ΑΛΕΞΑΝΔΡΕΙΟ Τ.Ε.Ι. ΘΕΣΣΑΛΟΝΙΚΗΣ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

BACHELOR THESIS

Characterize the Energy Consumption of IEEE 802.11 WLAN Interfaces

Author: **Vasileios Papadopoulos**

Supervisors:

Andres Garcia-Saaverda, Universidad Carlos III de Madrid (UC3M), Spain

Pablo Serrano, Universidad Carlos III de Madrid (UC3M), Spain

Periklis Chatzimisios, Alexander TEI of Thessaloniki, Greece

Thessaloniki 2011

TABLE OF CONTENTS

List of figures

List of tables

ABSTRACT

Over the past two decades the wireless communications market has experienced an explosive growth, and wireless technology has become an integrated part of our lives providing services from telephony to data transfer to sensing and monitoring applications. Business networks today are evolving to support people who are in movement. Employees and employers, students and faculty, government agents and those they serve, sports fans and shoppers, all are mobile and many of them are ”connected”. This is the vision of mobility- an environment where people can take their connection to the network along with them on the road. But mobile devices are dependent on battery power, it is important to minimize their energy consumption, while providing the required quality of experience. The energy consumption of a network interface can be significant, especially for smaller devices. The network interface is one of the main system components from the battery consumption standpoint. Hence, the energy used by the network interface to successfully transmit a message is an important figure for a mobile computer.

The need to minimize the footprint of mobile communication protocols has led to a shift in their design paradigm: the figure to optimize is no longer the number of bits transmitted per second, but instead the number of bits transmitted per joule. However, different mobile devices present very different power consumption figures (e.g., a laptop can consume ten times the power of a small access point), and therefore it is not clear how very - efficient devices should share the wireless resources with less - efficient devices. The design of protocols and/or deployments for wireless networks require accurate information of the energy consumption to be energy-efficient.

In general, the lack of reliable device specifications tends to stunt every try of estimating the device energy consumption. For example, idle mode, switching from transmit mode to receive mode and the effects of internal energy management strategies may not be reflected. Energy-aware design and evaluation of network protocols require knowledge of the energy consumption behavior of actual wireless interfaces. But little practical and reliable information is available about the energy consumption behavior of well-known wireless network interfaces and device specifications do not provide information in a form that is helpful to protocol developers.

RELATED WORK

There are few published measurements of the energy consumption of network interfaces. Experiments measuring the power consumption of IEEE 802.11 PC cards are reported in [1]. Emphasizing operation in conjunction with a base station, the methodology is based on sampling the current draw of another wise idle laptop as it sends and receives traffic over an extended period of time. By comparing the total energy used while sending or receiving traffic with the energy consumed by an idle laptop, the total cost of processing traffic can be computed. This approach has the advantage of measuring total system cost, but provides little packet oriented detail. Also, the results are potentially more dependent on particular system-wide energy management techniques than on the behavior of the network interface. Packet-oriented measurements of the energy consumption of wireless interfaces are described in [2]. This work is based on work presented in [14]. Because little practical information is available about the energy consumption and device specifications do not provide information in a form that is helpful to protocol developers. The data is presented as a collection of equations for calculating the energy consumed in various operations the network interface could occur.

SUMMARIZE

Engineers face a lack of knowledge on energy consumption figures when designing green protocols or deployments. Previous works tried to preliminary develop a model that could provide some information in a more packet oriented approach. However, those works did not take into account the variety of tune able parameters like e.g. transmit power levels, modulation coding scheme, or frame length that each station operates at. We present a methodology which aims to build the energy profile of network interfaces considering the different parameters that could affect the consumption.

In this work we first emphasize the need of minimizing uncertainties that would occur while performing experiments. The objective aim is to characterize the energy behavior in terms of repeatability. We could summarize our work as a packet oriented energy consumption while addressing issues by protocol configurable parameters. We will separate the factors that could play a major role to the energy profile into two categories: Internal factors and External. Furthermore, the wireshark facility was used to ensure that no other traffic was present on the channel as well as for monitoring purposes. Energy consumption was determined by direct measurement of the input voltage and current draw at the network device.

CHAPTER 1

<STATE OF THE ART>

The IEEE P802.11 committee developed the 802.11 Wireless LAN standards to cover wireless networks for fixed, portable, and moving stations within a local area. This family of standards addresses the need for wireless connectivity to stations, equipment, or automatic machinery that requires rapid deployment and may be portable, handheld, or mounted on moving vehicles. The P802.11 group planned to develop one media access control layer (MAC) that can support various physical layers using electromagnetic waves through the air (that is, radio waves as well as infrared). The standard specifies three physical layers to give the user maximum flexibility. In this section, we will discuss about the PHY and MAC specifications of IEEE 802.11 standard.

IEEE 802.11: PHY SPECIFICATION

The IEEE 802.11 standard defines both the physical (PHY) and medium access control (MAC) layers (Figure 1). The PHY layer specifications allows three transmission options namely, diffuse infrared (DFIR), direct-sequence spread spectrum (DSSS) and frequency-hopping spread spectrum (FHSS). While the DFIR PHY layer operates at base band, the two radio frequency options (i.e., DSSS and FHSS) operate at the 2.4 GHz band. This frequency band is a global band primarily set aside for industrial, scientific and medical (ISM) use, but can be used for operating WLANs without the need for end-user licenses. These options enable WLANs to be deployed different coverage areas ranging from a single room to an entire campus.

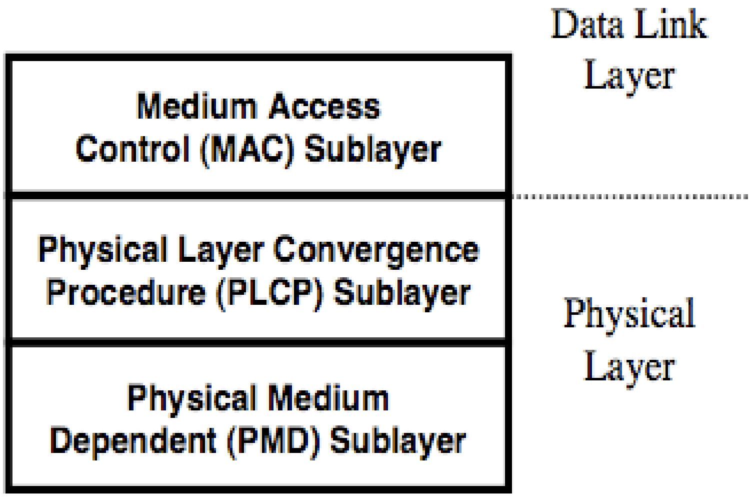

Medium Access Control (MAC) Sublayer
Physical Layer Convergence Procedure (PLCP) Sublayer
Physical Medium Dependent (PMD) Sublayer
Data Link Layer
Physical Layer

Figure 1 "IEEE 802.11 Basic Reference Model."

The PHY layer is divided into two sub layers. The physical medium dependent (PMD) sub layers deals with the characteristics of the wireless medium and defines a method of transmitting and receiving data through the medium. The physical layer convergence procedure (PLCP) sub layer specifies a method of mapping the MAC sub layer protocol data units (MPDUs) into a framing format suitable for the PMD sub layer. The DSSS IEEE 802.11 WLAN standard supports mandatory bit rates of 1 and 2 Mbps. For the FHSS standard, the 1 Mbps bit rate is mandatory while the 2 Mbps bit rate is optional.

IEEE 802.11: MAC SPECIFICATION

The basic MAC protocol is known as carrier sense multiple access with collision avoidance (CSMA/CA) which is very similar in operation to the carrier sense multiple access with collision detection (CSMA/CD) employed by Ethernet. In both protocols, the availability of the transmission medium is detected through carrier sensing. The distinguishing feature between CSMA/CA and CSMA/CD is that the former requires the receiver to send a positive acknowledgment (ACK) back to the transmitter if a frame is received correctly. Retransmission is scheduled by the transmitter if no ACK is returned. In CSMA/CD, the transmitter makes use of collision detection to determine whether a data frame has been transmitted successfully. The IEEE 802.11 standard requires ACKs to be issued only on receipt of unicast frames. Broadcast or multicast MPDUs are not acknowledged since this will lead to collisions among multiple ACKs. The function of the MAC protocol is common to all 3 PHY layer options (i.e., DSSS, FHSS, DFIR) and is independent of the data rates.

The MAC protocol can be enhanced by incorporating the virtual carrier sense mechanism which distributes reservation information by announcing the impending use of the wireless medium. The exchange of request-to-send (RTS) and clear-to-send (CTS) frames prior to the transmission of the data frame serves this purpose. The RTS and CTS frames contain a duration field that defines the period of time the medium is to be reserved for the transmission of the data frame and the returning ACK frame. The short RTS and CTS frames also allow the transmitter to infer collisions quickly. In addition, the CTS frame alerts neighboring network nodes (that are within the range of the receiver but not of the transmitter) to refrain from transmitting to the receiver, thereby reducing hidden node collisions. Like the ACK mechanism, RTS/CTS cannot be applied to MPDUs with broadcast and multicast addressing due to a high probability of collision among a potentially large number of CTS frames. Because of the large overheads involved, the mechanism is not always justified, particularly for short data frames.

The various IEEE 802.11 frame formats relevant to the performance measurements are shown in Figures 2 and 3. It is important to note that the PLCP preambles and headers are transmitted at a bit rate of 1 Mbps. This allows a lower-rate (and lower cost) IEEE 802.11 WLAN to interoperate with a higher-rate (and higher cost) counterpart. At the same time, the relatively low rate of 1 Mbps enables the PLCP preambles and headers to be decoded without using power hungry equalizers

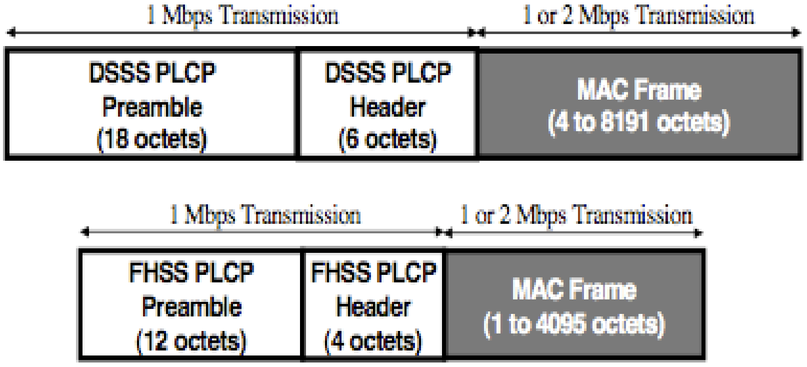

1 Mbps Transmission
1 or 2 Mbps Transmission
DSSS PLCP Preamble (18 octets)
DSSS PLCP Header (6 octets)
MAC Frame (4 to 8191 octets)
1 Mbps Transmission
1 or 2 Mbps Transmission
FHSS PLCP Preamble (12 octets)
FHSS PLCP Header (4 octets)
MAC Frame (1 to 4095 octets)

Figure 2. Physical Layer Convergence Protocol Frame Formats (DSSS and FHSS)

Figure 2 "Physical Layer Convergence Protocol Frame Formats (DSSS and FHSS)."

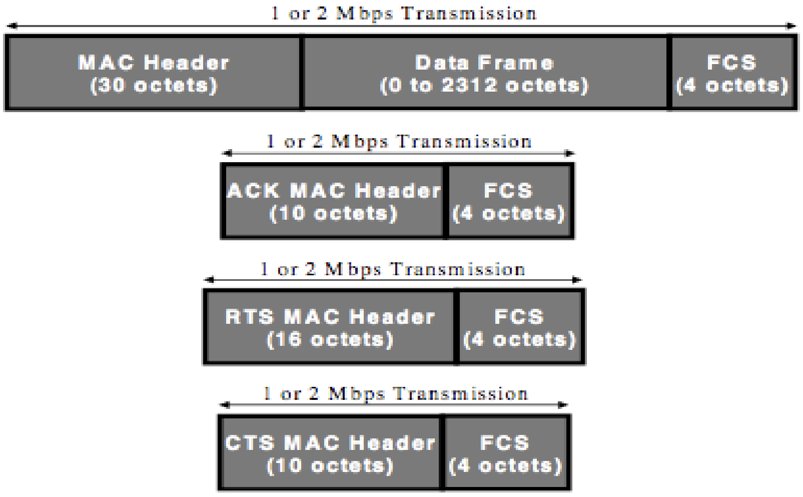


Figure 3 "Medium Access Control Frame Formats for Data,ACK, RTS and CTS Frames."

DISTRIBUTED COORDINATION FUNCTION

This section briefly summarizes the DCF as standardized by the 802.11 protocol. For a more complete and detailed description, refer to the 802.11 standard [4]. A station with a new packet to transmit monitors the channel activity. If the channel is idle for a period of time equal to a distributed inter-frame space (DIFS), the station transmits. Otherwise, if the channel is sensed busy (either immediately or during the DIFS), the station persists to monitor the channel until it is measured idle for a DIFS. At this point, the station generates a random back - off interval before transmitting (this is the Collision Avoidance feature of the protocol), to minimize the probability of collision with packets being transmitted by other stations. In addition, to avoid channel capture, a station must wait a random back-off time between two consecutive new packet transmissions, even if the medium is sensed idle in the DIFS time.

