ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Volume 1, Issue 6, August 2013

Implementation of Customized Greencall Algorithm for Energy-Efficient Of Wireless LANs

Dr.K.P.Kaliyamurthie[1], D.Parameswari[2]

Professor and Head, Dept. of IT, Bharath University, Chennai, Tamilnadu, India

Asst. Prof.(SG), Dept. of Computer Applications, Jerusalem College of Engg., Chennai, Tamilnadu, India

ABSTRACT: Emerging dual-mode phones incorporate a Wireless LAN (WLAN) interface along with the traditional cellular interface. The additional benefits of the WLAN interface are, however, likely to be outweighed by its greater rate of energy consumption. This is especially of concern when real-time applications, that result in continuous traffic, are involved. WLAN radios typically conserve energy by staying in sleep mode. With real-time applications like data transfer, this can be challenging since packets delayed above a threshold are lost. Moreover, the continuous nature of traffic makes it difficult for the radio to stay in the lower power sleep mode enough to reduce energy consumption significantly. In this work, I propose the GreenCall algorithm to derive sleep/ wake-up schedules for the WLAN radio to save energy during data transfer while ensuring that application quality is preserved within acceptable levels of users. I evaluate GreenCall on commodity hardware and study its performance over diverse network paths and describe my experiences in the process. I further extensively investigate the effect of different application parameters on possible energy savings through trace-based simulations. I show that, in spite of the interactive, real-time nature of data transfer, energy consumption during calls can be reduced by close to 80 percent in most instances.

KEYWORDS: wireless LANs, energy consumption, portable communication devices, Internet.

I. INTRODUCTION

DUAL-MODE phones like the Apple iPhone and RIM Blackberry are an emerging trend with both a cellular and a Wireless LAN (WLAN) interface. Apart from data access, the WLAN interface can also be leveraged for making packets transfer. This offers two advantages over traditional calling over the cellular interface: 1) calls over the Internet through WLANs are more cost effective and 2) these calls are not affected by lack of coverage of the cellular network in some indoor areas like the office or home, or in certain outdoor areas. The caveat, however, with using the WLAN interface is that now energy consumption is of greater concern. An active or even idle wireless network interface is a significant drain on the relatively limited capacity batteries found in smart phones.

Previous studies suggest that for high end devices like laptops, at least 15-20 percent of the total energy capacity is consumed by an active WLAN interface ,while for low end devices like a PDA, this number increases to about 65 percent of the total energy consumption . Reducing the energy consumed by the WLAN interface for data transfer is thus a critical step toward extending the operating lifetime of these mobile devices when utilized for such applications.
For applications with long periods of inactivity like Network File Systems or those with large tolerable latencies of the order of seconds like web browsing, energy can be saved by letting the WLAN radio stay in the low power sleep mode frequently and for long periods of time. Adopting this approach with real-time applications, however, is more challenging. Data transfer has tolerable latencies of the order of only hundreds of milliseconds, and any greater delay induced by periodic transitions to the sleep mode would degrade the quality of the call beyond the user's tolerable limits. Packet generation intervals of the order of tens of milliseconds exacerbate the situation by making it difficult for the radio to spend any significant amount of time in the sleep mode.

In this paper, I address the issue of reducing energy consumption of the WLAN interface during a packet transfer while preserving the quality within acceptable levels. My approach is based on using an algorithm that can be implemented as

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Volume 1, Issue 6, August 2013

a software solution to save energy. This solution would work with legacy interfaces whose radios may not be as power efficient as those of emerging interfaces. Further, when improved hardware solutions emerge that are more power efficient, our approach would complement such advances. My contributions can be summarized as the following:

1. I propose the GreenCall algorithm that conserves energy during data transfer over WLANs. My algorithm saves energy by computing sleep/wake-up schedules that allow the radio to stay in the low power sleep mode for significant periods of time during acall. These schedules are computed keeping in mind the maximum delay users can tolerate in their conversations. This enables my algorithm to maximize energy consumption while targeting a specified level of application quality. Moreover, the algorithm requires a software upgrade only on the mobile device that desires to save energy.

2. I present extensive evaluations of our algorithm through trace-driven simulations as well as experiments on commodity hardware/software. I evaluate my algorithm on commodity hardware and quantify the energy saved between widespread geographic points of the Internet and describe our experiences of the process. Through trace-based simulations, I further evaluate the effectiveness of our algorithm for various configurable parameters of applications. I demonstrate that, with this algorithm, close to 80 percent energy savings can be achieved on most paths during data transfer. More importantly, from a fundamental perspective, my evaluations show that significant reduction of the energy consumed due to the WLAN interface can be achieved even for real-time interactive applications.

