes in charge of generating (and receiving) synthetic traffic patterns. At least two *Daemons* must be active in the network in order to obtain meaningful statistics. *Daemons* are implemented using the Click Modular Router [7].
- The *Profiler*. It is a software process in charge of triggering the transmission of different traffic patterns from any active *Daemon* in the network and collecting the power consumption statistics using a power consumption meter.

B. Workflow

Joule's' exploits a workflow consisting of two phases (see Fig. 7): the *Training Phase* and the *Online phase*. In the *Training Phase* the *Profiler* runs a series of synthetic traffic patterns between the AP/client and client/AP while collecting the power consumption statistics from the *Energino* in order to build the power consumption models. In the *Online phase* the *Virtual Meter* exploits such models in order to estimate the actual power consumption of the AP using measurable network statistics as input. We imagine that the training phase could be performed by the manufacturers in their lab, the resulting model could then be made available to the customers or be integrated in traditional Network Management software, e.g. Nagios [8].

Notice that the power consumption meter is not represented in the picture. It can actually be any device capable of exporting the instantaneous power consumption of the AP trough a serial interface. The current implementation of the *Profiler* supports the *Energino* device. However, the system can be easily extended to support other commercial solution such as managed POE switches and injectors commonly used in enterprise WLANs.

The *Profiler* takes as input a file named *Joule Descriptor* which is composed of two sections. In the first section all the *Daemons* currently active in the network are listed. For each *Daemon* information like IP address and control ports are provided. The second section defines a set of synthetic traffic patterns named *Stints*. Each *stint* specifies the source and the destination *Daemons* as well as the stream bitrate, its duration and the datagram size. At the moment only UDP streams are supported. The system does not impose any limit on the number of *Daemons*, At any given time a *Daemon* can transmit traffic to a single *Daemon*. However, *Daemons* can receive traffic from multiple *Daemons*.

After a *Stint* has been completed, the *Profiler* polls the *Daemons* in order to collect the network level statistics, i.e. the number of packets transmitted and received. The *Profiler* computes then the average and the median power consumption and their confidence interval. Finally, after all the stints have been completed, the *Profiler* computes the power consumption models and save the results to a model file which will be later used for the online power estimation.

It is worth noticing that, albeit depicted as separate entities, the *Daemons* and the *Profiler* can actually collapse into a single device, e.g. a powerful PC capable of generating the synthetic traffic patterns and of polling the power consumption meter for the power consumption samples.

V. Evaluation

In this section we report on the evaluation of the *Joule* virtual power meter. In particular we are interested in studying models generation time and accuracy.

A. Methodology

In order to study the relationship between model accuracy and stint duration we developed four different training programs using the combinations of packet lengths and bitrates summarized in Table I (sets D_1) and differentiated by the duration of each *stint*. The value considered have been 30s, 60s, 90s, and 120s. Each training program took between ≈ 2 hours and 7 hours to complete.

The actual composition of the training program in terms of datagram sizes and bitrates has a significant impact on the time required to generate the models. In order to understand the tradeoffs involved in the definition of the training sets we developed three additional training programs whose datasets used are summarized in Table I (sets $D_{2,3,4}$). Stint duration was set to 30s. Each training program took between $\approx$ 20 minutes and 2 hours to complete.

Finally, we also used a real application, namely FTP, in order to assess the accuracy of *Joule* under real–world workloads. The FTP session consisted in downloading a single file from a public FTP server[3] while monitoring the actual and the estimated power consumption. Although representative only for a fraction of the total Internet traffic, FTP provides us with a significant insight into the operation of a WiFi network when its back–haul link is completely saturated, i.e. when the power consumption of the WiFi AP depends only the conditions of the wireless link. Results reported in the next sections are the average of 10 runs. The 95% confidence intervals were always smaller than 5mW.

B. Implementation details

Traffic statistics have been obtained by running a simple instance of the Click Modular Router on the AP. Click is mostly used as high performance software for forwarding and manipulating packets. However, its architecture naturally supports other tasks such as traffic generation and monitoring. In our case we used Click to aggregate incoming and outgoing wireless frames according to their length. The process takes very little CPU resources (less than 1%). The aggregated statistics are accessed by the *Virtual Meter* over a Socket interface made available by Click. Binning is used in order to

[3]The Linux kernel version 3.10.6 ($\approx$ 73Mb).