For efficiency reasons, DCF employs a discrete-time back-off scale. The time immediately following an idle DIFS is slotted, and a station is allowed to transmit only at the beginning of each slot time. The slot time size is set equal to the time needed at any station to detect the transmission of a packet from any other station. The slot time depends on the physical layer, and it accounts for the propagation delay, for the time to switch from the receiving to the transmitting state (RX-TX-Turnaround-Time), and for the time to signal to the MAC layer the state of the channel (busy detect time).

DCF adopts an exponential back-off scheme. At each packet transmission, the back-off time is uniformly chosen in the range. The value is called contention window, and depends on the number of transmissions failed for the packet. At the first transmission attempt, is set equal to a value called minimum contention window. After each unsuccessful transmission, is doubled, up to a maximum value. The values and reported in the final version of the standard [4] are PHY. The back-off time counter is decremented as long as the channel is sensed idle, frozen when a transmission is detected on the channel, and reactivated when the channel is sensed idle again for more than a DIFS. The station transmits when the back-off time reaches zero.

Figure 3 illustrates this operation. Two stations A and B share the same wireless channel. At the end of the packet transmission, station B waits for a DIFS and then chooses a back-off time equal to 8, before transmitting the next packet.

We assume that the first packet of station A arrives at the time indicated with an arrow in the figure. After a DIFS, the packet is transmitted. Note that the transmission of packet A occurs in the middle of the Slot Time corresponding to a back-off value, for station B, equal to 5.

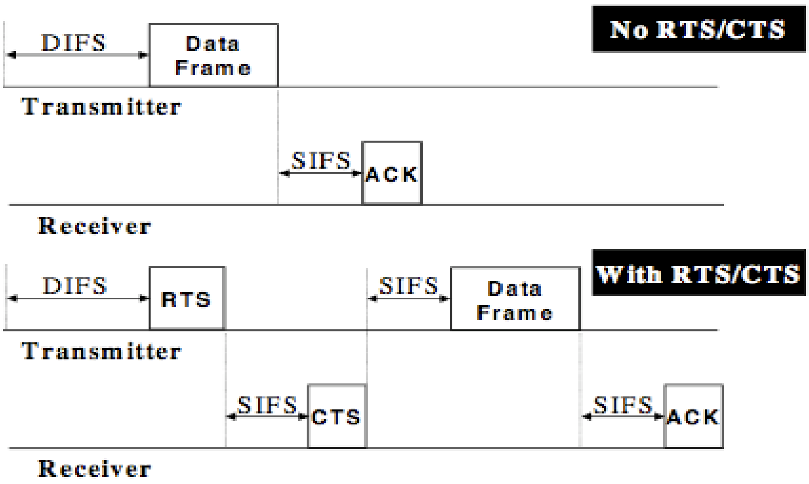


Figure 4 "Example of basic access mechanism."

As a consequence of the channel sensed busy, the back-off time is frozen to its value 5, and the back-off counter decrements again only when the channel is sensed idle for a DIFS.

The above described two-way handshaking technique for the packet transmission is called basic access mechanism. DCF defines an additional four-way handshaking technique to be optionally used for a packet transmission. This mechanism, known with the name RTS/CTS, is shown in Fig. 5.

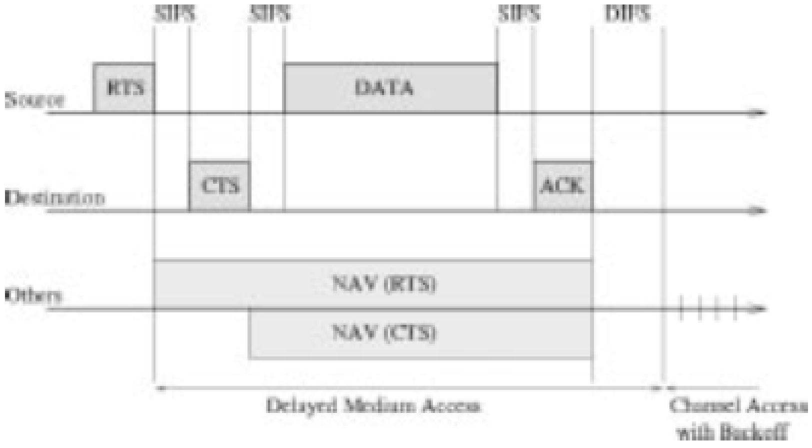


Figure 5 "RTS/CTS Access Mechanism."

For a in depth analysis of performance of PHY and MAC specifications, you can refer Bianchi's work defined in [5].

ENHANCED DISTRIBUTED CHANNEL ACCESS

The market of wireless networks is growing rapidly. Thus,many of applications applied in wireless networks require a level of services similar to the services applied in conventional wired networks. These applications have time sensitive data such as real-time data, voice and video. Their traffic data must be delivered at specific data rates and within specific delays, packet loss and jitter bounds, which are the parameters requirements of QoS. The legacy IEEE 802.11 (DCF) does not support realtime or multimedia application. Since, it is designed for equal priorities. There is no notion of high or low priority traffic. And also, a station node may keep the medium for as long as it takes.

IEEE 802.11e (EDCA) , defines traffic classes to classify packets in four Access Categories(AC) depending on their priorities. With EDCA, high priority traffic has a higher chance of being sent than low priority traffic: a station with high priority traffic waits a little less before it sends its packet, on average, than a station with low priority traffic. This is accomplished by using a shorter contention window (CW) and shorter arbitration inter-frame space(AIFS) for higher priority packets. In addition, EDCA provides contention-free access to the channel for a period called a Transmit Opportunity (TXOP). A TXOP is a bounded time interval during which a station can send as many frames as possible (as long as the duration of the transmissions does not extend beyond the maximum duration of the TXOP). If a frame is too large to be transmitted in a single TXOP, it should be fragmented into smaller frames. The use of TXOPs reduces the problem of low rate stations gaining an inordinate amount of channel time in the legacy 802.11 DCF MAC. A TXOP time interval of 0 means it is limited to a single MAC service data unit (MSDU) or MAC management protocol data unit (MMPDU).

AC	CWmin	CWmax	AIFSN	Max TXOP	Priority
Background (AC_BK	31	1023	7	0	12
Best Effort (AC_BE)	31	1023	3	0	3
Video (AC_VI)	15	31	2	3.008ms	45
Voice (AC_VO)	7	15	2	1.504ms	67
Legacy DCF	15	1023	2	0	-

Table 0 "Default EDCA parameters for each AC”

CHAPTER 2

<TEST BED AND METHODOLOGY>

So far we have seen an overview of IEEE 802.11 standard. We discussed about physical layer and medium access specifications and presented the basic access mechanism. The goal of this work was to investigate the energy consumption of a wireless network interface via direct measurement.

A variety of measurements of the power consumption of an idle laptop computer are found in [6],[7],[8]: reported results range from 6 to 14W. However, the growth of mobile computing is leading to the development of low- power mobile processors and systems.

In the following, we present the tools and equipment we use for our methodology. Real measurements could suffer from the impact of uncertainties and errors. In this part of the work we will talk about the Test bed and Methodology and all possible errors brought by the measurement of voltage and current.

TEST BED

In this section, we will present the equipment we use to perform our experiments. To draw valid usage of the devices we have first analyzed the requirements of each component; the weak spots, the strengths and the physical limits of equipment. We set up a scenario as the one displayed in Fig. 6. The station 1 (STA 1) is a SOEKRIS NET-4820 [12] with a wireless NIC installed and associated to an AP , it uses a Alfa Network AWPCI085S card. The power meter is a PCE-PA 6000 [13] power analyzer from PCE- iberica1 and the wireless monitor is a laptop with a serial connection to the power analyzer and a wireless card set to monitoring the channel. The traffic will be generated using Iperf over a 802.11a PHY layer, i.e. using a channel in the 5Ghz band so to avoid interferences from wireless networks deployed on campus using channels in the 2.4Ghz band.

The Alfa Network AWPCI085S supports adhoc mode operation as well as infrastructure mode operation. The latter was used throughout the measurements. Prior to proceed with the measurements, it is required to assess this setup in terms of accuracy and error handling.

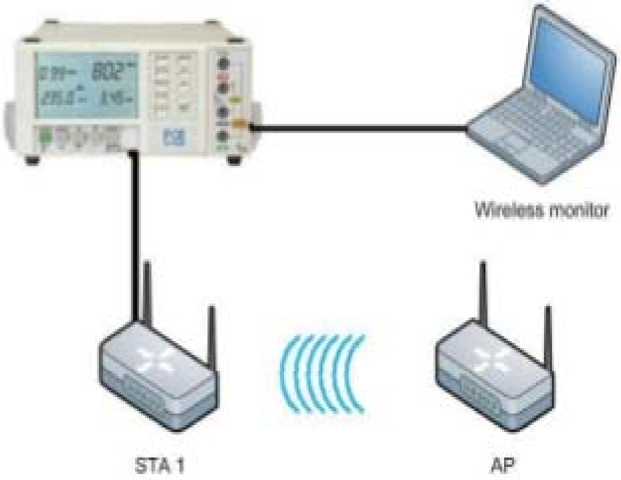


Figure 6 "Scenario for the experiments."

It is important to properly manage the impact of errors and uncertainties in our measurements and therefore to the power properties of the device. The initial work was to achieved a minimum possible variation in measurements. A measurement may be said to be repeatable when this variation is smaller than some agreed limit. Indeed, repeatability is the key to analyze and understand the energy consumption of WLAN devices. In this work we use 3 different configurations. Socket configuration delivers electric current from a wall outlet to the devices. We observed that scalability is responsible for little variations to voltage and current and therefore to measurements inaccuracy. A building could have many wall outlets that serves the users. On different times of the same day the electric current condition varies a lot.We present experimental work using devices(eg. UPS Hybrid configuration) that minimize the variation of repeatability and finally we switch to complete autonomous configuration using batteries as main source power of the devices.

ACCURACY, ERROR, PRECISION AND UNCERTAINTY

All measurements of physical quantities are subject to uncertainties due to the proper measurement technique. Variability in the results of repeated measurements arises because variables that can effect the measurement result are impossible to hold constant. Even if the ”circumstances,” could be precisely controlled, the result would still have an error associated with it. This is because the scale was manufactured with a certain level of quality, it is often difficult to read the scale perfectly, fractional estimations between scale marking may be made and etc. Of course, steps can be taken to limit the amount of uncertainty but it is always there.

In order to interpret data correctly and draw valid conclusions the uncertainty must be indicated and dealt with properly. For the result of a measurement to have clear meaning, the value cannot consist of the measured value alone. An indication of how precise and accurate the result is must also be included. Thus, the result of any physical measurement has two essential components: A numerical value (in a specified system of units) giving the best estimate possible of the quantity measured, and the degree of uncertainty associated with this estimated value. Uncertainty is a parameter characterizing the range of values within which the value of the measurement can be said to lie within a specified level of confidence.

Whether a number is an approximation of an actual value as a result of a calculation or a measurement, we must be able to describe the approximation. We will use the term precision to describe how close an approximation could be to an actual value and accuracy to describe how close it actually is. Given a series of measurements or approximations, there is an error associated with each value.

We will use the term precision to measure the spread of errors relative to each other. Precision is independent of the actual correctness of the measurements or approximations. Precision is usually described in terms of the number of digits used to make a measurement or approximation though it can also be described in terms of the standard deviation of the errors. Accuracy describes how close an approximation is to a correct answer. In this section, we will see how we can describe accuracy using either absolute or relative error and applying this techniques to the results gathered from out measurements in order to determine which methodology is more suitable to our needs. Given an approximation a of a correct value x, we define the absolute value of the difference between the two values to be the absolute error. We will represent the absolute error as follows:

$$\epsilon_{absolute} = |a - x| \qquad (1)$$

Although the absolute error gives us a feeling of the size of the error but how significant is the error? Furthermore , the absolute error will change depending on the units used. In order to handle concepts like these we have to take into account another quantity, we define the relative error to be the ratio between the absolute error and the absolute value of the correct value and denote it by

$$\epsilon_{relative} = \frac{\epsilon_{absolute}}{x} = \frac{a}{x} - 1 \qquad (2)$$

EXTERNAL IMPACTS

In summary, factors coming from external impulses referred as External Impacts. In order to illustrate the risks of using a configuration based on wall outlets we will run a simple experiment with a no wireless device such a lamp. The characteristics are known by manufacturer and will help us to calibrate the power analyzer. In this section we assess the accuracy and the precision of the methodology being modeled. To do this we will perform three experiments: We will first show how the power consumption varies depending on the electrical needs , we will improve this variation trying to keep voltage constant and finally we will search the optimal set up configuration using batteries. The experiments took place on different time on the same day. In addition, we had to take into account mechanisms such a ACPI (Advanced Configuration Power Interface) used by operating system which aims to consolidate and improve the power management. Furthermore, it was important to isolate the wireless device to only forwarding the traffic instead of generating and then transmitting. We will discuss these impacts further later in this document when we will use a network device.

Let us consider a scenario with a lamp as we referred before, the power consumption is P = 10W as this provided by the manufacturer. We then compute the overall power consumption using the typical configuration (Socket), using the hybrid (Hybrid.), and compare them against results from autonomous (Batteries).
For the first experiment (named Socket), we obtain average power P = 11.448W; whereas for the second experiment (named Hybrid) we obtain P = 11.196W and finally for the third experiment we obtain average power consumption P = 10.19984W. Table 1 shows the obtained values for the per-experiments absolute and relative errors. The Socket configuration, as expected, overestimates the power consumption of lamp offering an extremely high absolute error (E_typical = 1.448) , or 14.48 % o□ its actual value . In the Hybrid set up, still the inaccuracy occurring but with lower value (E_hybrid =1.196) or 11.96 indeed the absolute and relative error (E_autonomous = 0.19984) and 1.9984).