The organization of this paper is as follows: In Section I provide some background on saving energy that is wasted in the idle mode of the WLAN interface and describe related work. Section 3 considers the end-to-end perspective of a data transfer call in saving energy consumed due to the WLAN interface and formulates the problem I address in this paper. In Sections 4 and 5, I develop the solution to our problem and describe how our algorithm computes energy efficient sleep/wake-up schedules. Details of my GreenCall algorithm and its pseudocode are then presented in Section 6. Jitter & Collision are analysed by using Backoff Algorithm in section 7. Section 8 presents the evaluation results of trace-driven simulations that analyze the performance of GreenCall & Backoff over diverse network paths in detail for various application configurations. Concluding remarks are made in Section 9.

II. BACKGROUND AND RELATED WORK

I begin this section by providing some background on how energy wastage by time spent in the idle mode of the WLAN interface can be minimized. Subsequently, I will look at related work that saves energy in the idle mode classified based on the type of traffic under consideration:

Saving Energy Consumed by WLAN Interface

The key idea of saving energy of the wireless interface is to allow it to sleep as much as possible by reducing the time spent in idle mode. There is typically about one order of magnitude difference between the power consumption in the idle and sleep states. This is a difficult problem since the radio may not know when exactly it has to wake up to receive incoming packets and will lose them if it stays in the sleep state. Other researchers have thus proposed schemes that use multi radio solutions. The data and control channels are separated, with the control channel generally using a lower power, always active radio to wake up the higher powered Wireless LAN radio .

A standardized solution to this issue is the Power Save Mode (PSM) which was introduced in the IEEE 802.11 standard for infrastructure WLANs . PSM allows a node to transition to the lower power sleep state when it is not actively sending or receiving packets by notifying the AP. Subsequently, the AP buffers any packets it receives destined for this node. Periodically, a beacon is sent out from the AP that informs all associated nodes if they have any packets buffered through a traffic indication map (TIM). This beacon is sent every beacon interval (BI) and is a preconfigured value at the AP. Each node on receiving notification of buffered packet(s) within a beacon, can leave sleep mode and request buffered packets from the AP. Within these retrieved packets, the AP sets a “more data” bit as long as there are pending packets. The client goes to sleep immediately after it finds no more packets buffered to it.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Volume 1, Issue 6, August 2013

When the client does not want to use PSM anymore, it notifies the AP and the latter does not buffer packets destined to the client anymore. The client's network card consumes much less power while sleeping by shutting off power to all components except for a timing circuit. Because PSM has been part of the standard for many years now, all current deployments support PSM. Using PSM obviates the need for any supplementary devices to reduce the energy consumption of the WNIC. As explained next, there have been important additions to PSM in recent years; thus, throughout this paper, I refer to the scheme outlined above as legacy PSM.

III. PROBLEM DEFINITION

Based on the background provided in the earlier section, I state the problem that needs to be solved in this section. I begin by studying the latency components of a data transfer from an end-to-end perspective and then provide a more formal problem statement.

3.1 PSM to Save Energy during data transfer

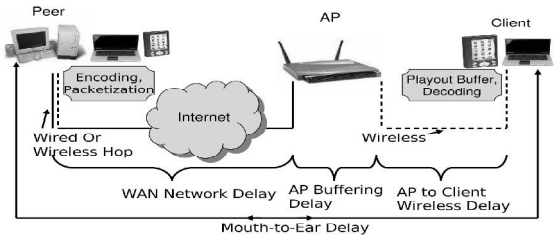


Fig. 1. Illustration of end-to-end latency components of a data transfer between a peer on a wired/wireless network and the client on a wireless network.

The two ends of a data transfer are peers of each other. To simplify the description, I term the device on which energy savings is sought and runs an energy saving algorithm as the client. My descriptions will be based on the perspective of the client. The device on the other end of the client will be referred to as the peer. If both ends are running an energy saving algorithm, they are symmetric for the purposes of my descriptions and either of these can be referred to as the client and other as the peer.

Looking at Fig. 1, if the client uses PSM to go to sleep, any packets arriving from the peer will be buffered at the AP. If the arriving packets were delay-tolerant, the client could sleep for long durations (order of seconds) before collecting packets from the AP. The traffic, however, has small tolerable latencies and each packet must reach the client by its playout deadline. Thus, the client sleep schedules must be precise enough to ensure no packets are lost due to missed playout deadlines.