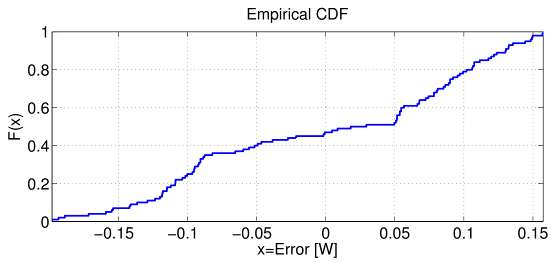


Fig. 8: Empirical CDF of the power estimation error for 120 seconds long stints. AP is acting as transmitter.

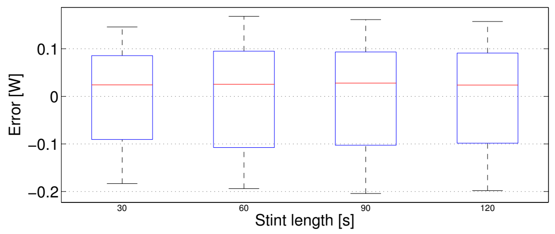


Fig. 9: Power estimation errors for different stint duration values. AP is acting as transmitter.

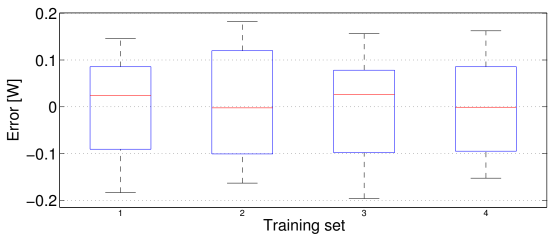


Fig. 10: Power estimation errors using different training sets. AP is acting as transmitter.

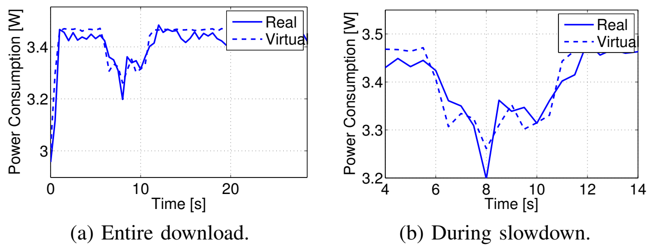


Fig. 11: Real (solid line) and virtual (dashed line) power consumption while download a large file using FTP.

consolidate this data into the intervals used during the training phase. Statistics are periodically polled by the *Virtual Meter* that exploits them in order to compute the estimated power consumption using Eq. 1.

C. Model Accuracy

The power consumption model described in Sec. II is simple and easy to calibrate: γ, $\beta(d)$, and $x_{max}(d)$ can be directly measured while $\alpha(d)$ varies in accordance with fixed overheads. Such model provides a good fit to the empirical measurements across the full spectrum of operating condition considered in this study. For example the energy consumption predictions are reported as dashed line in Fig. 1 and Fig. 2.

In Fig. 8 we report the empirical CDF of the power estimation error when using models generated from 120s long stints. The figure refers to the outgoing traffic, i.e. the AP–as–transmitter model has been used. The median error is very small (23mW) while the 10% and the 90% quantiles are, respectively, -0.132W and 0.130W.

Figure 9 summarizes the power estimation errors resulting from models generated using different stint duration values. As it can be seen, the error distributions are quite similar for all training sets. This allows us to conclude that even very short stints (30s) can actually lead to reliable models.

Finally, Fig. 10 summarizes the power estimation errors resulting from models generated using the training set in Table I. As it can be seen, using smaller training set in terms of either datagram sizes and/or bitrates does not affect significantly the accuracy of the models. On the contrary, it seems that using smaller training set can actually slightly improve the model accuracy. We ascribe this behavior to beneficial statistical smoothing effects achieved by considering fewer training points. However, it is worth noticing that, the selection of the training points in the alternatives training set has been done with some a–priori knowledge about the network MTU (1500 bytes) and about the maximum goodput achievable by an IEEE 802.11g AP ($\approx 28Mb/s$).