Configuration	Absolute Error	Relative Error %	Std Deviation
Socket	1.448	14.48	0.3504
Hybrid	1.196	11.96	0.1811
Batteries	0.19984	1.9984	0.08423

Table 1 "Absolute, Relative Error and STD Deviation non network device”

The bottom Fig. 7 indicates 10 different experiments . The left and right axes shows the corresponding power consumption. In a brief glance we will realize the Batteries configuration meet our expectations in terms of accuracy, on the other hand the other scenarios suffering from unstable behavior. As noted above, this work emphasizing to IEEE 802.11 WLANs interfaces and their power consumption characteristics, thus significant effort were made to avoid uncertainties and anomalies to construct appropriate methodology that examines and characterize the behavior of interfaces.

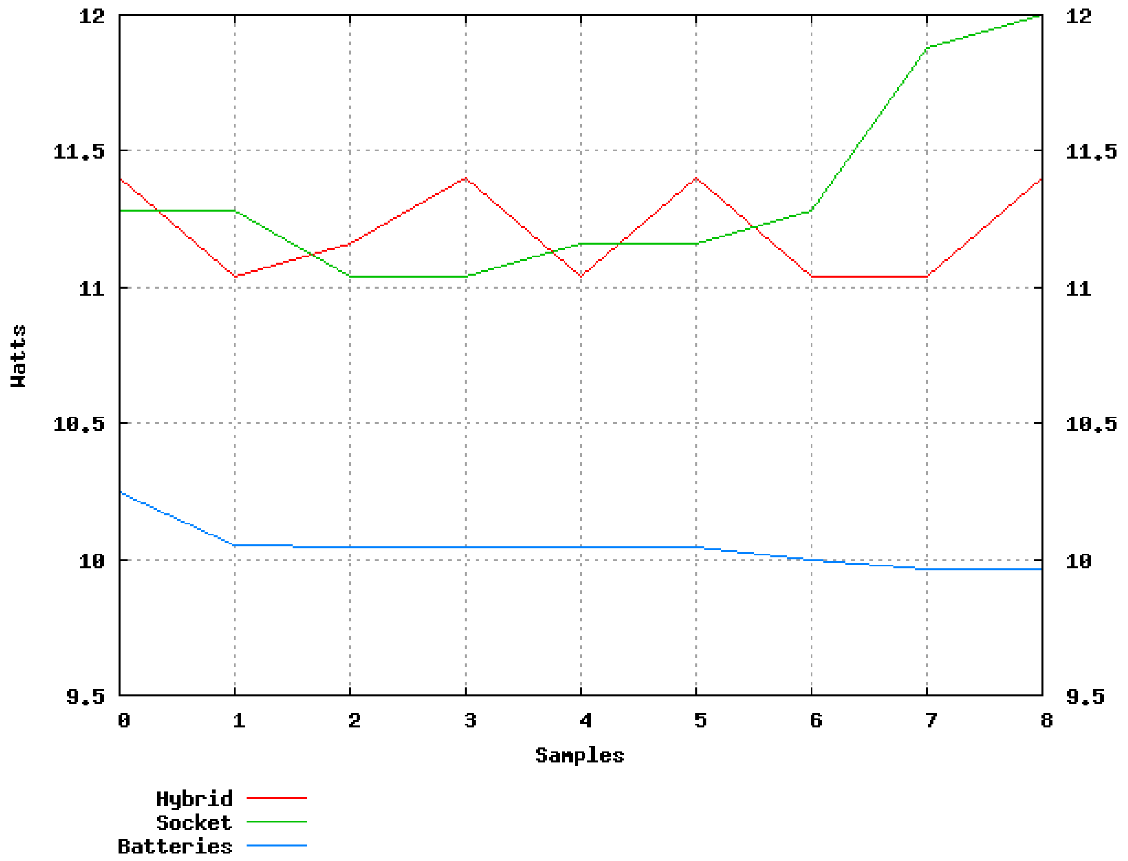


Figure 7 "Experiments for different configurations on different time NON wireless device

MEASUREMENT METHODOLOGY

Is the transport layer protocol a significant component in overall system energy consumption?

Next we compare with the results of three different scenarios with a network device. To this aim we plot in Figure 8 the power consumption (one for each configuration). A primarily tuning of the device included an Access Point association and information which sent upon the network took a place. We used as a transport layer protocol, a model without implicit handshaking dialogues for providing reliability, ordering, or data integrity. TCP implements reliable data delivery by re-transmitting segments that are not acknowledged within some retransmission timeout (RTO) interval. Accurate dynamic determination of an appropriate RTO is therefore essential to TCP performance. In our work we define a procedure that it could extract the power properties of interface, without taking account features of transport layer protocol. A User Datagram Protocol (UDP) meets our needs. The energy consumption closely depends on time. For this time-sensitive consumption it's better to use UDP because dropping packets is preferable to waiting for delayed packets. Re-transmission , flow control and other features from different transport layer protocols such as "TCP" might be referred as enemies that definitely weren't helpful to our laboratory work. However, this was not the only convention.

We followed an individual process, means a per-station extraction of power properties. In order to avoid collisions from different stations we assumed that the bandwidth didn't spread out for fair channel access, but instead we followed the isolation. A station has it's own channel to operate to network as it's the only device associated to the Access Point. The graphical overview in figure 8 reflects the cost of energy being spent over a bunch of transmissions of UDPs packets using iperf software, while protocol parameters remains untouched. Energy consumption was determined by direct measurement of the input voltage and current draw at the network device. Furthermore, in this primarily attempt we weren't concerned about influences caused by internal factors that we will discuss about latter in this work. We could consider as a factor the energy that the CPU consumes to create the packets. Finally, the energy was spending on pure a transmission mode as the station had always ready packets to its queue, Saturated Conditions. From the figure it is evident that the "Batteries" set up configuration shows better linearity and extremely low deviation. If we trust the results from the previous experiments with a lamp as it seen in the figure below the power consumption of Soekris net 4086 transmitting after 20 repetitions of the same experiment that the device transmitted all the time is around 5W. As we have seen before the "Hybrid" scenario slightly overestimates this value and the socket spreads out randomly the specific value.

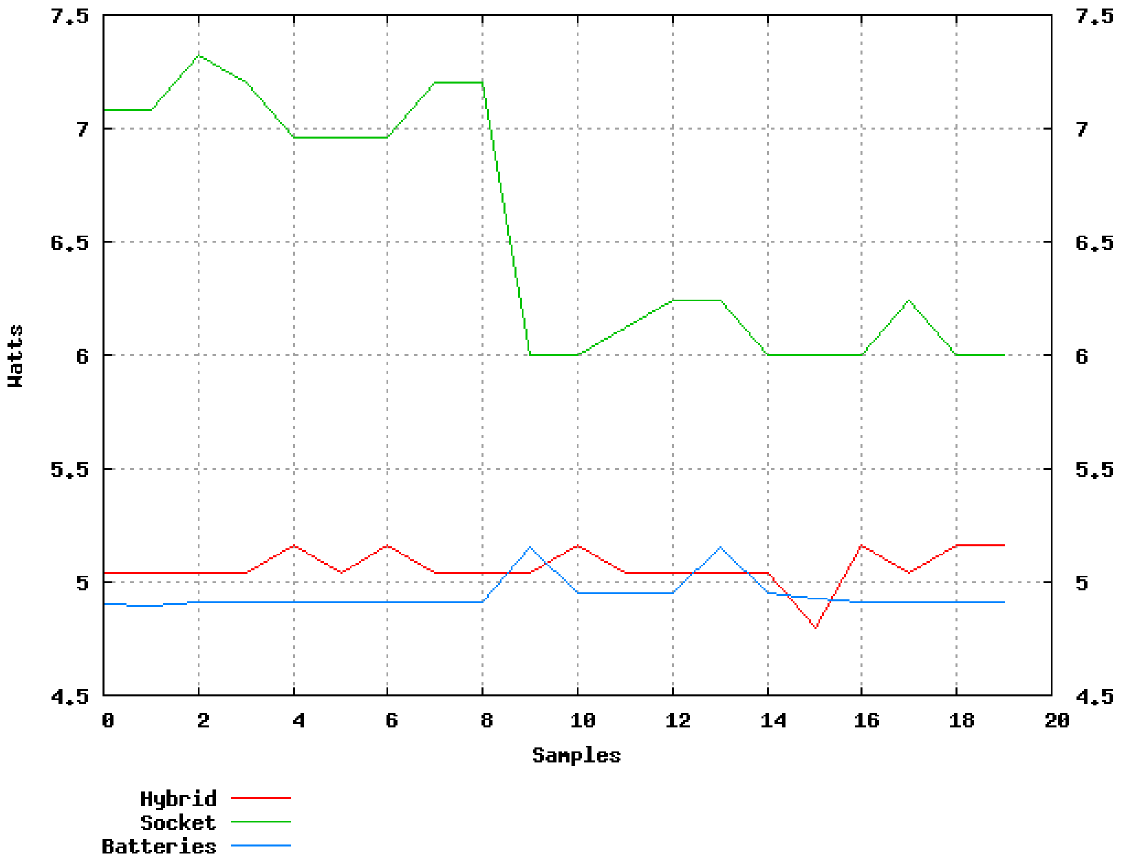


Figure 8 "Experiments for different configurations on different time WLAN device”

Configuration	Average Power Consumption (W)	Std Deviation
Socket	6.54	0.5387
Hybrid	5.064	0.0835
Batteries	4.94509	0.0734

Table 2 "Average power consumption and STD deviation. Soekris”

Significant efforts were made to avoid uncertainties that might led to power overestimation or inaccuracy. The mathematical approach was to ensure the repeatability and the elimination of other factors that could play significant role to misleading our methodology. Again noted that depending on the buildings conditions a decision about having different option to perform measurements was important. The unpredictability powering up the equipment direct from buildings electricity didn’t make it easy to draw many significant comparisons. Furthermore, major role played by features from transport layer protocol that encourages the station retransmit packets without acknowledgement frames which means more energy for the same packet. To avoid this situation we saturate the channel and ensure a negligible number of retry frames (i.e., using a non-populated channel). Finally, our machine running Gentoo 10.0 OS with Kernel 2.6.24.7 has a mechanism for power saving, which needed to turn it off because of an anomalies that could occur due an experiment.

So far we have presented results from 3 different configurations to minimize errors and uncertainties in order to achieve high repeatability on results regardless of when they were occurred. In the following section we will present results in the following configurations. ”Socket” and ”Batteries”.

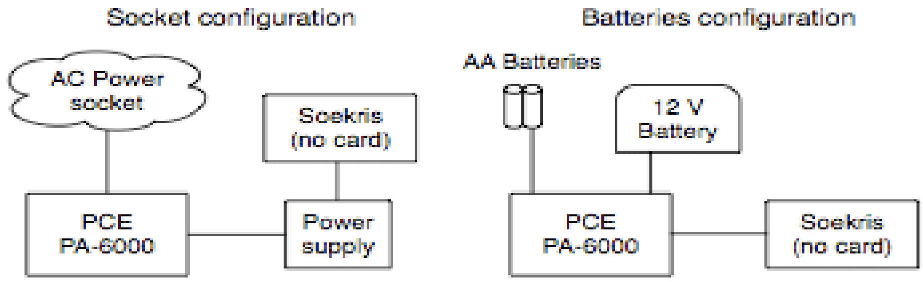


Figure 9 "Set up overview”

In figure 9 shows the connection to the above configurations. In ”Socket Configuration” The Power analyzer and wireless device powered by building’s socket. In addition the wireless card of the device is ”turned off” did not participate in a network , and not carried out any processes that could affect the energy consumption of the device. In Batteries configuration all the devices were isolated from external factors. Batteries used to cover power needs. In order to illustrate the possible risks whether were chose a typical configuration, we performed the following experiment. In figure 10 we can see the power consumption, phase ,the current and the voltage during this experiment. Even if the device didn’t handle any processes and the power savings mechanism were disable, the total power consumption suffered by inconsistency. The level of voltage oscillated randomly, as a result uncertainties could occur to our effort to characterize the power consumption of IEEE WLAN’s interfaces. On the other hand the BATTERIES configuration in figure 11 shows an extremely stable behavior until the time the batteries lose their power. It is evident that using batteries for power needs was by far the most secure set up configuration.