To calculate such a strict sleep/wake-up schedule, I need to consider the latency (mouth-to-ear delay) of a packet from the peer to the client. It can be broken into different components as illustrated in Fig. 1. The latency from the peer to client's AP is mainly the network delay for the packet once it is sent out from the application layer of the peer station. The peer incurs an encoding and packetization delay before it hands the packet to the network layer. Once a packet reaches the AP, it is buffered there until the client comes out of PSM and is ready to receive the packet. Finally, once the packet reaches the client, it is kept in a playout buffer to reduce jitter on playback. The minimum latency induced by the playout buffer is only the time to decode and playout the packet. When the mouth-to-ear delay exceeds a specific tolerable value (thus, termed tolerable delay), the packet is dropped.

3.2 Problem Statement

The possible packet arrival patterns for two cases at the client—when it does not go to sleep at all, and when it periodically goes to sleep. For simplicity, this illustration and associated description assumes that only the client is trying to save energy. For the case when client periodically transitions to sleep, packets arrive at the client in bursts.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Volume 1, Issue 6, August 2013

The first packet delivered to the client after a sleep period is likely to be closest to its playout deadline compared to the other packets sent by the AP for the same sleep period.

The client needs to calculate a sleep/wake-up schedule that allows it to sleep as much as possible, while ensuring it receives all packets before their playout deadline. Larger sleep periods are preferable to smaller sleep periods as they minimize the total overhead incurred in transitioning to and from the sleep state by notifying the AP each time.

If the client knew the network latency for each packet throughout the data transfer the problem can be easily solved. Unfortunately, network latencies of packets arriving in the future are unknown to the client when it has to calculate its schedule. If the estimate of network latency used by the client is larger than the underlying network latency, it is missing out on opportunities to save energy by sleeping more. On the other hand, packet loss is likely if the underlying network latency increases above any estimate used by the client. To achieve energy savings under the circumstances, I trade off the possibility of some packet loss due to transitions to sleep mode by introducing a parameter LR that specifies the tolerable loss rate of the application. Let $T = \{\delta^1 \ldots \delta^n\}$ be the set of sleep periods used by the client during the data transfer, with n being the number of times it transitions to sleep mode during the call. The energy consumed for this sleep/wake-up schedule can be modeled as

E_τ=_ PtxTtx + PrxTrx + PidleTidle + PsleepTsleep;
where Ps are the known power consumption values and Ts are the total time spent during the entire call in transmit, receive, idle, and sleep states, respectively.

The challenge involved in the design of GreenCall can be better understood by looking at some typical numbers of the parameters involved. Network latency between sites vary from 0 to 1,000 ms based on geographic distance and/ or characteristic of the path. Tolerable latency is widely used as 100-300 ms. Latency to communicate between the client and its AP varies from 0 to 20 ms based on contention on the medium. These numbers vary due to a number of factors which will be discussed at various places throughout the rest of the paper.

IV. DERIVATION OF SLEEP/WAKE-UP SCHEDULES

In this section, we will describe our approach to derive sleep/ wake-up schedules and present our GreenCall algorithm to solve the problem.

4.1 Derivation of Schedules When PSM Used Only by Client

Assume that our energy saving algorithm is running at the client. To calculate sleep periods, the client needs to perform the following three steps: 1) determine playout deadlines of each arriving packet, 2) estimate times at which packets would have been received at the client if it never used PSM, and 3) calculate sleep period for future packets based on difference between the playout deadline and theoretical receive time at client without PSM of previously received packets. I begin by describing the calculation of playout deadlines.

4.1.1 Playout Deadlines

The playout deadline for a packet can be calculated as the sum of tolerable latency from the time the packet content was generated.

4.1.2 Network Latency

Now that the client can calculate the playout deadline of each packet, it requires an estimate of packet network latencies to calculate the receive times for each packet if it were never using PSM. For this, I use the concept of spare time of a packet which is the difference between its playout time and arrival time at client. The spare time of any received packet it can be directly calculated by the client based on the difference between its arrival time and its playout deadline.

4.1.3 Sleep Periods

Finally, I will use the playout deadlines and spare times calculated for the significant packets to derive sleep periods at various decision points through the data transfer. A decision point can be defined as the time when the client has to

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Volume 1, Issue 6, August 2013

calculate the next sleep period to use after receiving packets from the AP buffered during previous sleep period. If the client knew the network latency of the next arriving packet, it can calculate the next sleep period to use.