D. FTP Download

Figure 11 reports the instantaneous power consumption of the AP while downloading a large file from a public server. The graph refers to one particular run, out of the 10 performed during the measurements campaign, where an network slowdown happened between the 5th and the 15th second of the download. The network slowdown is caused by an interfering WiFi station active on a different network SSID and sharing the same channel with our experimental setup. As it can be seen, the virtual power meter closely tracks the actual power consumption during the slowdown. The median values of the actual and estimated power consumption across the 10 runs are, respectively, 3.437W and 3.446W. The median absolute error is 28mW.

VI. Related Work

System–level power consumption models are important for power management and optimization on a broad range of scenarios and use cases ranging from per–application energy consumption to datacenters power monitoring.

In the last years considerable attention has been devoted, for example, at devising on–line power modeling and estimation solutions for smartphones under the assumption that such model would enable the development of energy efficient “Apps”. System calls made by applications are traced in [9] in order to infer the energy used by specific processes. *WattsOn* [10] is in principle similar to the approach presented in this work in that it aims at generating accurate power consumption model for the various sub–systems found in a smartphones by exploiting reference benchmarks and empirical measurements. Similar approaches are presented in [11], [12]. The latter eliminates the need for external power measurement equipment, by using the battery drain measurements available on the mobile device itself. The *V-Edge* system [13] on the other hand is self–constructive and does not require current sensing, by exploiting instantaneous fluctuations in the battery voltage reading in order to infer the actual current absorption.

The body of literature on energy–aware solution for Ad–Hoc networks is humbling. In [14] an on–line power aware routing for Ad–Hoc network is proposed. A detailed study of the energy efficinecy of a number of contention–based MAC–layer protocols, including IEEE 802.11, is presented in [15]. However, most of these works exploit linear or step–wise power consumption models ignoring the non–linearity found in these networks. The authors already devised both a cheap and accurate power meter and a set of models for WiFi networks [5], [16], [17] and for Femtocell–based UMTS networks [2], however these works stopped at the analysis stage, while in this paper we refined and exploited those models for on–line estimation of an AP's power consumption under both synthetic and real workloads.

Energy efficiency is becoming evermore important in the operation of large IT infrastructures. The importance of achieving plug–load visibility is acknowledged in [18] where a power measurement study of a variety of networking gear such as hubs, edge switches, core switches, routers and wireless access points is performed. The authors then propose a benchmarking suite that allow network administrators to compare the power consumed by networking gear using different configuration profiles. The challenge of power metering in virtualized datacenters in tackled in [19], [20]. Power models are used in order to infer the power consumption at application and virtual machines level. The authors also show that existing instrumentation in server hardware and hypervisors can be used to build the required power models on real platforms with low error.

Sentilla [21] and JouleX [22] are two commercial solutions providing power consumption intelligence in datacenter environments. The former started as a wireless power metering company exploiting hardware sensors for tracking the actual energy consumption of datacenter gear. Acknowledging that such an approach does not scale well in terms of deployment and management costs, they moved to a software only approach exploiting virtual meters to calculate the power consumption of datacenters assets by extracting information about the equipment's workload. Similarly, JouleX exploits knowledge already available within a customer network in order to support the implementation of energy saving policies. However, the hardware requirements and the high licensing fees (JouleX is now part of the Cisco EnergyWise [23] offering) make JouleX not affordable for large scale hotspots deployment, that are already moving toward cloud–based network management solutions [24], [25] in order to reduce operational costs.

VII. Discussion

In this paper, we presented *Joule* a new approach for implementing large scale energy consumption tracking applications that do not rely on additional and expensive hardware power meters. The proposed solution relies on a flexible power consumption modeling framework capable of generating accurate models without any human intervention. Such models can then be embedded within traditional network monitoring and management solutions providing empirical support for adaptive energy saving strategies (such power cycling of APs according to the traffic conditions). A preliminary implementation tested over a real–world deployment has shown that *Joule* can indeed support accurate power consumption monitoring with a median error of 26mW and that such performances can be achieved with a training phase as short as ≈ 20 minutes. The system is also capable of closely tracking fluctuations in power consumption. As future work we plan to further study the impact of environmental noise on the power consumption model as well as to investigate mixed packet size scenarios and the applicability of the models to other wireless technologies such as the high throughput 802.11n and 802.11ac extensions and mobile technologies such as LTE and LTE-A.