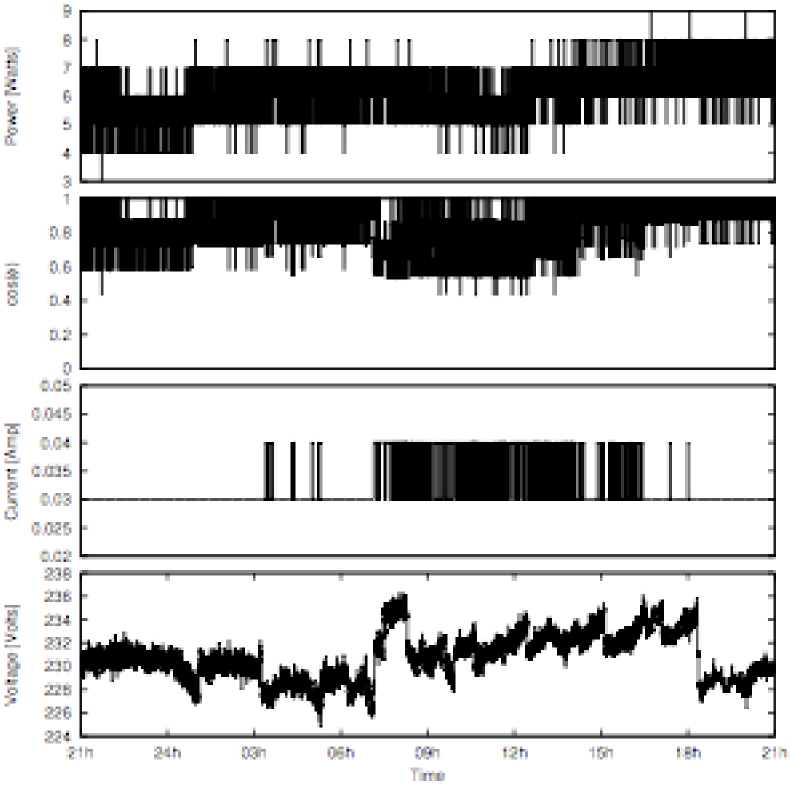


Figure 10" Power, Phase, Current, Voltage for 24 hours uptime SOCKET configuration”

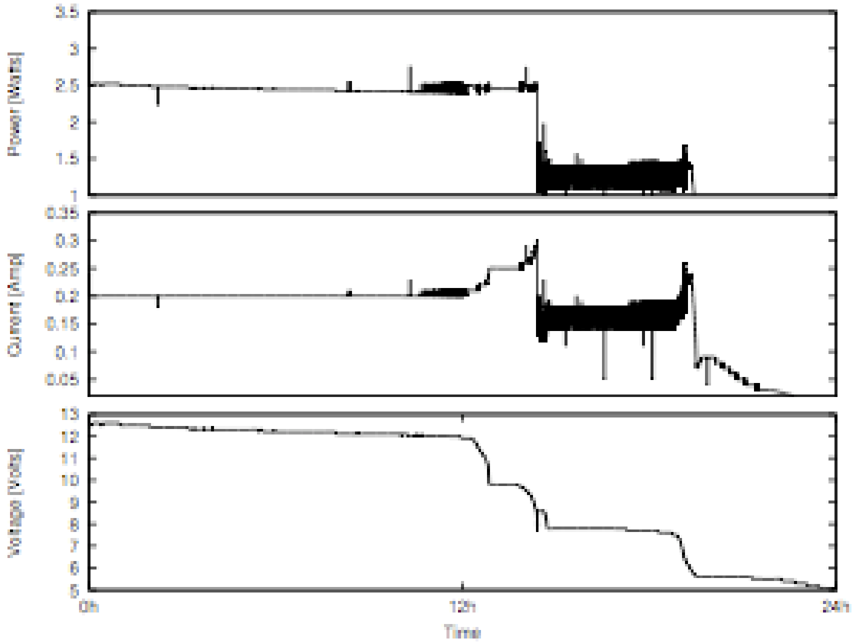


Figure 11 "Power,Current,Voltage for 24 hours uptime BATTERIES configuration”

CHAPTER 3

<ENERGY CONSUMPTION PARAMETERS>

So far we have seen that external factors can affect the outcome of an experiment. It was important to ensure a consistent results, i.e. to identify and tackle influences coming either from the external environment of the device (we decided BATTERIES CONFIGURATION) or by mechanisms implemented by the operating system as ACPI mechanism. During this section we will discuss the parameters that modeling the energy consumption of the wireless interface. The energy consumption is a time sensitive process. We presented the Distributed coordination function which is the fundamental MAC technique of the IEEE 802.11 based WLAN standard. DCF employs a CSMA/CA with binary exponential backo□ algorithm.

DCF requires a station wishing to transmit to listen for the channel status for a DIFS interval. If the channel is found busy during the DIFS interval, the station defers its transmission. In order to model the energy consumption of a WLAN we follow a similar approach to the one of [9] which defines three parameters to model the power consumption information of an 802.11 interface:

- Rtx : Power consumption of interface while transmitting.
- Rrx : Power consumption of interface while receiving.
- Rid : Power consumption of interface while neither transmitting nor receiving, but idling.

We will assume that stations only transmit data to the Access Point. While running controlled experiments we will collect the following information:

- Sniffer traces: The wireless monitor will capture in a trace files all the frames exchanged during the experiment. This will allow us later processing of this information. For this we will use wire-shark over a monitoring wireless interface
- Power consumption: The power analyzer will measure the energy consumed by STA 1 for the same period of time.

In order to measure the power parameters of a wireless NIC we will have to perform the numerical analysis on transmissions traces. We will use the three power parameters to characterize the interfaces behavior:As one interface can be in three different states: transmitting, receiving or idling, we can calculate the total energy consumed with the following expression where we have:

$$e = \rho_{id} T_{id} + \rho_{tx} T_{tx} + \rho_{rx} T_{rx} \tag{6}$$

- T_tx : Time spent in the transmitting state.
- T_rx : Time spent in the receiving state.
- T_id : Time spent in the idling state.

In order to compute these values we run controlled experiments where we saturate the channel and ensure a negligible number of retry frames (i.e. using an non-populated channel). The computerized processing of the trace files will allow us to count the number of frames and their length to calculate the value of T_tx and for T_rx T_total, T_id we have:

$$T_{TX} = (T_{PLCP} + \frac{H+L}{C})N_{TX} \tag{3}$$

$$T_{RX} = (T_{PLCP} + \frac{ACK}{B})N_{RX} \tag{4}$$

$$T_{id} \approx (N_{tx} + N_{rx})(DIFS + \frac{CW_{min}}{2}T_e + SIFS) \approx T_T - T_{tx} - T_{rx} \tag{5}$$

where we have:

- TPLCP : is the length of the frame preamble.
- H : is the frame header.
- C : the modulation rate being used.
- B : base modulation rate being used.
- ACK : represents the length of an acknowledgement frame.
- N_tx : Number of frames transmitted.
- N_rx : Number of frames received.
- DIFS , SIFS , Te : Constants defined by the standard 802.11.

We will perform a series of experiments to calculate the power properties of a wireless card. The processing of the traces for each measurement will give us the T_id , T_rx , and T_tx that will be used in the above equation. The results from the power analyzer (and gathered by the wireless monitor) provide the total energy consumed. Hence, we will have four equations that can be solved using matrix computation. In our work we assume that the interface only transmits packets to Access point hence we will calculate the TX power property.

$$[T] * [\rho] = [e] \tag{7}$$

$$[T_T T_{id} T_{rx} T_{tx}] * [\rho_{tx}] = [e_{tx}] \tag{8}$$

CHAPTER 4

<CHARACTERIZATION>

Communication protocols, and in particular the technologies used in the access network, have been originally conceived to optimize metrics other than energy, such as throughput or delay. Greening these protocols thus represents a shift in the design paradigm, where energy instead of time is the most critical network resource. We no longer want to maximize the bits sent per time unit, but instead the bits the network can send per each joule consumed. Works as the one in [3] presented a model for fair channel access taking into account the energy consumption properties. It is important that before work on the design of green protocols to understand where the energy is being spent.

OCCUPATION RATE

In this section we will introduce the Occupation Rate (OC). We define the rate between the time spent in each possible state over the total time of the measurement as occupation rate. We denote, OCTX , OCRX , OCID as occupation rate for transmit , receive and idle state, respectively. Theoretically, as the data rate raises the OCTX value decreases; the device has sent more packages, but also received more ACKs. Thus it is important to mention that:

$$OCTX + OCRX + OCID = 1 \qquad (9)$$

$$\frac{T_{TX} + T_{RX} + T_{ID}}{Measurement Duration} = 1 \qquad (10)$$

If we analyze the inter-frame spacing of 802.11 from energy consumption scope we will make the following observations. In order a station to transmit its packet, it has to sense the channel idle for a period of time, in other words it has to remain to idle state for DIFS period. After this time the device can send the information over the channel. After that it returns to idle mode for SIFS and then switches to receiving mode as the destination station sending back the ACK for the previous packet. DIFS and SIFS are standards defined by the protocol therefore OCID it’s not depending by internal factors(data rate,datagram length, tx power).

MODULATION

Modulation creates a radio or light signal from the network data so that it is suitable for propagation through the air. This involves converting the digital signal contained within the computer into an analog signal. As part of this process, modulation superimposes the information signal onto a carrier, which is a signal having a specific frequency. In effect, the information rides on top of the carrier. In order to represent the information, the modulation signal varies the carrier in a way that represents the information. A modulator mixes the source information signal, such as voice or data, with a carrier signal. The transceiver couples the resulting modulated and amplified signals to an antenna. The modulated signal departs the antenna and propagates through the air. The receiving station antenna couples the modulated signal into a demodulator, which derives the information signal from the radio signal carrier. One of the simplest forms of modulation is amplitude modulation, which varies the amplitude of a signal in order to represent data. This is common for light-based systems whereby the presence of a 1 data bit turns the light on, and the presence of a 0 bit turns the light off. Actual light signal codes are more complex, but the main idea is to turn the light on and off in order to send the data. This is similar to giving flashlights to people in a dark room and having them communicate with each other by flicking the flashlight on and off to send coded information.

In this work, we utilized Alfa Network AWPCI085S wireless card. Some characteristics that provided by manufacturer is about the modulation type the card uses for different data rates. For example rates between 6/9 Mbps the hardware using BPSK(Binary Phase Shift Keying) digital modulation technique and 12/18 Mbps QPSK(Quadrature Phase Shift Keying), respectively. Table 3 summarizes the characteristics for different data rates. It is worth to mention that for the rest of the paper we made the following convention; different modulation types do not affect the energy consumption and thus they have same similar power characteristics

Data Rates (Mbps)	Modulation Type
6/9	BPSK
12/18	QPSK
24/36	16QAM
48/54	64QAM

Table 3 "Modulation types used by Alfa network AWPCI085S inteface”

SATURATE

We have already discussed about factors that could play major role to constitution of the energy consumption profile. We referred to these components as ”Internal factors”. Significant factor is the data rate that the wireless device is operating.

First we will analyze the affection of using different modulation rates, at saturated conditions; the wireless device has always packets to send. Another laptop is used to generate the traffic and pass it to the considered device via another interface on another network(eg. Ethernet). The device being modeled, forward the traffic through the medium to the Access Point; a scenario that we presented before.

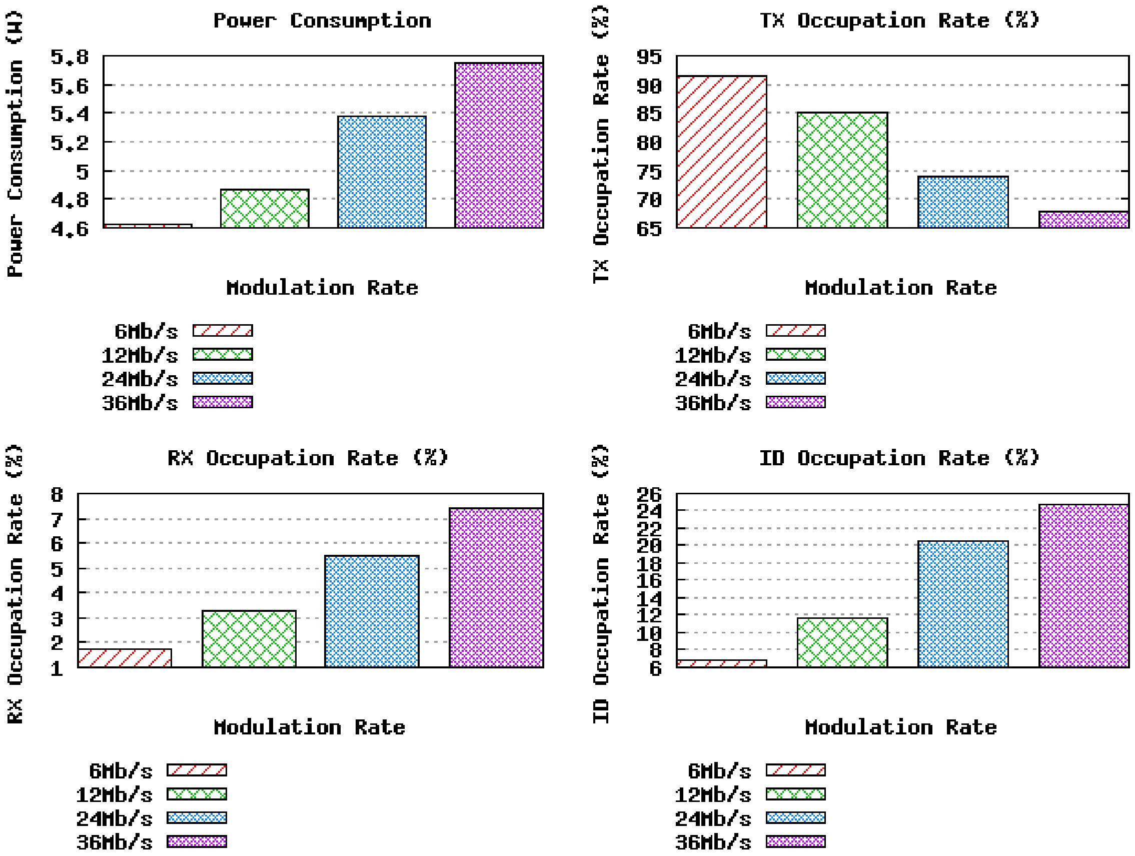


Figure 12 "Occupation rates and power consumption operating at saturated conditions"

If we take a look at the top-left section in the figure 12 we will see that the power consumption increases for higher modulation rates. From previous works we know that highly consumption state is the transmit state. The top-right section shows the OCTX. Even though we saturated the channel for all possible data rates(6,12,24,36 Mbps) the time the interface spends on transmit state decreases. This is an expected behavior once we transmit the information over the medium with higher rates. Thus, a successful transmission requires an acknowledgment frame as defined by the standard; therefore on same measurement duration we have more ACK's as the data rate increases. Exactly this behavior shows the bottom-left section of figure 12. Another expected conduct is OCID value that depends only the number of datagrams the interface forward regardless of the data rate.