4.2 Derivation of Schedules when PSM is Used by Both Client and Peer

If the peer also desires to save energy and calculates sleep periods to use with PSM, both the client and peer must ensure that their calculations take into account the fact that each packet might be delayed at both ends. If the network latencies suffered by packets are symmetric in both directions, the sleep periods calculated at both ends should be the same. Under the circumstances, both ends using half the calculated sleep period is a fair way to share the possible sleep times between both ends and ensure packet meet their playout deadlines.

V. BOUNDING NETWORK LATENCY AND ADAPTATION WITH LOSS RATE

In this section, I look at how I can bound future network latencies through a systematic study of actual network latency characteristics. I will also present an adaptive approach to deal with cases where this bound does not hold and ensure that ensuing packet losses are controlled. I begin by giving details of the collection and description of the data sets we use to study how we can predict bounds on network latencies.

5.1 Network Latency Data Sets

I collected multiple WAN latencies and WLAN latencies to an AP. The sum of these gave us traces of end-to-end latency between any two end points. The WAN traces were collected from the University of Massachusetts (UMASS), Amherst, MA to different node locations within and outside US with both end points having Ethernet access to the Internet.

5.2 Network Latency Characteristics

As the data transfer proceeds, estimates of previous network latencies are available for making predictions about bounds on future network latencies. Thus, the first step is to study how accurately previous latencies reflect future network latencies. I studied how often the next network latency exceeds the P^{th} percentile of previous network latencies. Thus, for a given P, I have statistical information on how many times the next latency will exceed the P^{th} percentile value. Instead of considering all previously seen latencies while calculating the P^{th} percentile.

5.3 Adaptation of Latency Bound with Respect to Loss Rate.

Note that the sleep period calculation in Section 4 relies heavily on estimates of many parameters; when these estimates are incorrect, there are either missed opportunities to save energy, or packets are lost. Further, the bounds on latency for a certain value of P may have a larger than expected loss rate for some trace, and it is important to adjust to such scenarios. I, thus, rely on adding or subtracting a shift value S from the current latency bound to control how conservative or aggressive the predicted bound is during the call. If the current loss rate is greater than the user specified loss rate LR, the value of S can be increased. Conversely, when the current loss rate is smaller than user specified loss rate, S can be decreased. This adaptation helps balance the trade-off between energy savings and loss rate.

VI. GREENCALL ALGORITHM

Having described my approach to calculate sleep periods, in this section, I present the complete GreenCall algorithm to derive sleep schedules during the call. Green- Call handles unknown network latencies by keeping track of latencies suffered by previous packets received at client and predicts latency bounds for future network latencies, based on which subsequent sleep periods are calculated. The magnitude of shift value S used for network latency bound depends on the current loss rate. Consequently, at higher loss rates, more conservative sleep periods are used. This enables a smooth trade-off between loss rate and energy savings and is the main feature of the algorithm.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Volume 1, Issue 6, August 2013

The algorithm can be divided into three phases:

```
/*Phase 0: Initialization*/
-Get values of timing constants, application parameters and
 algorithm parameters related to history window H.
- Set counter k = 0.
- Based on feedback from peer about its status use C_share = 0 or 1.
- Obtain estimates l_pc, l_cp, and l_ac
WHILE For each packet i received as call continues
    /*Phase 1: Spare Time Calculation and PSM Execution*/
    IF this is first packet received after a transition to sleep
        - Increment k
        - Calculate spare time t^k_spare
        - Calculate possible new sleep period γ^k = t^k_spare - 2.l_ac
        - Sleep period to use γ^k = γ^k.C_share
        - Transmit packets in send buffer when awake
        IF AP has no more packets buffered for client
            -If γ^k > 0, execute sleep period γ^k
    ELSE
        - Stay awake
    /*Phase 2: Adaptation of Shift Parameter and Possible
      Feedback*/
    IF packet i misses playout deadline
        - Increase packet loss count
    IF i > C_min and i mod C_interval = 0
        IF current packet loss rate > LR - τ1
            - Increase S
            IF peer not using GreenCall
                - Send feedback to increase its l_cp estimate by C_fb
            ELSE
                - Wait till S >= S_max before sending feedback to
                  increase its l_cp estimate by C_fb
        ELSE IF current packet loss rate < LR - τ2
                - Decrease S
                - Send feedback to decrease its l_cp estimate by C_fb
```

Phase 0, deals with the collection of parameters defined by the application as well as tunable parameters of the algorithm. The final step of this phase is to estimate the one way first packet network latencies l_{pc} and l_{cp} between the client and the peer and vice versa, and the one way latency between client and AP, l_{ac}. This is done by sending special control packets (ICMP echo packets) from the client to each of these points to get the RTT. This RTT is then divided by two to be used as a one way latency estimate.