References

[1] "The Power of Wireless Cloud: An analysis of the energy consumption of wireless cloud," CEET, Tech. Rep., June 2013.

[2] Roberto Riggio and Douglas J.Leith, "A measurement-based model of energy consumption in femtocells," in *Proc. of IEEE WD*, 2012.

[3] K. Gomez, C. Sengul, N. Bayer, R. Riggio, T. Rasheed, and D. Miorandi, "Morfeo: Saving energy in wireless access infrastructures," in *Proc. of IEEE WoWMoM*, 2013.

[4] R. Riggio, C. Sengul, K. Mabell Gomez, and T. Rasheed, "Energino: Energy saving tips for your wireless network," in *Proc. of ACM SIGCOMM*, 2012.

[5] K. M. Gomez and Roberto Riggio and Tinku Rasheed and Daniele Miorandi and Fabrizio Granelli, "Energino: An hardware and software solution for energy consumption monitoring," in *Proc. of WinMee*, 2012.

[6] Roberto Riggio and Cigdem Sengul and Lalith Suresh and Julius Schulz–Zander and Anja Feldmann, "Thor: Energy programmable wifi networks," in *Proc. of IEEE INFOCOM*, 2013.

[7] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek, "The click modular router," *ACM Transactions on Computer Systems (TOCS)*, vol. 18, no. 3, pp. 263–297, 2000.

[8] "Nagios." [Online]. Available: http://www.nagios.org/

[9] A. Pathak, Y. C. Hu, M. Zhang, P. Bahl, and Y.-M. Wang, "Fine-grained power modeling for smartphones using system call tracing," in *Proc. of ACM EuroSys*, 2011.

[10] R. Mittal, A. Kansal, and R. Chandra, "Empowering developers to estimate app energy consumption," in *Proc. of ACM MobiCom*, 2012.

[11] L. Zhang, B. Tiwana, Z. Qian, Z. Wang, R. P. Dick, Z. M. Mao, and L. Yang, "Accurate online power estimation and automatic battery behavior based power model generation for smartphones," in *Proc. of IEEE/ACM/IFIP CODES*, 2010.

[12] M. Dong and L. Zhong, "Self-constructive high-rate system energy modeling for battery-powered mobile systems," in *Proc. of ACM MobiSys*, 2011.

[13] F. Xu, Y. Liu, Q. Li, and Y. Zhang, "V-edge: fast self-constructive power modeling of smartphones based on battery voltage dynamics," in *Proc. of USENIX NSDI*, 2013.

[14] Q. Li, J. Aslam, and D. Rus, "Online power-aware routing in wireless ad-hoc networks," in *Proc. of ACM MobiCom*, 2001.

[15] J.-C. Chen, K. M. Sivalingam, P. Agrawal, and S. Kishore, "A comparison of MAC protocols for wireless local networks based on battery power consumption," in *Proc. of IEEE Infocom*, 1998.

[16] K. Gomez, D. Boru, R. Riggio, T. Rasheed, D. Miorandi, and F. Granelli, "Measurement-based modelling of power consumption at wireless access network gateways," *Computer Networks*, vol. 56, no. 10, pp. 2506–2521, 2012.

[17] K. Gomez, T. Rasheed, R. Riggio, C. Sengul, and N. Bayer, "Achilles and the Tortoise: Power consumption in IEEE 802.11n and IEEE 802.11g networks," in *Proc. of IEEE Greencom*, 2013.

[18] P. Mahadevan, "A power benchmarking framework for network devices," in *Proc. of IFIP Networking*, 2009.

[19] A. Kansal, F. Zhao, J. Liu, N. Kothari, and A. A. Bhattacharya, "Virtual machine power metering and provisioning," in *Proc. ACM SOCC*, 2010.

[20] A. Kansal and F. Zhao, "Fine-grained energy profiling for power-aware application design," *ACM SIGMETRICS Performance Evaluation Review*, vol. 36, no. 2, pp. 26–31, 2008.

[21] "Sentilla." [Online]. Available: http://www.sentilla.com/

[22] "Joulex." [Online]. Available: http://www.joulex.net/

[23] "Cisco EnergyWise." [Online]. Available: http://www.cisco.com/

[24] "Meraki." [Online]. Available: http://meraki.cisco.com/

[25] "Tanaza." [Online]. Available: http://www.tanaza.com/