NON SATURATE

Let's talk about a more realistic scenario. A scenario that the device doesn't has always ready packets to transmit. In order to compare the differences while changing data rates, we performed the following experiment. The consider station manages/forwards the amount of information which is the same at all time. Neither of rates were saturated, more accurately we have been sending traffic while keeping the throughput unchanged at 5-Mbps and analyze the occupation rates.

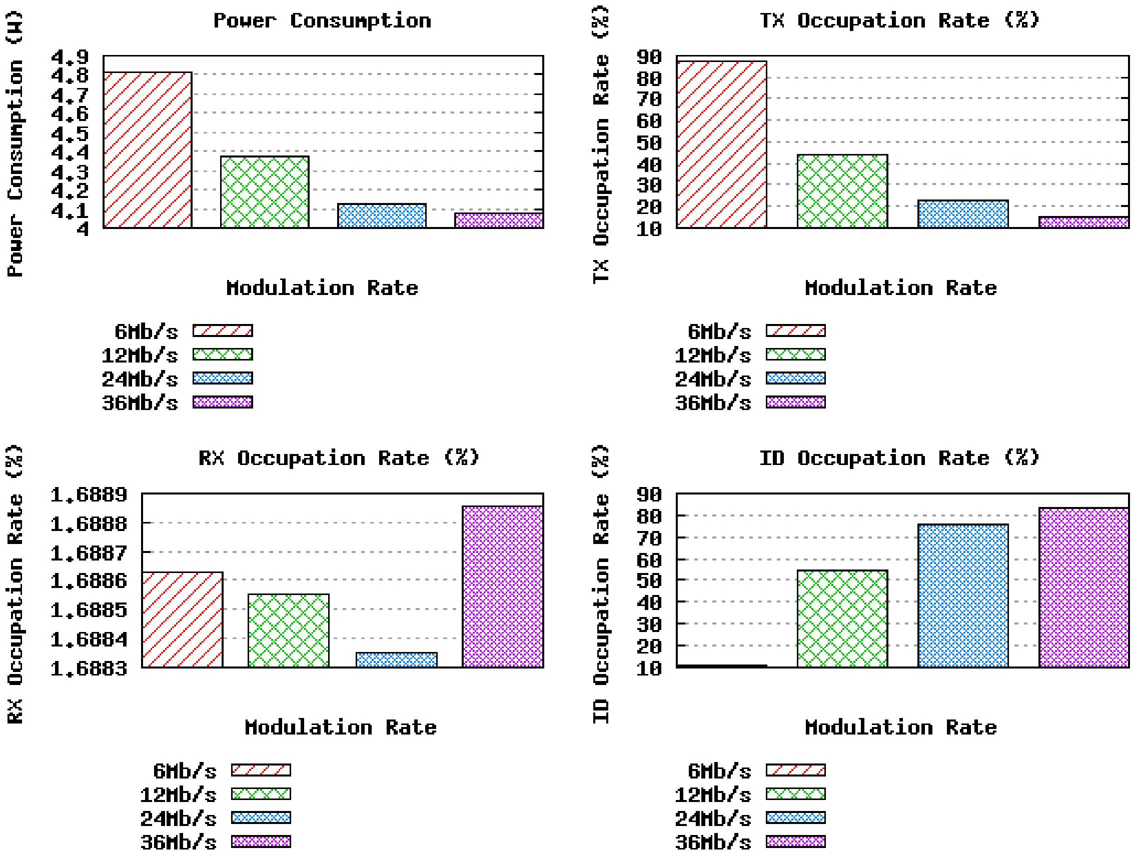


Figure 13 "Occupation rates and power consumption non saturated conditions, Constant Throughput"

In figure 13 we could see that the energy consumption while sending the same throughput is decreasing as the data rates increases. This is also an expected behavior as the time for a successful transmission depends by modulation rate. The top-right section of the figure shows how the OCTX falling for higher rates. If we look more closely to the OCRX value we could draw the following observation; the occupation rate for the reception is around 1.68%, on the other hand the idling state is growing linearly as most of the time the interface is operating to this state , considered to non saturated conditions.

DATAGRAM LENGTH

As we have seen before the appropriate Transport Layer Protocol to characterize the energy consumption is UDP. UDP uses a simple transmission model without implicit handshaking dialogues for providing reliability, ordering, or data integrity. UDP assumes that error checking and correction is either not necessary or performed in the application, avoiding the overhead of such processing at the network interface level. Time-sensitive applications often use UDP because dropping packets is preferable to waiting for delayed packets, which may not be an option in a real-time system or the measure of the energy consumption.

In this section we will make an analysis about how the length of the datagram(messages transport with UDP referred as datagrams) affects the consumption. We will use the same set up as it described before. A station A generate the traffic using ”iperf”. The wireless device B forward the information to a Access point. We use this technique because we are aware only about the behavior of the network interface and its energy profile.

As shown in Fig. 14, in the IEEE 802.11, irrespective of the payload size, the overheads such as MAC header, FCS, PLCP preamble/header, ACK, and some IFSs are used per frame basis. These overheads become relatively large when the size of the payload is small. In the next section, we will explain how these overheads can affect the throughput performance. We can easily see how these overheads can affect the system throughput through a simple numerical analysis [10]. The following assumptions are made for the analysis. One sender and one receiver operate with the DCF mode. The sender always has frames to transmit. Each frame has a fixed-size payload. Finally, this throughput is determined at the Link Layer Control (LLC) Service Access Point (SAP), which is the interface between the MAC and its immediate higher-layer, i.e., the 802.2 LLC. In this analysis, we use 36-Mbps for the data frame transmissions, and 24-Mbps for the ACK transmissions. Channel errors are not considered in order to emphasize the impact of the overheads. With the IEEE 802.11a, the transmission time of a data frame or MAC Protocol Data Unit (MPDU) with L-byte long payload at C Mbps PHY rate and the transmission time of the corresponding ACK frame at B Mbps PHY rate are given by equations (3),(4). Note that, in equations (3),(4) we calculate a bunch of transmissions as we multiply by the number of TX’s. Then, the throughput performance (T) of the system, when the data frames are transmitted at C Mbps and the ACK frames are transmitted at B Mbps, is determined as follows:

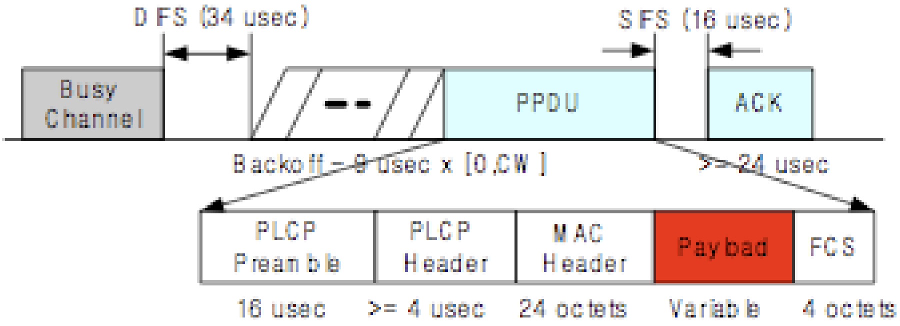


Figure 14 "Basic operation in DCF"

$$T(C,B) = \frac{8 * L}{DIFS + \bar{T}_{bk} + T^{C}_{data}(L) + SIFS + T^{B}_{ack}}(bps) \qquad (11)$$

where the average back-off time is given by :

$$\bar{T}_{bk} = \frac{CW_{min}}{2} * SlotTime \qquad (12)$$

A further analysis about throughput enhancement can be found in [11].

We followed the following experiment. The network between the access point and the station operate at 36-Mbps. The latter sends UDP traffic and performs a variation to data-gram length, starting 300KB and ending at the maximum possible size (1470KB) without fragmentation. This is known as maximum transmission unit (MTU), is the size (in bytes) of the largest protocol data unit that the layer can pass onwards. A larger MTU brings greater efficiency because each packet carries more user data while protocol overheads, such as headers or underlying per packet delays, remain fixed; the resulting higher efficiency means a slight improvement in bulk protocol throughput. A larger MTU also means processing of fewer packets for the same amount of data. In some systems, per-packet-processing can be a critical performance limitation. In figure 15, we present the Payload, Overhead and Throughput efficiency depending by the data-gram size. At the top of the figure we focus on the amount of bytes the interface has sent for different sizes for 30 seconds.

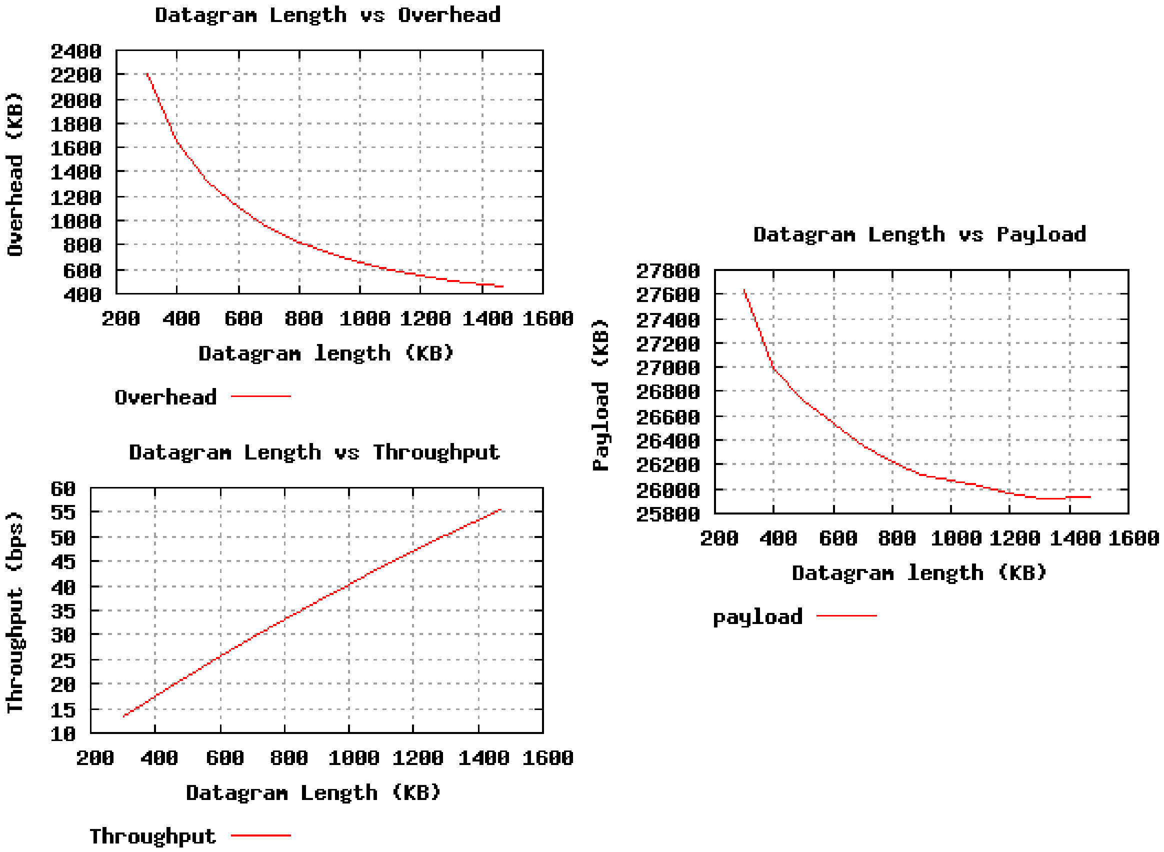


Figure 15 "Datagram variation vs Payload, Overhead, Throughput"

At the top section we could see what we have described before. A larger frame length tends to minimize the non user data information. Those overheads were responsible for the total payload increase, among the performed data-gram size modulation. Furthermore if we use the equations (11),(12) we can easily calculate also the throughput. The bottom of figure 15 shows the throughput performance in (bps).

So far we have discussed how larger data-gram lengths can have better efficiency. Next we will try to analyze the interface's behavior in terms of energy. We will use the Occupation Rates time units to this task. We also assume that the energy consumption while changing the data-gram length follows an exponential curve.

$$Energy_{dataLength} = A * B^{DatagramLength(KB)} \tag{13}$$

A,B are coefficients. Figure 16 presents this behavior. As expected for longer lengths the time the interface spends in TX state is becoming less. In terms of energy, this means a significant decrease in overall consumption, as the TX state is quite demanding on the energy. Another interesting element is that the ACKs frames also decreasing drastically as a consequence of having decline OCRX value, All this suggests an increase of the time being spent at the idle state as the interface remains at this for long periods.