To account for variability, these estimations are collected over a series of 10 packets with the second to largest of these chosen in an attempt to avoid an underestimate of network delays. Since, at this point the client has not begun transitions to sleep mode, these measurements give the true picture of latencies between these points without introducing any delays due to buffering at AP.

Phase 1 begins with the calculation of spare time for packets as the algorithm loops for each packet received until the call continues. Note that this calculation is done only for the first packet after a sleep period as this signifies a decision point of the algorithm from where the next sleep period is derived. At this decision point, only the estimated spare time of this packet is utilized along with values of known application constants. Subsequently, once the AP has no more packets buffered for it, the client goes to sleep When the sleep period is not greater than zero, the client just stays in the constantly awake mode (CAM). To ensure that the client does not interrupt its sleep period to send packets, the client buffers any generated packets until it wakes up. On wakeup, the client contends for the medium with downlink packets from AP to send all its packets.

Phase 2 deals with the adaptation of S. Large values of S will result in more conservative sleep period. On the other hand, if more network losses are tolerable or if the network latency is estimated to vary only slightly over time, I can save more energy by being more aggressive in selecting a sleep period with smaller values of S. To stay within a target loss rate LR, and achieve maximum possible energy savings, the algorithm constantly monitors the current loss rate and adapts the value of S. If the peer is running an energy saving algorithm, the client tries to control its loss rate through adaptation of S. Once the maximum S has been reached, it sends a feedback to the peer for it to increase its estimate of l_{cp} so that future sleep periods take that into account.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Volume 1, Issue 6, August 2013

VII. BACKOFF ALGORITHM

The delay jitter and collision problems are analysed by Backoff algorithm . It is proposed to cope with these problems and improve Quality of Service of data transmission. Simulation results show that WDB can dramatically reduce delay jitter and collision probability, as well as end-to-end delay, while maintaining acceptable drop rate.

WDB introduces WT as the control factor of BI function. In WDB, the BI range window [0, CW] is changed to [W_{il} ,W_{iu}], W_{il} and W_{iu} first increase with WT , then decrease when WT reaches T_{th} . If WT reaches WT_{max} or r reaches $_{max}r$, the backoff procedure is ended and the packet is dropped.

W_{il} : Lower bound of BI_i window.
W_{iu} : Upper bound of BI_i window.
WT_{max} : Denotes a maximum Services Data Unit
BI : Backoff time initial value.
r : $_{max}r$ denotes the retry limit.
Subscript i : the identifier of a STAi.

VIII. EVALUATIONS THROUGH TRAC BASED SIMULATIONS

In this section, I ran our GreenCall algorithm over multiple actual traces of network latencies using a custom-built simulator. The aim was to see how our algorithm adapts as network latencies fluctuate over a period of time, and how the duration of sleep periods corresponds to energy savings over different paths. To quantify the energy savings through the simulator we now rely on an energy model as opposed to actual measurements as in the previous section. I begin by describing our energy model followed by the results of my evaluations.

8.1 Energy Model

I allowed the client to specify an exact sleep period in the simulator. At the end of the call, with being the set of sleep periods used during the call, the energy consumption was found based on the energy model described in (1) in Section 3.

$E\tau = P_{tx}T_{tx} + P_{rx}T_{rx} + P_{idle}T_{idle} + P_{sleep}T_{sleep}$

8.2 Traffic Model

In simulations, the traffic is based on constant packets generated with intervals of 30 or 60 ms. The network latency of these packets is based on gathered delay traces. The trace used had the client and peer each actively transferring for only about 40-50 percent of the time. During the on-time of a node, packets are generated every fixed interval as mentioned above.

8.3 Results

In this section, I present the results obtained by running GreenCall under different settings based on the experimental setup described in the previous section. In this section, I also examine the effect of asymmetric network paths on the playout deadline estimation accuracy of GreenCall. I conclude this section by studying the performance of GreenCall with varying levels of contention in the wireless link between the client and its AP.