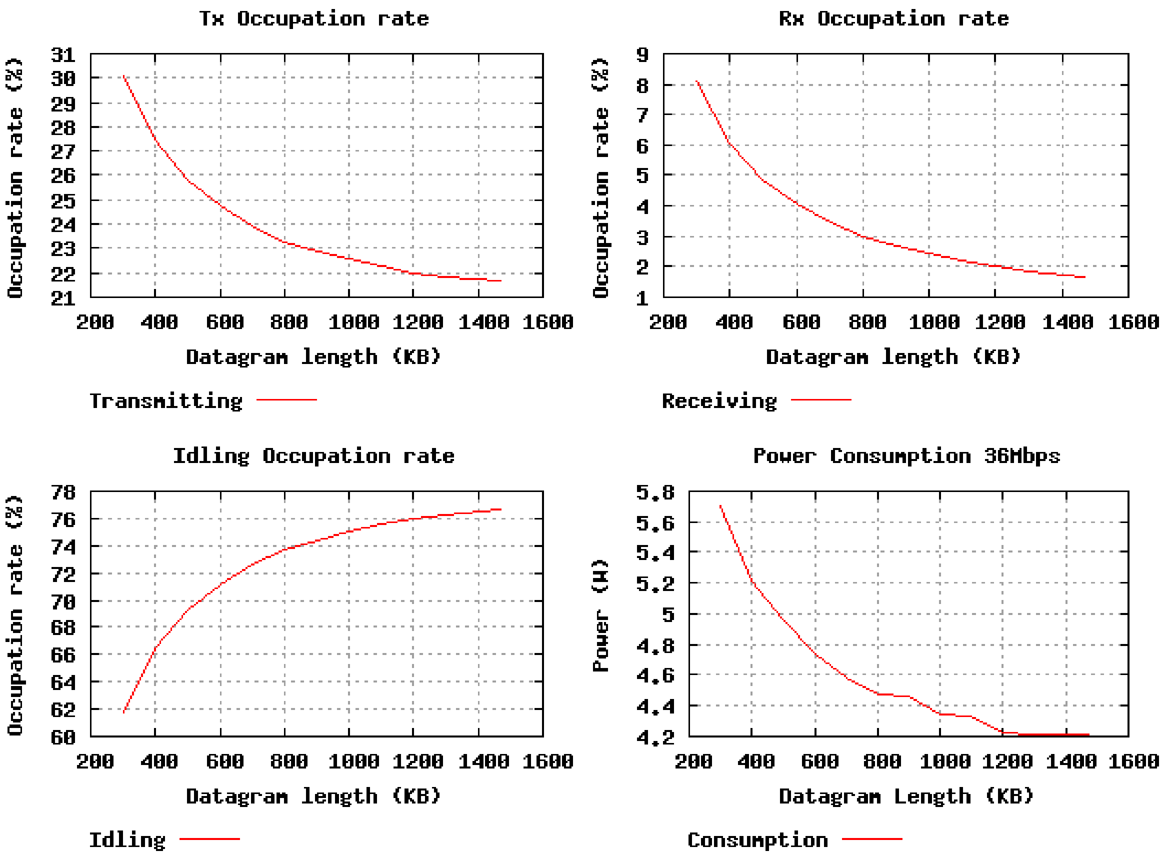


Figure 16 "Occupation Rates and Power consumption. Data-gram variation, 36-Mbps"

All these factors constitute the energy profile of the interface. Larger frames not only improve the efficiency in terms of performance, but also in terms of energy.

In this part of the work we have discussed concepts about, Occupation rates, Data rates and the appropriate frame size. We have seen that tuning the interface in terms of energy is quite similar to the performance tuning. The latter section of the characterization is about the transmission power levels.

TRANSMIT POWER LEVEL

In this section we will discuss about the impact of the Transmit Power Levels that could affect the total energy consumption. Before present experimental results, it is important to understand basic concepts.

Wireless access is known to be power-hungry for mobile devices. A key reason is that devices radiate power in all directions and much of this power will not reach the destination. Current mobile devices employ omnidirectional antennas and radiate power in all directions for wireless transmission. Such omnidirectional radiation not only introduces interference between peers, but also leads to directional waste of power. This is particularly true when the devices are only communicating with a single party, e.g. in infrastructure networks such as cellular and WLAN. In wireless local-area networks (WLANs), most wireless devices such as laptops and palmtops are battery-powered, and extending the operating time of such devices is always desirable and important.

The 802.11h standard specifies two TPC (Transmit Power Control)related functions. First, the AP in an infrastructure BSS or a wireless station in an independent BSS advertises the regulatory and local maximum-power level for the current frequency channel in the Beacon and Probe Response frames. They do sousing Country and Power Constraint elements. The local maximum specifies the actual maximum power level allowed in the BSS, which is less than or equal to the regulatory maximum. The stations in the BSS can use any transmit power less than or equal to the local maximum value.

Moreover, 802.11h provides a transmit-power reporting mechanism. It defines a TPC Report element that contains Transmit Power and Link Margin fields. The Transmit Power field simply contains the transmit power used to transmit the frame containing the TPC Report element, while the Link Margin eld contains the link margin, calculated as the ratio of the received signal strength (of the corresponding TPC request frame) to the minimum desired by the station. TPC Report elements are included in TPC Report frames in response to TPC Request frames. In addition, the AP in an infrastructure BSS or a wireless station in an independent BSS autonomously includes a TPC Report element in any Beacon frame it transmits.

Work in [3] shows that, depending on power properties of each device a trade-off is needed. There is a scenario that less demanding interfaces could be choked by higher for better energy efficiency ratio. The referred work focused on the basic power properties without taking into account important energy profile characteristics we shown in this work.

One component of the energy profile is the transmit power level. To be able to modulate the transmit power we require discrete power levels which can be set on the network interface. For better savings we require as many power levels as possible. The SOEKRIS NET-4820 series with MADWIFI [15] drivers offers 6 discrete power levels. The table below lists the power levels in dBm and mW. The default transmit power is 17 dBm.

mW	dBm
5	7
8	9
13	11
20	13
32	15
50	17

Table 4 "Range of transmit power levels available on the Soekris”

We performed experiments for various values of the TX Power of the interface shown in Table 4. Computed each time the SNR levels for each value of the table to see whether the signal is more powerful. Repeat the above procedure for all available modulation rates. The following figure indicating the behavior of the interface , can easily see that each time the TX Power is increased, equivalent to a better quality signal, as shown the SNR levels, but also a dramatic increase of the consumption of the network device. Furthermore, instead of dBm, if we use mW as unit of measure the TX Power , will acknowledged that the energy consumption increasing linearly with the growth of mW. We could describe the energy consumption using a linear equation.

$$Energy_{tx} = m * TransmitLevel(mW) + b. \tag{14}$$

Experimental results confirm the accuracy of the linear model and are used to determine values for the linear coefficients [m,b] and for various operations. Transmit Level is the transmission power in mWatts. The same behavior is followed for all the modulation rates as the interface is always at saturated conditions.

Future works might targeting at taking into account more criteria to determine the wireless station position. We argue that node positioning it could be important. We consider that RSSI (Received Signal Strength Indicator) metric technology could help us in the design of more Green Implementations and establish the trade-off between energy efficiency and throughput performance scenarios as well.

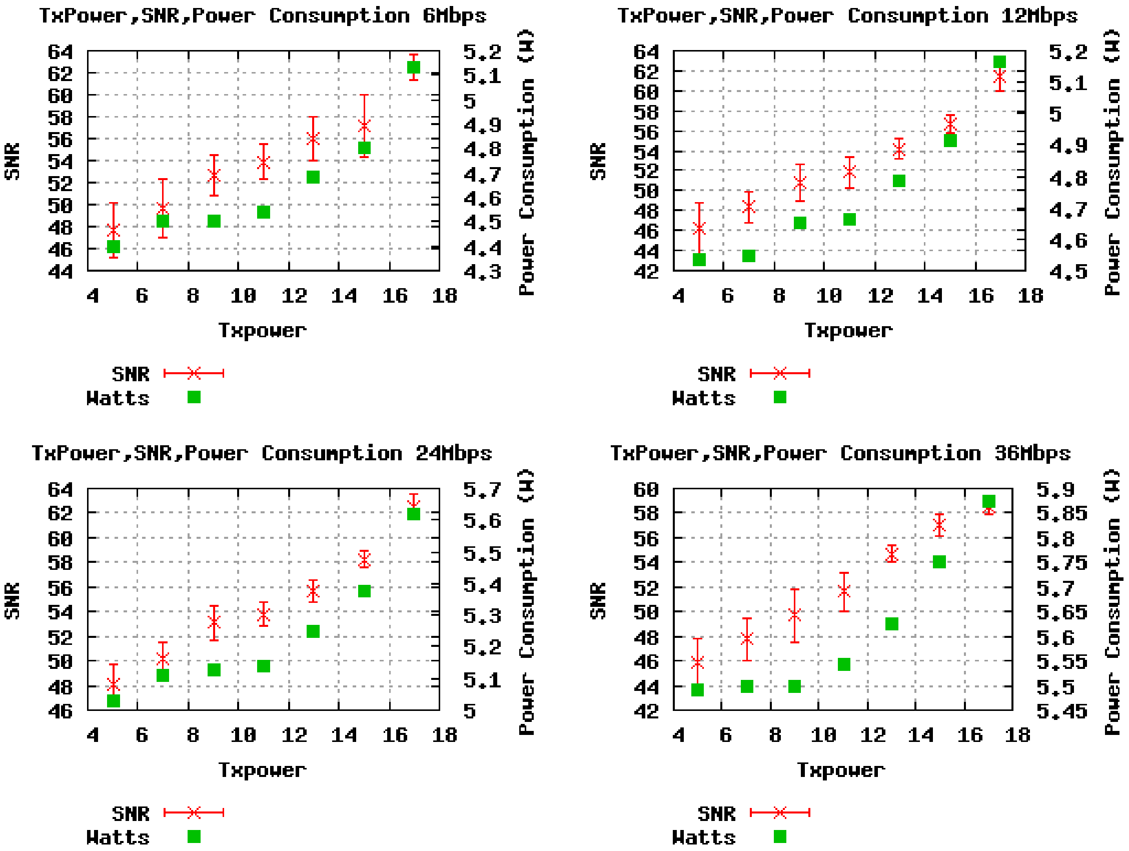


Figure 17 "x-axis:”TxPower”,y-axis:”Signal to Noise ratio”,y2-ayis:”Power Consumption 6,12,24,36 Mbps"

CHAPTER 5

<CONCLUSION>

SUMMARIZE

The results of this simple series of experiments show that the energy consumption of an IEEE 802.11 wireless interface has a complex range of behaviors that are relevant to the design of network layer protocols energy consumption is not synonymous with bandwidth utilization. Energy-aware protocol design and evaluation must consider factors such as the relative proportions of broadcast and point-to-point traffic, packet size and reliance on promiscuous mode operation. Energy is a precious resource in wireless networks because most nodes in the network are driven by battery and cannot be recharged in most cases. In order to keep the network functional as long as possible, energy-efficient protocols should be devised.

FUTURE WORKS

We have also presented an experimental methodology to characterize the power consumption of wireless devices. We have derived some preliminary figures about the power consumption of an 802.11 interface. However, although these numbers are consistent with the considered analytical model, we have also identified that the proposed methodology requires a more careful validation given the bias introduced by the measurement device. How to characterize this bias and the methodology to lessen its impact constitutes part of our future work as well. Furthermore, as part of our future work could be a more packet-oriented approach considered the internal impacts.

RESULTS SUMMARIZED

In this section will summarize all the gathered experimental results. We can divine our characterization methodology to 3 logical segments. At first segment we focused on how di erent data rates could build the energy profile. After processing the sniffer files we show in tables 5,6 the considered results.

(Mbps)	Idling(ms)	Receiving(ms)	Transmitting(ms)	Data	ACK
6	1497815	530200	27424595	13255	13255
12	2754601	976086	25497336	24377	24377
24	4682834	1658390	22170480	41442	41435
36	6310711	2234164	20328024	55847	55847

Table 5 "Results Modulation rate for 6/12/24/36Mbps. Saturated conditions”

(Mbps)	Idling(ms)	Receiving(ms)	Transmitting(ms)	Data	ACK
6	1427061	507189	26127332	12629	12628
12	1426512	505966	13203698	12624	12624
24	1425834	505215	6750135	12618	12618
36	1423848	504444	4586076	12600	12603

Table 6 "Results Modulation rate for 6/12/24/36Mbps.Constant throughput”

The second segment is the Data-gram variation. We have seen how important to performance and to the energy as well is the data-gram size. In order to achieve better efficiency the interface have to choose larger sizes without exceeding the MTU value. Otherwise , fragmentation would occur, tables 7,8 are referred to this topic.

(KB)	Idling(ms)	Receiving(ms)	Transmitting(ms)	Data	ACK
300	9808836	2430532	9027568	86804	86803
400	7332054	1816922	8240296	64886	64882
500	5880696	1457521	7753867	52040	52051
600	4905028	1214755	7423309	43412	43379
700	4200904	1041093	7174831	37176	37177
800	3646058	903824	6969080	32266	32266
900	3263666	808878	6873734	28882	28882
1000	2939918	729212	6763408	26014	26035
1100	2674032	662846	6672994	23664	23664
1200	2450986	607596	6593512	21690	21691
1300	2262163	561158	6545615	20019	20019
1470	2020545	499996	6510896	17889	17832

Table 7 "Results for different datagram lengths (300K - 1470K)”

(KB)	Power(W)
300	0.4604
400	0.4208
500	0.4000
600	0.3819
700	0.3700
800	0.3609
900	0.3600
1000	0.3500
1100	0.3500
1200	0.3402
1300	0.3402
1470	0.3400

Table 8 "Power consumption datagram length”

Finally, tables 9-12 show the overall power consumption, the signal- to-noise ratios and the standard deviation of signal’s strength.

(dBm)	Power(W)	SNR(db)	STD Deviation
5	4.394668856172	47.6171	2.5018
7	4.5010192	49.6825	2.6779
9	4.5023246	52.6646	1.7966
11	4.5363016	53.8956	1.5909
13	4.6801274	56.0604	1.9844
15	4.8012856	57.1742	2.8735
17	5.1342724	62.5303	1.2180

Table 9 "Results for different Tx Powe (5-17dbm) 6Mbps Saturated contitions”

(dBm)	Power(W)	SNR(db)	STD Deviation
5	4.536	46.2430	2.5705
7	4.5464958	48.3335	1.5595
9	4.6512774	50.8150	1.8162
11	4.662	51.8594	1.5272
13	4.788	54.1450	1.0149
15	4.914	56.6536	0.8780
17	5.166	61.4469	1.4283

Table 10 "Results for different Tx Powe (5-17dbm) 12Mbps Saturated contitions"

(dBm)	Power(W)	SNR(db)	STD Deviation
5	5.03125	48.0141	1.7321
7	5.11125	50.1727	1.2920
9	5.125	53.0826	1.3911
11	5.14125	53.7783	0.9493
13	5.25	55.6533	0.9161
15	5.375	58.2295	0.6875
17	5.6175	62.5068	1.0366

Table 11 "Results for different Tx Powe (5-17dbm) 24Mbps Saturated contitions"

(dBm)	Power(W)	SNR(db)	STD Deviation
5	5.49	45.9096	1.9619
7	5.5	47.7381	1.6492
9	5.5	49.6768	2.1108
11	5.54375	51.6328	1.5958
13	5.625	54.7088	0.6602
15	5.7525	57.0224	0.8840
17	5.875	58.5266	0.6835

Table 12 "Results for different Tx Powe (5-17dbm) 36Mbps Saturated contitions"

REFERENCES

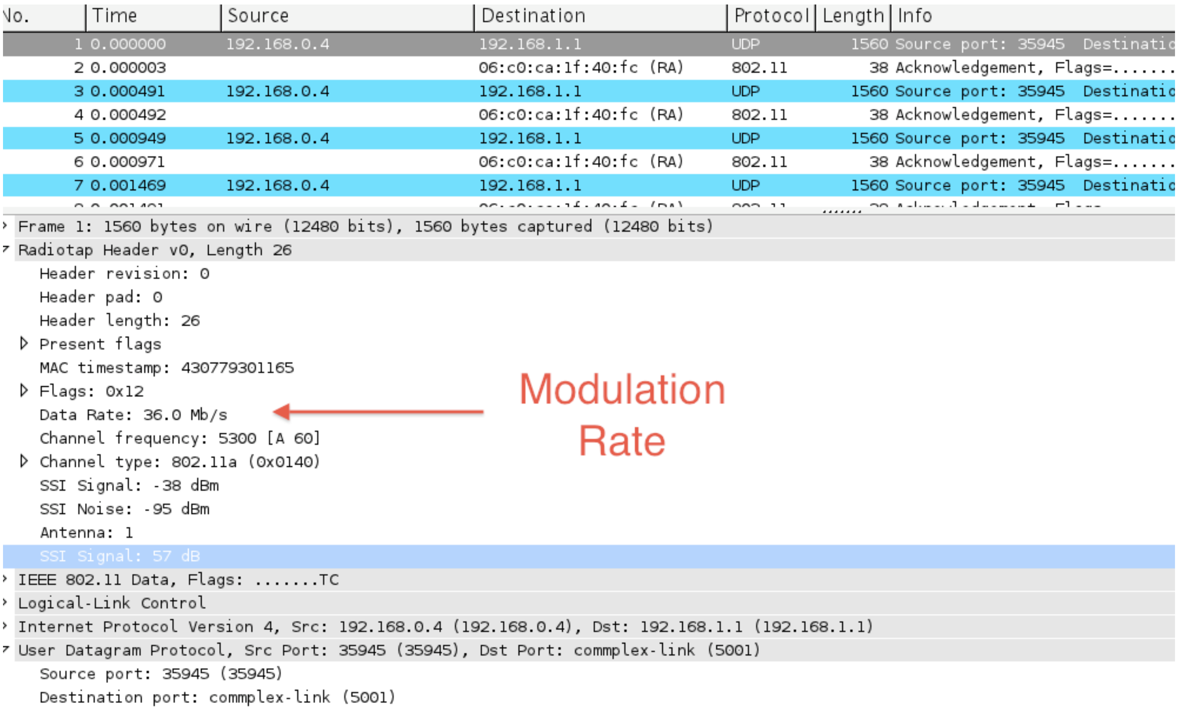
1. Robin Kravets and P. Krishnan, “Power Management Techniques for Mobile Communication”, in Proceedings of MobiCom 98. Dallas, TX, October 1998.
2. Paul Gauthier, Daishi Harada and Mark Stemm. “Reducing Power Consumption for the Next Generation of PDAs: Its in the Network Interface”, in Proceedings of MoMuC 96 September, 1996.
3. Andres Garcia Saavedra, “Energy - efficient fair channel access for IEEE 802.11 WLANs”, University Carlos III de Madrid Department of Telematics Engineering 2010.
4. IEEE Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, Nov. 1997, P802.11.
5. Giuseppe Bianchi, “Performance Analysis of the IEEE 802.11 Distributed Coordination Function”, IEEE Journal on selected areas in communication, vol. 18, no. 3, March 2000.
6. Keith I. Farkas, Jason Flinn, Godmar Back, Dirk Grunwald, and Jennifer Anderson, “Quantifying the Energy Consumption of a Pocket Computer and a Java Virtual Machine”, in Proceedings of the International Conference on Measurement and Modeling of Computer Systems (ACM SIG-METRICS) Santa Clara, CA, June, 2000.
7. Robin Kravets and P.Krishnan, “Power Management Techniques for Mobile Communication”, in Proceedings of MobiCom98, Dallas, TX, October 1998.
8. Wendi Rabiner Heinzelman, Joanna Kulik and Hari Balikrishnan, “Adaptive Protocols for Information Disseminationin Wireless Sensor Networks”, in Proceedings of MobiCom 99. Seattle, WA,August, 1999.
9. M. Ergen and P. Varaiya, “Decomposition of energy consumption in ieee 802.11. In Communications”, in Proceedings of IEEE International Conference on Communications (ICC), pages 403-408, June 2007.
10. Da ji Qiao, Sunghyun Choi, and Kang G. Shin, “Good-put Analysis and Link Adaptation for IEEE 802.11a Wireless LANs.”. IEEE Trans. on Mobile Computing (TMC), vol. 1, no. 4, pp. 278-292, October-December 2002.
11. Youngsoo Kim, Sunghyun Choi , Kyunghun Jang , Hyosun Hwang, “Throughput Enhancement of IEEE 802.11 WLAN via Frame Aggregation”.

IEEE Trans. on Mobile Computing (TMC), vol. 1, no. 4, pp. 278- 292, October-December 2002.

12. http://www.soekris.com
13. http://www.industrial-needs.com/technical-data/power-analyser-PCE-PA-6000.htm
14. Laura Marie Feeney, Martin Nilsson, “Investigating the Energy Consumption of a Wireless Network Interface in an Ad Hoc Networking Environment”, in Proceedings of Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), 2001.
15. http://madwifi-project.org/

ANNEXES

We used the traffic captured by the monitor station in order to examine some useful information. The information is encapsulated by different protocols and standards for succeful communication in an infrastructure topology. In the following section we present wireshark screenshots for the three types of the expreriments.

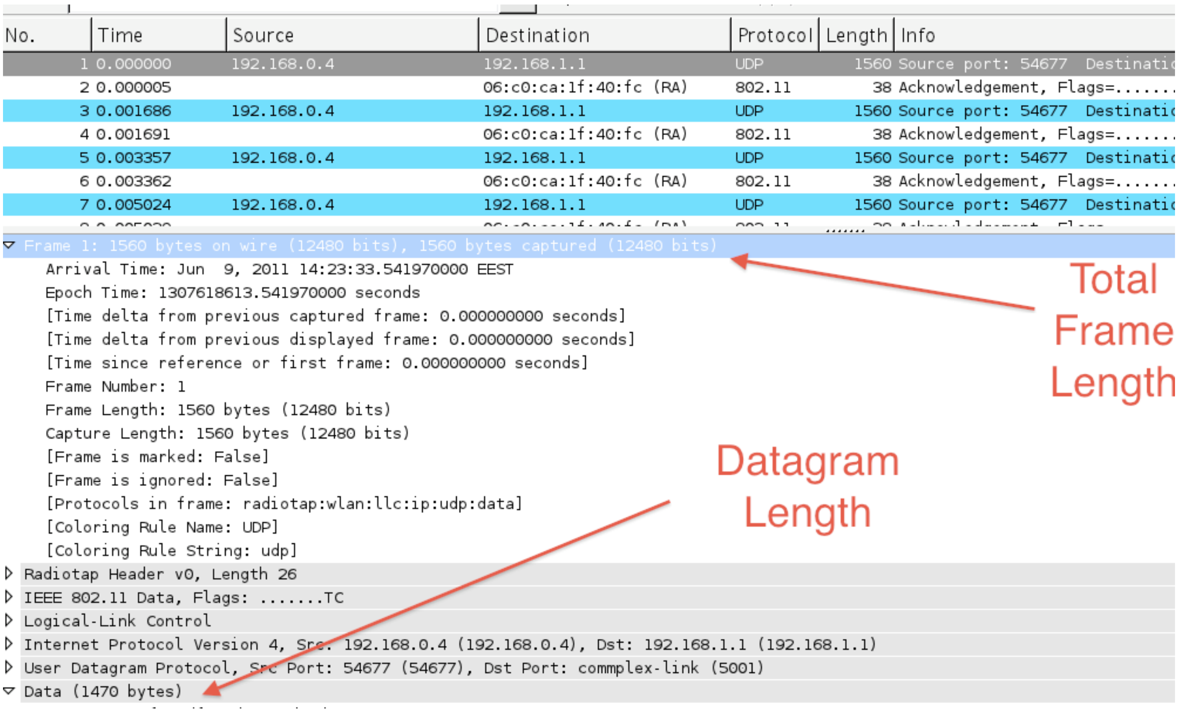


```
No.    Time       Source        Destination              Protocol Length Info
     1 0.000000   192.168.0.4   192.168.1.1              UDP        1560 Source port: 35945  Destinatio
     2 0.000003                 06:c0:ca:1f:40:fc (RA)   802.11       38 Acknowledgement, Flags=.......
     3 0.000491   192.168.0.4   192.168.1.1              UDP        1560 Source port: 35945  Destinatio
     4 0.000492                 06:c0:ca:1f:40:fc (RA)   802.11       38 Acknowledgement, Flags=.......
     5 0.000949   192.168.0.4   192.168.1.1              UDP        1560 Source port: 35945  Destinatio
     6 0.000971                 06:c0:ca:1f:40:fc (RA)   802.11       38 Acknowledgement, Flags=.......
     7 0.001469   192.168.0.4   192.168.1.1              UDP        1560 Source port: 35945  Destinatio
```

```
› Frame 1: 1560 bytes on wire (12480 bits), 1560 bytes captured (12480 bits)
▾ Radiotap Header v0, Length 26
    Header revision: 0
    Header pad: 0
    Header length: 26
  ▷ Present flags
    MAC timestamp: 430779301165
  ▷ Flags: 0x12
    Data Rate: 36.0 Mb/s
    Channel frequency: 5300 [A 60]
  ▷ Channel type: 802.11a (0x0140)
    SSI Signal: -38 dBm
    SSI Noise: -95 dBm
    Antenna: 1
    SSI Signal: 57 dB
› IEEE 802.11 Data, Flags: .......TC
› Logical-Link Control
› Internet Protocol Version 4, Src: 192.168.0.4 (192.168.0.4), Dst: 192.168.1.1 (192.168.1.1)
▾ User Datagram Protocol, Src Port: 35945 (35945), Dst Port: commplex-link (5001)
    Source port: 35945 (35945)
    Destination port: commplex-link (5001)
```

Modulation Rate

Figure 18 "Wireshark screenshot 'Data Rate' "