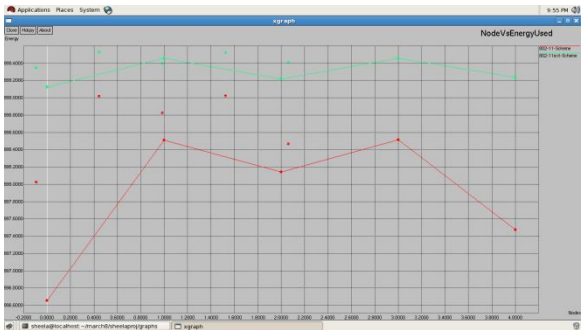
ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Volume 1, Issue 6, August 2013

8.4 Performance Evaulation



IX. CONCLUSIONS

I have addressed the important problem of saving energy for mobile clients due to the wireless interface during data transfer. I presented the GreenCall , WDB algorithm that leverages the IEEE 802.11 PSM mode to save energy consumed by the wireless radio while at the same time ensuring that application quality is preserved. Through extensive evaluations, both through experiments and trace-based simulations over diverse Internet paths, I showed the utility of GreenCall and underscored the great potential of saving energy even with real-time applications. This paves the way for saving energy during active communication with more delay-tolerant traffic like audio and video as well.

REFERENCES

1. Vinod Namboodiri, Member, IEEE, and Lixin Gao, Senior Member, (APRIL 2012) IEEE "Energy-Efficient VoIP over Wireless LANs" IEEE Transactions On Mobile Computing VOL. 9, NO. 4, PP.566-581.
2.Chih-Shun Hsu, Jang-Ping Sheu, Yu-Chee Tseng, "Minimize Waiting Time and Conserve Energy by Scheduling," Telecommunications, 2003. ICT 2003. 10th International Conference on Publication Date: 23 Feb.- 1 March 2003 Volume: 1, On page(s): 393- 399 vol.
3.Fusao NUNO, Ichihiko TOYODA and Masahiro UMEHIRA, (2004) "Performance Evaluation Of Qos Control Scheme That Uses Back Pressure Traffic Control," Personal, Indoor and Mobile Radio Communications.
4.Gupta.A and P. Mohapatra,(2007) "Energy Consumption and Conservation in WiFi Based Phones: A Measurement-Based Study," Proc. IEEE Comm. Soc. Conf. Sensor, Mesh, and Ad Hoc Comm. And Networks (SECON).
5.Jiang, Hongbo; Moore, Andrew; Ge, Zihui; Jin, Shudong and Wang, Jia, Lightweight Application Classification for Network Management, SIGCOMM Workshop on Internet Network Management (INM), August, 2007.
6.Luis, O. Rodolfo, P. Miguel, M. Mario,P. Paulo (September 2007). A Wireless Sensor MAC Protocol For Bursty Data Traffic. IEEE Personal, Indoor and Mobile Radio Communications (PIMRC'07), Athens, Greece, 3-7 .7. Kim. M.-G. , J. Choi, and M. Kang, (Jan. 2008) "Trade-off guidelines for power management mechanism in the IEEE 802.16e MAC," Elsevier Comp. Commun., published online.
7.Mahesri.A and V. Vardhan, "Power Consumption Breakdown on a Modern Laptop," Proc. Power-Aware Computer Systems (PACS), 2004.
8.Markopoulou.A, F. Tobagi, and M. Karam, (June 2006) "Loss and Delay Measurements of Internet Backbones," Computer Comm., vol. 29, no. 10, pp. 1590-1604.
9.Mills.D.L, "Internet Time Synchronization: the Network Time Protocol", IEEE Transactions on Communications, vol. 39, no. 10, October 1991, pp. 1482-1493.
10.Min-Gon Kim, Student Member, IEEE, JungYul Choi, Member, IEEE, and Minho Kang, Senior Member, *(*DECEMBER 2008) IEEE "Scheduled Power-Saving Mechanism to Minimize Energy Consumption in IEEE 802.16e Systems" IEEE COMMUNICATIONS LETTERS, VOL. 12, NO. 12, PP. 874-876
11.Raghunathan.V, T. Pering, R. Want, A. Nguyen, and P. Jensen, "Experience with a Low Power Wireless Mobile Computing Platform," Proc. Int'l Symp. Low Power Electronics and Design (ISLPED), pp. 363-368, 2004.
12.Shih.E, P. Bahl, and M.J. Sinclair, "Wake on Wireless: An Event- Driven Energy Saving Strategy for Battery Operated Devices," Proc. ACM MobiCom, 2002.