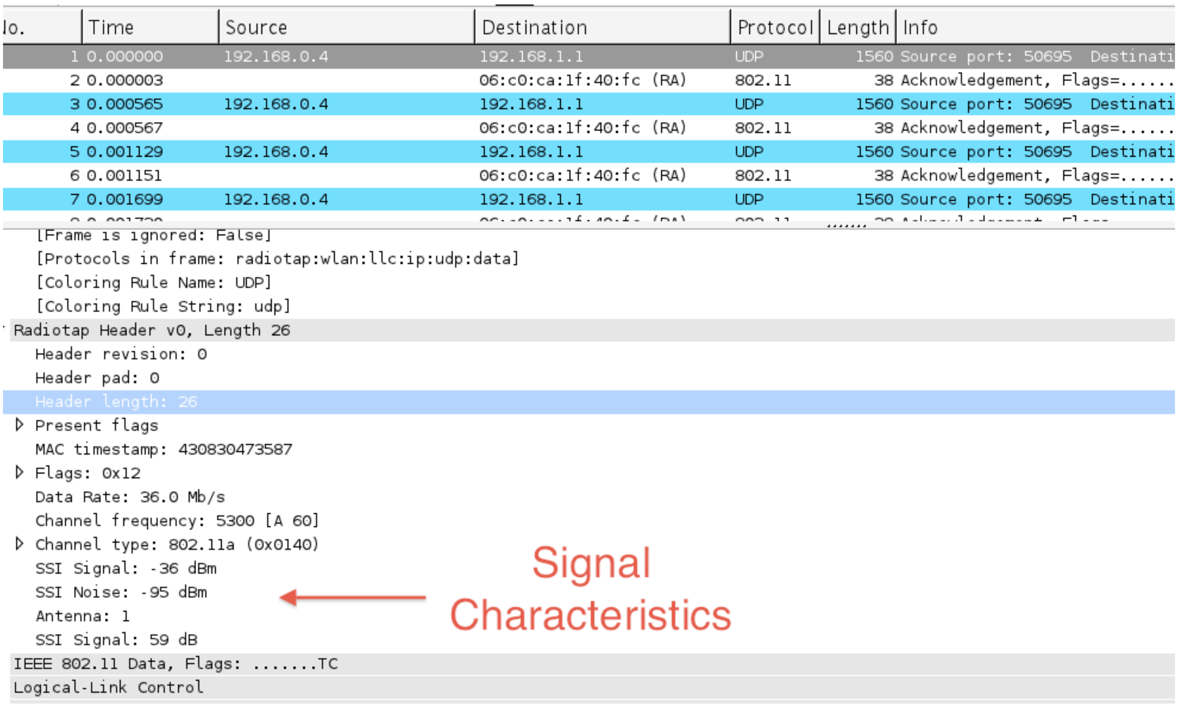
```
No.   Time      Source        Destination              Protocol Length Info
    1 0.000000  192.168.0.4   192.168.1.1              UDP        1560 Source port: 54677  Destinatio
    2 0.000005                06:c0:ca:1f:40:fc (RA)   802.11       38 Acknowledgement, Flags=.......
    3 0.001686  192.168.0.4   192.168.1.1              UDP        1560 Source port: 54677  Destinatio
    4 0.001691                06:c0:ca:1f:40:fc (RA)   802.11       38 Acknowledgement, Flags=.......
    5 0.003357  192.168.0.4   192.168.1.1              UDP        1560 Source port: 54677  Destinatio
    6 0.003362                06:c0:ca:1f:40:fc (RA)   802.11       38 Acknowledgement, Flags=.......
    7 0.005024  192.168.0.4   192.168.1.1              UDP        1560 Source port: 54677  Destinatio
```

```
▽ Frame 1: 1560 bytes on wire (12480 bits), 1560 bytes captured (12480 bits)
    Arrival Time: Jun  9, 2011 14:23:33.541970000 EEST
    Epoch Time: 1307618613.541970000 seconds
    [Time delta from previous captured frame: 0.000000000 seconds]
    [Time delta from previous displayed frame: 0.000000000 seconds]
    [Time since reference or first frame: 0.000000000 seconds]
    Frame Number: 1
    Frame Length: 1560 bytes (12480 bits)
    Capture Length: 1560 bytes (12480 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: radiotap:wlan:llc:ip:udp:data]
    [Coloring Rule Name: UDP]
    [Coloring Rule String: udp]
▷ Radiotap Header v0, Length 26
▷ IEEE 802.11 Data, Flags: .......TC
▷ Logical-Link Control
▷ Internet Protocol Version 4, Src: 192.168.0.4 (192.168.0.4), Dst: 192.168.1.1 (192.168.1.1)
▷ User Datagram Protocol, Src Port: 54677 (54677), Dst Port: commplex-link (5001)
▽ Data (1470 bytes)
```

Total Frame Length

Datagram Length

Figure 19 "Wireshark screenshots 'Datagram Length' "



```
lo.   Time      Source        Destination              Protocol Length Info
    1 0.000000  192.168.0.4   192.168.1.1              UDP        1560 Source port: 50695  Destinati
    2 0.000003                06:c0:ca:1f:40:fc (RA)   802.11       38 Acknowledgement, Flags=......
    3 0.000565  192.168.0.4   192.168.1.1              UDP        1560 Source port: 50695  Destinati
    4 0.000567                06:c0:ca:1f:40:fc (RA)   802.11       38 Acknowledgement, Flags=......
    5 0.001129  192.168.0.4   192.168.1.1              UDP        1560 Source port: 50695  Destinati
    6 0.001151                06:c0:ca:1f:40:fc (RA)   802.11       38 Acknowledgement, Flags=......
    7 0.001699  192.168.0.4   192.168.1.1              UDP        1560 Source port: 50695  Destinati
```

```
   [Frame is ignored: False]
   [Protocols in frame: radiotap:wlan:llc:ip:udp:data]
   [Coloring Rule Name: UDP]
   [Coloring Rule String: udp]
 Radiotap Header v0, Length 26
   Header revision: 0
   Header pad: 0
   Header length: 26
 ▷ Present flags
   MAC timestamp: 430830473587
 ▷ Flags: 0x12
   Data Rate: 36.0 Mb/s
   Channel frequency: 5300 [A 60]
 ▷ Channel type: 802.11a (0x0140)
   SSI Signal: -36 dBm
   SSI Noise: -95 dBm
   Antenna: 1
   SSI Signal: 59 dB
 IEEE 802.11 Data, Flags: .......TC
 Logical-Link Control
```

Signal Characteristics

Figure 20 "Wireshark screenshots 'Tx Power' "

SOFTWARE MANUAL

The algorithm that process the sniffer files wirtten in C. It counts the number of frames transmitted either from the consider station (STA) or the Access point (AP). Then it calculates important parameters as follows:

- T_tx : Time spent in the transmitting state.
- T_rx : Time spent in the receiving state.
- T_id : Time spent in the idling state.
- Transmit Power Level (dB).
- Signal-to-noise Ration (dB).

taking into account the different types of frames.

```
#include <stdio.h>
#include <pcap.h>
#include <stdlib.h>
#include <netinet/ip.h>
#include <arpa/inet.h>
#include <math.h>
#define MAX_PKTS 1000000
#define ACKBYTES 12
#define DATABYTES 1534 //1560-26=1534 equation doesn't work on define
command??
#define RATE 36
#define BASERATE 24
#define CW 16
#define Te 9
#define PLCP 24
#define SIFS 16
#define DIFS 34
//-------------------------------------------------------------------
int main(int argc, char **argv)
{
    unsigned long pkt_counter=0;   // packet counter
    unsigned long current_ts=0; //current timestamp
    unsigned long tref=0;
    unsigned long ttx=0;
    unsigned long trcv=0;
    unsigned long tid=0;
    unsigned int data_tref_vector[MAX_PKTS];
    unsigned int ack_tref_vector[MAX_PKTS];
    unsigned long data_frames=0;
    unsigned long data_tref=0;
    unsigned long data_frames_tx=0;
    unsigned long data_frames_rcv=0;
    unsigned long ack_frames=0;
    unsigned long ack_frames_tx=0;
    unsigned long ack_frames_rcv=0;
    unsigned long ack_tref=0;
    unsigned long beacon_frames=0;
    unsigned long proberqst_frames=0;
    unsigned long probersp_frames=0;
    unsigned long unknown_frames=0;
    unsigned char *MAC_AP="001b11c556a4";
    long sumSignal=0;
    long sumNoise=0;
    int SNR=0;
    //temporary packet buffers
    struct pcap_pkthdr header; // The header that pcap gives us
    const u_char *packet; // The actual packet

    //-------- Begin Main Packet Processing Loop ------------------
    //loop through  pcap file
    //open the pcap file
    pcap_t *handle;
    char errbuf[PCAP_ERRBUF_SIZE]; //not sure what to do with this, oh
well
    handle =
pcap_open_offline("/Users/vassilispapadopoulos/Desktop/Thesis/Bachelor
Thesis Topics/Datagram Length/Soekris/1470Kb/datagram1470.cap", errbuf);
//call pcap library function
    if (handle == NULL) {
      fprintf(stderr,"Couldn't open pcap file %s\n",errbuf);
      return(2);
    }
      //------------------
      //begin processing the packets in this particular file, one at a
time

      while (packet = pcap_next(handle,&header)) {
            // header contains information about the packet (e.g.
timestamp)
            u_char *pkt_ptr = (u_char *)packet; //cast a pointer to the
packet data


            //check to see if the next second has started, for statistics
purposes
            if (current_ts == 0) {  //this takes care of the very first
packet seen
                  current_ts = header.ts.tv_usec;
            } else if (header.ts.tv_usec > current_ts) {
                  tref=header.ts.tv_usec-current_ts;
                  //printf("Timeref=%d usecs\n",tref); //print
                  current_ts = header.ts.tv_usec; //update time interval
            }
            else if (header.ts.tv_usec < current_ts) {
                  current_ts= header.ts.tv_usec;
            }
            int ether_offset = 26;
            unsigned char *MAC_dst=&pkt_ptr[ether_offset+4];
            unsigned char tmp[12];
            int snr_offset = 22;
            char *snr_ratio = &pkt_ptr[snr_offset];
            int signal,noise;
            int i;
            for(i=0;i<6;i++)
            {
                  if(MAC_dst[i]<0x10)
                  {
                        sprintf(&tmp[2*i],"0%x",MAC_dst[i]);
                  }
                  else
                  {
                        sprintf(&tmp[2*i],"%x",MAC_dst[i]);
                  }
            }
            int dst=0;
            for(i=0;i<12;i++)
            {
                  if(tmp[i]==MAC_AP[i])
```

```
        {
                dst=1;
        }
        else
        {
                dst=0;
                break;
        }
}
signal=(int *)snr_ratio[0];
sumSignal+=signal;
noise = (int *)snr_ratio[1];
sumNoise+=noise;
SNR += (signal - noise);

//printf("%d\t%d\n",signal,noise);
int frame_type=pkt_ptr[ether_offset];
//printf("Frame Type=%x: ",frame_type);

switch(frame_type)
{
        case 0x08:
                //printf("DATA Frame, ");
                //printf("%d\t%d\n",signal,noise);
                tid+=((CW-1)/2)*Te+DIFS;
                if(dst==1)
                {
                        data_frames_tx++;
                        ttx+=PLCP+(DATABYTES*8/RATE);
                }
                else
                {
                        data_frames_rcv++;
                        trcv+=PLCP+(DATABYTES*8/RATE);
                }
                data_tref+=tref;
                data_tref_vector[data_frames]=(int)tref;
                data_frames++;
                //printf("ToDS=%x,retry=%x",ToDS,retry);
                //printf("\n");
```

```
            break;
      case 0xd4:
            //printf("ACK Frame, ");
            ack_tref_vector[ack_frames]=(int)tref;

            ack_frames++;
            tid+=SIFS;
            if(dst==1)
            {
                  ack_frames_tx++;
                  ttx+=PLCP+(ACKBYTES*8/BASERATE);
            }
            else
            {
                  ack_frames_rcv++;
                  trcv+=PLCP+(ACKBYTES*8/BASERATE);
            }
            ack_tref+=tref;

            //printf("DS=%x,retry=%x",ToDS,retry);
            //printf("\n");

            break;
      case 0x80:
            //printf("Beacon Frame\n");
            beacon_frames++;
            break;
      case 0x40:
            //printf("Probe Request Frame\n");
            proberqst_frames++;
            break;
      case 0x50:
            //printf("Probe Response Frame\n");
            probersp_frames++;
            break;
      default:
            //printf("Unknown Frame\n");
            unknown_frames++;
            break;
}
pkt_counter++; //increment number of packets seen
```

```
      } //end internal loop for reading packets (all in one file)
      pcap_close(handle);  //close the pcap file
     //end for loop through each command line argument
    //---------- Done with Main Packet Processing Loop --------------



    //printfprintf("--------------------------------------------------------
--------\n");
    printf("Processed %d packets in %d files\n", pkt_counter, argc-1);
    printf("Data Frames=%d\nACK Frames=%d\nUnknown Frames=%d\nBeacon
Frames=%d\nProbe Request Frames=%d\nProbe
Response=%d\n",data_frames,ack_frames,unknown_frames,beacon_frames,prober
qst_frames,probersp_frames);
    printf("---------------Transmitted---------------\n");
    printf("DATA frames=%d\nACK
Frames=%d\n",data_frames_tx,ack_frames_tx);
    printf("Time transmitting=%d ns\n",ttx);
    printf("---------------Received---------------\n");
    printf("DATA frames=%d\nACK
Frames=%d\nBeacons=%d\n",data_frames_rcv,ack_frames_rcv,beacon_frames);
    printf("Time receiving=%d ns\n",trcv);
    printf("--------------Idled------------------\n");
    printf("Time idling=%d ns\n",tid);
    printf("Total time=%d\n",ttx+trcv+tid);
    float t ;
    t = SNR/pkt_counter;
    printf("Average SNR %f",t);
    return 0; //done
} //end of main() function
```